Received March 24, 2021, accepted March 27, 2021, date of publication March 30, 2021, date of current version April 6, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3069888

Transient Process Optimization for Dual-Arm Cluster Tools With Wafer Revisiting

JIPENG WANG[1], (Student Member, IEEE), HESUAN HU[1,2], (Senior Member, IEEE), CHUNRONG PAN[3], (Senior Member, IEEE), AND LIANG LI[4]
[1]School of Mechano-Electronic Engineering, Xidian University, Xi'an 710071, China
[2]State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710054, China
[3]School of Mechanical and Electrical Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China
[4]School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan 430081, China

Corresponding authors: Hesuan Hu (huhesuan@gmail.com) and Chunrong Pan (crpan@jxust.edu.cn)

This work was supported in part by the Distinguished Professor Program of Jinggang Scholars in Institutions of Higher Learning, Jiangxi, China; in part by the Natural Science Foundation of China under Grant 51665018, Grant 61573265, and Grant 61973242; in part by the Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China under Grant ICT2021B37; and in part by the Natural Science Foundation of Shaanxi Province through the Major Fundamental Research Program under Grant 2017ZDJC-34.

ABSTRACT In wafer fabrication, it is imperative to minimize the transient process of cluster tools for the sake of on-demand and preventive maintenance. Due to the trend of multi-type and small-batch production, transient processes appear more and more frequently. Thus, the optimization problems of transient processes have gained increasing attention from both industry and academia. The requirement for wafer revisiting tend to complicate this problem significantly. However, only a few studies take such a challenge for cluster tools with wafer revisiting. This paper focuses on the schedule optimization of transient processes for dual-arm cluster tools with wafer revisiting. To accelerate transient processes, including both start-up and close-down ones, we adopt a program evaluation and review technique to analyze and harness a cluster tool's state evolution. We then propose computationally efficient algorithms to speed up transient processes. Finally, we provide illustrative examples to show their applications and validate their effectiveness.

INDEX TERMS Cluster tool, semiconductor manufacturing, wafer fabrication, scheduling, transient process, wafer revisiting.

NOMENCLATURE

$\mathbb{N}_n$	$\{1, 2, \cdots, n\}$.
$\mathbb{N}_j^i$	$\mathbb{N}_i \setminus \mathbb{N}_j$.
Ω_n	$\mathbb{N}_n \cup \{0\}$.
PM	Process module.
SACT	Single-arm cluster tool.
DACT	Dual-arm cluster tool.
ALD	Atomic layer deposition.
WRP	Wafer revisiting process.
k-WRP	k-time Wafer revisiting process.
1-WCS	One-wafer cyclic scheduling.
3-WCS	Three-wafer cyclic scheduling.
VWM	Virtual wafer method.
SUTP	Start-up transient process.
CDTP	Close-down transient process.
PERT	Program evaluation and review technique.
Θ	A state at which a PM or the robot is empty.
M	The system state.
$W_p(q)$	The pth wafer that is released to the system with its qth operation being processed, $p \in \mathbb{N}_n$, $q \in \mathbb{N}_{2k+1}$, $k \in \mathbb{N}_1^5$.
$R_b(W_p(q))$	The pth wafer is being held by the robot and will be delivered into process step b for its qth operation, $b \in \Omega_3$, $p \in \mathbb{N}_n$, $q \in \mathbb{N}_{2k+1}$, $k \in \mathbb{N}_1^5$.
$\mathcal{S}_d =$	$\{W_p(q)\}$, $d \in \mathbb{N}_3$, $p \in \mathbb{N}_n$, $q \in \mathbb{N}_{2k+1}$, $k \in \mathbb{N}_1^5$.
$\mathcal{S}_4 =$	$\{R_b(W_p(q))\}$, $b \in \Omega_3$.
α	Time taken for the robot to unload a wafer from the PM or loadlock.
β	Time taken for the robot to load a wafer into the PM or loadlock.
μ	Time taken for the robot to move among two different modules.
λ	Time taken for the robot to execute swapping operation at a process step.

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Zakarya.

$\omega_{i,j}$	Robot waiting time before its unloading operation at process step i during the transit from M_{j-1} to M_j.
a_i	Wafer processing time in PM_i, $i \in \mathbb{N}_3$.
T_k	Time taken from the idle state at Node 0 to the terminal state at Node k in the PERT model of the SUTP.
Γ_k	Time taken from the initial state at Node 1 to the terminal state at Node k in the PERT model of the CDTP.
ϕ_O	Time taken from the idle state to the first target steady state by the PERT-based method.
ϕ_V	Time taken from the idle state to the first target steady state by the VWM-based method.
Φ_O	Time taken for completing the CDTP by the PERT-based method.
Φ_V	Time taken for completing the CDTP by the VWM-based method.

I. INTRODUCTION

To achieve higher quality, productivity, and yield, cluster tools with the single-wafer processing technology are widely applied in numerous semiconductor wafer fabrication processes, such as etching, chemical vapor deposition, and rapid processing technology with high temperature. In particular, cluster tools occupy an essential place in large-size wafer fabrication. A cluster tool compactly integrates several process modules (PMs), input/output loadlocks, and a wafer-handling robot with a radial way and holds no intermediate buffer. After being loaded into a cluster tool through loadlocks, raw wafers are delivered into PMs for processing in sequence according to pre-specified order, and finally return to loadlocks when all necessary processes are completed. All these operations are dominated by the wafer-handling robot. The robot is equipped with one or two arms fixed in an opposite direction, leading to the single-arm cluster tool (SACT) and dual-arm cluster tool (DACT, see Fig. 1).

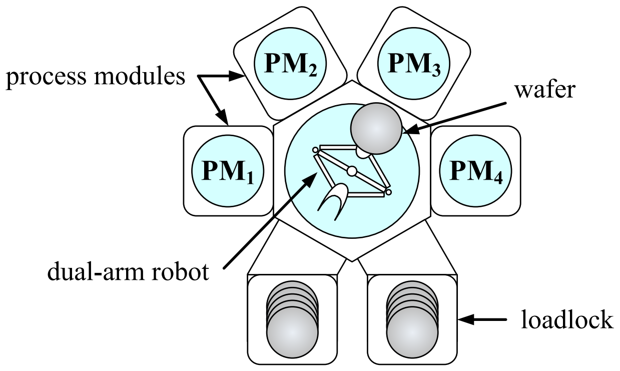


FIGURE 1. A dual-arm cluster tool with four PMs.

Considering their vast investment and production cost, it is of great importance to seek efficient and practical scheduling and control methodologies for cluster tools [1]. However, there is so far no generic solution due to coupling constraints. For instance, in the wafer fabrication process of low-pressure chemical vapor deposition, the completed wafer must be unloaded from the PM within a restricted time in case of its degradation. This temporal restriction is called the wafer residency time constraint (WRTC). Kim *et al.* [2] propose a systematic modeling, analysis, and scheduling method for the DACT with WRTCs. Lee and Park [3] derive necessary and sufficient conditions to verify the schedulability of the time-constrained DACT by using the negative event graph. Based on the resource-oriented Petri nets (ROPNs) model, Wu *et al.* [4], Wu and Zhou [5] present efficient algorithms to find the optimal steady state periodic schedule for the SACT and DACT with WRTCs, respectively. When the residency time is tight, there may not be a feasible schedule under conventional robot strategies. To deal with such a particular issue, Lim *et al.* [6] develop a novel robot strategy based on the interference-free sequence. Yang *et al.* [7] provide a general framework to find the optimal schedule for the time-constrained SACT within both process- and transport-bound regions. In order to reduce quality variability in wafer fabrication and guarantee high-quality output, Zhu *et al.* [9], [52] and Xiong *et al.* [10] propose efficient methodologies to regulate robot waiting times for reducing the wafer delay time in each process step.

Due to exceptional events, such as processing delay, communication delay, or wafer alignment failure, the activity time is practically subject to random variation [2]. The activity time variation (ATV) may make the wafer residency time delay in PMs exceed reasonable bounds such that the schedule obtained under the assumption of deterministic activity time become infeasible. Thus, it requires that the schedule for cluster tools should possess sufficient adaptiveness and robustness [11]–[15]. Kim and Lee [16] develop a necessary and sufficient condition for identifying the always schedulable case and never schedulable case. A graph-based computational procedure is presented to find the satisfied schedule if the system state belongs to the always schedulable case. However, the criterion concerning the never schedulable case is somehow conservative. Wu and Zhou [17]–[19] design a real-time control strategy to dynamically regulate the robot waiting time to maximally compensate the impact of activity time variations on the wafer residency time fluctuation in PMs. By using this strategy, some never schedulable cases identified in [16] are schedulabe in fact. Besides, a real-time scheduling approach with two-level operational architecture is proposed, exhibiting its optimality in terms of productivity. Based on the idea of [17]–[19], Qiao *et al.* [20], [21] present efficient algorithms for scheduling the SACT with WRTCs and ATV. Nevertheless, the upper bound of wafer residency time delay provided in [20], [21] is overestimated, which may impede the schedulability test of some cases. Pan *et al.* [22] obtain the exact bound by several polynomial algorithms. With the unfixed conventional backward strategy, Yang *et al.* [23] provide an efficient method to calculate the upper bound of wafer residency time delay for the SACT with WRTCs and ATV. By the mixed-integer programming model, Lim *et al.* [24] present an adaptive scheduling approach to cluster tools with tight wafer residency time constraints and large processing time variations.

In the above discussion, we focus on the cluster tool with nonrevisiting processes, whereas there are still quite a few revisiting ones. For instance, the atomic layer deposition (ALD) is a typical wafer revisiting process (WRP) in semiconductor manufacturing [25]. Wafers in the ALD process require to visit some process steps multiple times such that the film thickness can be precisely controlled. Due to the wafer revisiting, PMs are shared by multiple operations, being prone to deadlocks, which further complicates the scheduling and control of cluster tools. Lee and Lee [26] investigate the scheduling problem of SACTs with WRP for the first time. They develop a mixed-integer programming model to find the deadlock-free optimal schedule. However, the presented method is computationally inefficient due to its exponential complexity. To cope with this issue, Wu *et al.* [27] adopt ROPNs to describe the ALD process. They propose a necessary and sufficient deadlock avoidance policy and analytical expression to calculate the optimal schedule. Further, Yang *et al.* [28] develop efficient algorithms to find the optimal scheduling of SACTs with WRP and WRCTs.

For the DACT with WRP, Wu *et al.* [29] find that the system presents a three wafer cyclic process including three local and global cycles if the conventional swap strategy is applied. Based on the ROPN model, they obtain conditions to find the optimal 3-wafer cyclic schedule (3-WCS). Due to the delay at the revisiting process step in each switching operation from local to global cycle, the system may always be in the transient state, i.e., the lower bound of the systematic cycle time cannot be reached. By reducing the number of local and global cycles, Wu *et al.* [30] propose a 2-WCS method. Qiao *et al.* [31] extend the results in [29], [30] and propose a method to calculate the cycle time of the DACT with multiple revisiting times. As verified in [29]–[31], it remains inefficient that scheduling the DACT with ordinary swap strategy, especially the revisiting time $k > 2$ cases. To overcome this limitation, Qiao *et al.* [32] present a modified swap-based strategy and derive a 1-WCS. Based on the results in [29], [30], [32], Qiao *et al.* [33]–[36] propose efficient methods to find the optimal 1-WCS of DACTs under WRCTs and ATV, respectively.

It should be noted that the majority of studies mentioned-above are devoted to the steady state scheduling. Nevertheless, as the wafer lot size contracts continuously, transient processes scheduling plays an increasingly important role due to new wafer fabrication requirements such as lot switching operations [37]–[39] and concurrent processing of multiple wafer types [40]–[44]. For the DACT, Kim *et al.* [45] prove that the latest/earliest starting policy can minimize its start-up/close-down transient process (SUTP and CDTP for short), respectively. Kim *et al.* [46] further present a max-plus algebra method to optimize the transient process of DACTs. Based on the ROPN model, Qiao *et al.* [47] and Zhu *et al.* [48] propose efficient algorithms to find the optimal schedule of SACTs with WRTCs during the SUTP and CDTP, respectively. Note that in [47], [48], each process step is configured with only one PM. Considering the SACT with parallel PMs, Kim *et al.* [49] develop a generalized backward strategy. Subsequently, Yang *et al.* [50] extend the results in [49] and present linear programs to search the optimal feasible schedule for the SACT with WRTCs and parallel PMs. For the SACT with a failure CDTP, Qiao *et al.* [51] propose efficient response policies. For the linear dual-arm multi-cluster tool subject to WRTCs, Zhu *et al.* [52] present efficient algorithms to find the optimal integrated schedule for the whole process covering the steady state and transient processes.

Although great efforts [45]–[52] concerning the transient process scheduling have been conducted, these studies are unfortunately focused on the nonrevisiting processes, i.e., inapplicable to the transient process with wafer revisiting. In our previous work [53], we propose efficient algorithms to optimize the SUTP of DACTs with WRP for two times. As for the optimization to more general cases, for instance, transient processes with wafer revisiting for multiple times, including both the SUTP and CDTP, there is no research reported on this problem yet. Thus, our major motivation is to tackle the transient optimization problem for DACTs with wafer revisiting. In this paper, we build an analysis framework based on the program evaluation and review technique to investigate the temporal properties of DACTs with WRP during transient processes. With the system network model, we present computationally efficient algorithms to find the optimized transient process schedule for DACTs under diverse wafer revisiting cases.

The subsequent sections are organized as follows. In Section II, we briefly introduce the ALD process and notations of corresponding activity representations and definitions. In Section III and IV, we analyze the SUTP and CDTP comprehensively and propose optimization algorithms to minimize transient processes, respectively. Section V provides case studies. Finally, Section VI concludes this paper.

II. PRELIMINARIES

A. ATOMIC LAYER DEPOSITION PROCESS

In semiconductor manufacturing, the raw wafer needs to undergo a number of process steps. In general, each process step demands a unique operation. If the wafer fabrication process necessitates the visit of the same process steps for more than one time, this is called revisiting process; otherwise, it is called nonrevisiting one. According to the manufacturing requirement, the revisiting process may contain only one process step or more than two process steps. In practice, the revisiting process with two process steps is widely applied in wafer fabrication. ALD is such a typical revisiting process that the film thickness can be controlled by repeating the deposition operation. Since the ALD with two revisiting steps is a typical wafer fabrication process that is commonly adopted in cluster tools, the results derived from the two-step revisiting process

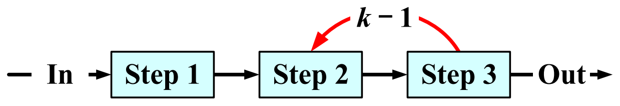


FIGURE 2. Wafer flow for the ALD process.

are useful for the other cases. Therefore, for the sake of simplicity, this paper only consider the ALD with two-step revisiting process. In the ALD process, as shown in Fig. 3, there are generally three steps (i.e., Al_2O_3 deposition, Ta_2O_5 deposition, and oxidation), and the last two of them are the revisiting process, which will be repeated several times, even more than five times [26]. Let m_i and k be the number of parallel PMs for process step i and the number of revisiting times, respectively. Then, the ALD process can be denoted as $(m_1, (m_2, m_3)^k)$, where $(m_2, m_3)^k$ indicates the revisiting process. For the sake of the consistency of wafer fabrication within the ALD process, each step normally is composed of a single PM, i.e., $m_1 = m_2 = m_3 = 1$. Thus, the wafer flow pattern of the ALD process can be denoted as $(\text{PM}_1, (\text{PM}_2, \text{PM}_3)^k)$ with $(\text{PM}_2, \text{PM}_3)^k$ being a k-time wafer revisiting process (k-WRP).

B. DESCRIPTION OF SYSTEM STATE AND ACTIVITY

Cluster tools are a type of highly automated manufacturing systems containing complex discrete state evolution and temporal properties. Let $\mathbb{N}_j^i = \mathbb{N}_i \setminus \mathbb{N}_j$ and $\Omega_n = \mathbb{N}_n \cup \{0\}$, where $\mathbb{N}_n = \{1, 2, \cdots, n\}$. For the state description, we use $W_p(q), p \in \mathbb{N}_n, q \in \mathbb{N}_{2k+1}, k \in \mathbb{N}_1^5$, to represent the pth wafer that is released to the system with its qth operation being processed, and $R_b(W_p(q))$ represents the pth wafer is being held by the robot and will be delivered into process step b for its qth operation, where $b \in \Omega_3$, $p \in \mathbb{N}_n$, $q \in \mathbb{N}_{2k+1}$, and $k \in \mathbb{N}_1^5$. In particular, process step 0 denotes the loadlock. Then, a state of the system can be denoted as $M = \{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4\}$, where $\mathcal{S}_d = W_p(q), d \in \mathbb{N}_3$, and $\mathcal{S}_4 = R_b(W_p(q))$. For instance, $M = \{W_3(1), W_2(2), W_1(3), R_1(W_4(1))\}$ represents that the first, second, and third wafers (i.e., W_1, W_2, and W_3) are being processed in PM_3, PM_2, and PM_1, respectively, whereas the fourth wafer is being held by the robot and will be delivered into PM_1 for its first operation at process step 1. There is no doubt that the first operation must be executed in PM_1, i.e., the robot is at PM_1, preparing for swapping. Therefore, the definition of the system state results in no confusion or misunderstanding. As for the temporal aspects of the system resources' activities, we use α, β, μ, and λ to denote the robot task time for unloading, loading, moving, and the swapping operation, respectively. Likewise, wafer processing time in PM_i is indicated as a_i, $i \in \mathbb{N}_3$.

C. CYCLIC SCHEDULING OF STEADY STATE

The SUTP starts from the idle state $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$, where Θ indicates that the PM or the robot arm is empty. The challenge lies in reaching the target state quickly as possible. The CDTP starts from the steady state under a given cyclic schedule. That is, the steady state schedule has a significant impact on both the SUTP and CDTP. Therefore, it is necessary to know the cyclic scheduling strategy under the steady state. For the DACT with 2-WRP, a 1-WCS method is proposed in [32]. As for k-WRP with $k > 2$, it remains unclear whether exist an unified 1-WCS. Instead, Qiao *et al.* [31] investigate this with the framework of 3-WCS.

According to the 1-WCS presented in [32] for DACTs with 2-WRP, in the steady state, the system starts from state $M_1 = \{W_3(1), W_1(4), W_2(3), R_1(W_4(1))\}$ and then evolves as $M_1 \rightarrow M_2 = \{W_4(1), W_1(4), W_2(3), R_2(W_3(2))\} \rightarrow M_3 = \{W_4(1), W_3(2), W_2(3), R_3(W_1(5))\} \rightarrow M_4 = \{W_4(1), W_3(2), W_1(5), R_2(W_2(4))\} \rightarrow M_5 = \{W_4(1), W_2(4), W_1(5), R_3(W_3(3))\} \rightarrow M_6 = \{W_4(1), W_2(4), W_3(3), R_1(W_5(1))\}$. Note that the process $M_3 \rightarrow M_4 \rightarrow M_5$ forms a cycle characterizing the wafer revisiting process, namely so-called the local cycle, whereas the remaining states form a global cycle containing the entire process steps. The evolution from M_1 to M_6 forms a one-wafer cyclic process, containing one local and one global cycle.

With the 3-WCS approach proposed in [31] for DACTs with k-WRP where $k > 2$, in the steady state, the system should start from state $M_1 = \{W_3(1), W_2(2), W_1(3), R_1(W_4(1))\}$. Then, the system state evolves as $M_1 \rightarrow M_2 = \{W_4(1), W_3(2), W_1(3), R_3(W_2(3))\} \rightarrow M_3 = \{W_4(1), W_1(4), W_2(3), R_3(W_3(3))\} \rightarrow M_4 = \{W_4(1), W_2(4), W_3(3), R_3(W_1(5))\} \rightarrow M_5 = \{W_4(1), W_3(4), W_1(5), R_3(W_2(5))\} \rightarrow M_6 = \{W_4(1), W_1(6), W_2(5), R_3(W_3(5))\} \rightarrow \cdots \rightarrow M_{3k-1} = \{W_4(1), W_3(2k), W_1(2k+1), R_3(W_2(2k+1))\} \rightarrow M_{3k} = \{W_4(1), W_3(2k), W_2(2k+1), R_1(W_5(1))\} \rightarrow M_{3k+1} = \{W_5(1), W_4(2), W_3(2k+1), R_1(W_6(1))\} \rightarrow M_{3k+2} = \{W_6(1), W_5(2), W_4(3), R_1(W_7(1))\} \rightarrow M_{3k+3} = \{W_7(1), W_6(2), W_4(3), R_3(W_5(3))\}$. It is obvious that M_1 and M_{3k+2}, and M_2 and M_{3k+3} are equivalent, respectively. This means that the evolution from M_1 (M_2) to M_{3k+2} (M_{3k+3}) forms a periodic work cycle. It should be noted that the state transit from M_3 to M_{3k-1} involves the revisiting process, whereas the others do not, such as transit from M_1 to M_2 or from M_{3k} to M_{3k+3}. Therefore, the state transformation from M_1 to M_{3k+2} or from M_2 to M_{3k+3} forms a periodic work cycle containing $3k - 3$ local cycles and three global ones.

D. SCHEDULING STRATEGY OF TRANSIENT PROCESSES

Assume that the cluster tool operates with a cyclic schedule during the steady state. When the system reaches its full work cycle, we must ensure the state is compatible with the steady state schedule. That is, the SUTP scheduling shall adapt to the steady state. Similarly, the CDTP is subject to the steady state scheduling. As revealed in [1], [4], [5], [47], [48], [51], the virtual wafer method (VWM) has advantages in a variety of scheduling problems of cluster tools, such as steady state scheduling implementation, PM failure response, and transient processes scheduling. Within the VWM-based scheduling framework, each PM is assumed to be occupied by a virtual wafer when the cluster tool boots up, and the virtual wafer will be loaded into the tool system following the last actual wafer W_n. In both the SUTP and CDTP, the system will be manipulated by the cyclic scheduling as it is executed in the steady state. In this way, cluster tools can be efficiently operated in accordance with the steady state during both the SUTP and CDTP.

The virtual wafer scheduling method provides a simple and efficient implementation framework for the transient process. However, some extra activities containing both the robot and PMs are performed due to the processing of virtual wafers. In other words, we can remove these redundant activities by manipulating the actual wafers only; that is, no virtual wafer is loaded into the system. There is no doubt that the transient process can be accelerated after eliminating unnecessary activities. To analyze temporal properties during the transient processes, we adopt a network technique, namely the program evaluation and review technique (PERT) that has been applied in the transient process scheduling for cluster tools [45]. In the PERT paradigm, the precedence relationships of the robot tasks and PMs activities can be expressed graphically as a network model. This means that we can find the optimized transient schedule by searching the critical path of the network. Consequently, in the subsequent parts of this paper, we will adopt the PERT-based approach to conduct the property and scheduling analysis for the DACT with WRP during transient processes, including both the SUTP and CDTP.

III. START-UP TRANSIENT PROCESS SCHEDULING

Within the scheduling framework of 1-WCS and 3-WCS proposed in [31], [32], the target steady state of DACTs with 2-WRP is $\{W_3(1), W_1(4), W_2(3), R_1(W_4(1))\}$, while for the case of k-WRP with $k > 2$ it is $\{W_3(1), W_2(2), W_1(3), R_1(W_4(1))\}$. For the scheduling and control of DACTs with WRP in the SUTP, the crucial problem is how to reach the first target steady state from the idle state in the shortest time. We will discuss how to achieve this goal in subsequent parts.

A. 2-WRP

For 2-WRP, the SUTP consists of two stages. One is the initial stage from the idle state to the first full work cycle state; the other is the regulation one from the first full work cycle state to the target steady state. In the initial stage, the system state evolves as $M_0 = \{\Theta, \Theta, \Theta, R_0(\Theta)\} \rightarrow M_1 = \{W_1(1), \Theta, \Theta, R_1(\Theta)\} \rightarrow M_2 = \{W_2(1), W_1(2), \Theta, R_2(\Theta)\} \rightarrow M_3 = \{W_3(1), W_2(2), W_1(3), R_3(\Theta)\}$. As indicated by [53], due to the difference of wafer processing time between PM$_2$ and PM$_3$, in the regulation stage, there are two evolution paths: one (denoted as Path A) is $M_{41} = \{W_3(1), W_1(4), W_2(3), R_3(\Theta)\} \rightarrow M_{51} = \{W_3(1), W_1(4), W_2(3), R_1(W_4(1))\} \rightarrow M_{61} = \{W_4(1), W_3(2), W_2(3), R_3(W_1(5))\} \rightarrow M_{71} = \{W_4(1), W_2(4), W_1(5), R_3(W_3(3))\} \rightarrow M_{81} = \{W_4(1), W_2(4), W_3(3), R_1(W_5(1))\}$, the other (denoted as Path B) is $M_{42} = \{W_3(1), W_1(4), W_2(3), R_2(\Theta)\} \rightarrow M_{52} = \{W_3(1), W_1(4), W_2(3), R_1(W_4(1))\} \rightarrow M_{62} = \{W_4(1), W_3(2), W_2(3), R_3(W_1(5))\} \rightarrow M_{72} = \{W_4(1), W_2(4), W_1(5), R_3(W_3(3))\} \rightarrow M_{82} = \{W_4(1), W_2(4), W_3(3), R_1(W_5(1))\}$.

We use $\omega_{i,j}$ to indicate the robot waiting time before its unloading operation at process step i during the transit from M_{j-1} to M$_j$. Then, to reach M_3 from M_0, the robot performs the following activities: ⟨unloading raw W_1 from the loadlock → moving to PM$_1$ → loading $W_1(1)$ into PM$_1$ → moving to the loadlock → unloading raw W_2 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,2}$) for $W_1(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → loading $W_1(2)$ into PM$_2$ → moving to the loadlock → unloading raw W_3 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,3}$) for $W_2(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,3}$) for $W_1(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → loading $W_1(3)$ into PM$_3$⟩.

To reach M_{81} from M_3 through Path A, the robot performs the following activities: ⟨waiting ($\omega_{3,4}$) for $W_1(3)$ at PM$_3$ → unloading $W_1(3)$ from PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,4}$) for $W_2(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → loading $W_2(3)$ into PM$_3$ → moving to the loadlock → unloading raw W_4 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,5}$) for $W_3(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,6}$) for $W_1(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,6}$) for $W_2(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,7}$) for $W_3(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,7}$) for $W_1(5)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_1 into the loadlock → unloading raw W_5 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,8}$) for $W_4(1)$ at PM$_1$⟩.

To reach M_{82} from M_3 through Path B, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,4}$) for $W_2(2)$ at PM$_2$ → unloading $W_2(2)$ from PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,4}$) for $W_1(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → loading $W_1(4)$ into PM$_2$ → moving to the loadlock → unloading raw W_4 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,5}$) for $W_3(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,6}$) for $W_1(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,6}$) for $W_2(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,7}$) for $W_3(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,7}$) for $W_1(5)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_1 into the loadlock → unloading raw W_5 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,8}$) for $W_4(1)$ at PM$_1$⟩.

We use U, L, M, and S to denote the robot activities of unloading, loading, moving, and swapping operation, respectively. Similarly, P$_i$, $i \in \mathbb{N}_3$, indicates the wafer fabrication in PM$_i$. Based on the above robot tasks and PM activity sequences, we can build the PERT model of the transient process scheduling as shown in Fig. 3. In this PERT model, Nodes 0, 1_L, 3_L, and 6_L respectively correspond to states M_0, M_1, M_2, and M_3. For the Path A, Nodes 8_1L, 4_1U, 8_1U, 11_1U, and 9_1U represent states M_{41}, M_{51}, M_{61}, M_{71}, and M_{81}, respectively. For the Path B, Nodes 8_2L, 4_2U, 7_2U,

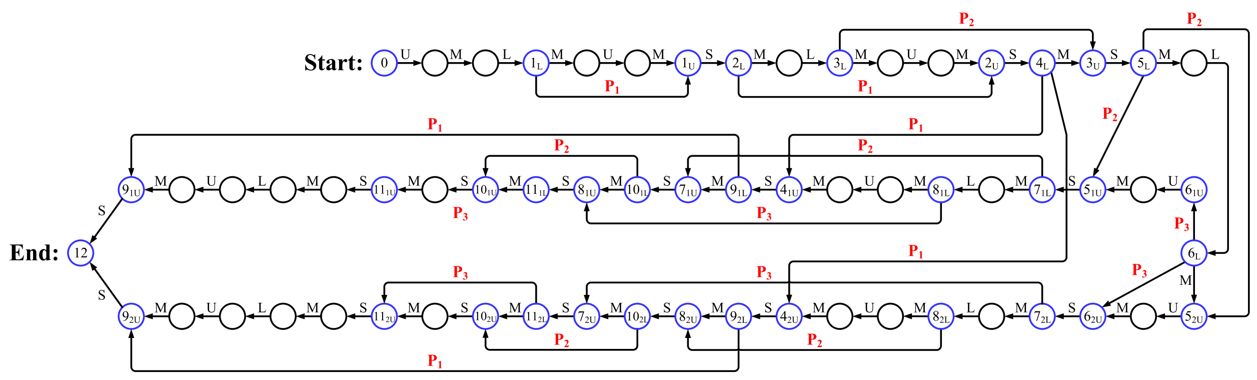


FIGURE 3. PERT model for the SUTP with 2-WRP.

11_{2U}, and 9_{2U} represent states M_{42}, M_{52}, M_{62}, M_{72}, and M_{82}, respectively.

Let ϕ_O denote the minimum time taken for the DACT with WRP to reach the first target steady state from the idle state. Correspondingly minimum taken time based on the VWM is denoted as ϕ_V. With the PERT model, we can calculate ϕ_O by searching the critical path in the network. We use T_k to indicate the time taken from the idle state $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$ (represented by Node 0) to the terminal state (Node k). Observing the system PERT model, our eventual goal is to identify which secondary end Node (i.e., Nodes 9_{1U} and 9_{2U}) will be reached at first. By searching the PERT model, we can recursively calculating T_k from the initial Node 0 to the final node. From Node 0 to Node 1_L, there is only one operation sequence (represented by U → M → L). This takes at least $\alpha + \mu + \beta$ time units. Then, we have $T_{1_L} = T_0 + \alpha + \mu + \beta$. However, to reach its succeeding node (i.e., Node 1_U), there are two operation sequences. One executes wafer processing in PM$_1$, taking a_1 time units; the other performs a series of robot tasks (represented by M → U → M), taking $2\mu + \alpha$ time units. Thus, we have $T_{1_U} = \max\{T_{1_L} + a_1, 2\mu + \alpha\}$. Similarly, we can successively calculate $T_{2_U}, T_{3_U}, \cdots, T_{9_{1U}}$, and $T_{9_{2U}}$. Finally, we have $\phi_O = T_{12} = \min\{T_{9_{1U}}, T_{9_{2U}}\} + \lambda$. Therefore, we have the following algorithm.

B. *k*-WRP WITH *k* > 2

Due to the multiple revisiting processes, the state evolution of DACTs with k-WRP during the SUTP is more complex than 2-WRP. Under the operation mechanism of 3-WCS, it can be divided into three main stages from the idle state to the first target steady state. In the first stage, the system state evolves as $M_0 = \{\Theta, \Theta, \Theta, R_0(\Theta)\} \rightarrow M_1 = \{W_1(1), \Theta, \Theta, R_1(\Theta)\} \rightarrow M_2 = \{W_2(1), W_1(2), \Theta, R_2(\Theta)\} \rightarrow M_3 = \{W_3(1), W_2(2), W_1(3), R_3(\Theta)\} \rightarrow M_4 = \{W_4(1), W_1(4), W_3(2), R_3(W_2(3))\}$. When M_4 is reached, the system will enter a repeated revisiting processes work cycle, i.e., $M_4 \rightarrow M_5 = \{W_4(1), W_1(4), W_2(3), R_3(W_3(3))\} \rightarrow M_6 = \{W_4(1), W_2(4), W_3(3), R_3(W_1(5))\} \rightarrow \cdots \rightarrow M_{3k+1} = \{W_4(1), W_3(6), W_1(7), R_3(W_2(7))\}$. In the final stage, the system undergoes two global work cycles and then reaches the first target steady state. That is, $M_{3k+1} \rightarrow M_{3k+2} = \{W_4(1), W_3(6), W_2(7), R_1(W_5(1))\} \rightarrow M_{3k+3} = \{W_5(1), W_4(2), W_2(7), R_1(W_6(1))\} \rightarrow M_{3k+4} = \{W_6(1), W_5(2), W_4(3), R_1(W_7(1))\}$.

To reach M_4 from M_0, the robot performs the following activities: ⟨unloading raw W_1 from the loadlock → moving to PM$_1$ → loading $W_1(1)$ into PM$_1$ → moving to the loadlock → unloading raw W_2 from the loadlock → moving to PM$_1$ → waiting (ω_{12}) for $W_1(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → loading $W_1(2)$ into PM$_2$ → moving to the loadlock → unloading raw W_3 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,3}$) for $W_2(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,3}$) for $W_1(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → loading $W_1(3)$ into PM$_3$ → moving to the loadlock → unloading raw W_4 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,4}$) for $W_3(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,4}$) for $W_2(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,4}$) for $W_1(3)$ at PM$_3$⟩.

To reach M_{3k+1} from M_4, the robot performs the following activities: ⟨swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,5}$) for $W_3(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,5}$) for $W_2(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,6}$) for $W_1(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,6}$) for $W_3(3)$ at PM$_3$ → swapping at PM$_3$ → $\cdots$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,3k-1}$) for $W_1(2k)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k}$) for $W_3(2k-1)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,3k+1}$) for $W_2(2k)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k+1}$) for $W_1(2k+1)$ at PM$_3$⟩.

To reach M_{3k+4} from M_{3k+1}, the robot performs the following activities: ⟨swapping at PM$_3$ → moving to the loadlock → loading completed W_1 into the loadlock → unloading raw W_5 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,3k+2}$) for $W_4(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,3k+3}$) for $W_3(2k)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ waiting ($\omega_{3,3k+3}$) for $W_2(2k+1)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_2 into the loadlock → unloading raw W_6

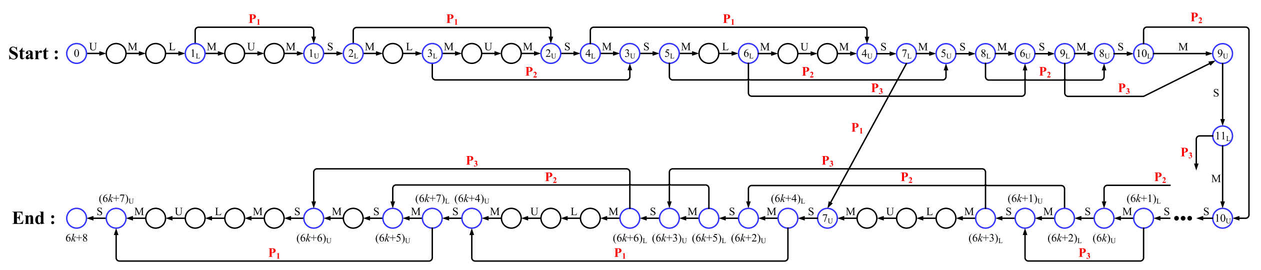


FIGURE 4. **PERT model for the SUTP with k-WRP and $k > 2$.**

from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,3k+3}$) for $W_5(1)$ at PM$_1$ → swapping at PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,3k+4}$) for $W_4(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k+4}$) for $W_3(2k+1)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_3 into the loadlock → unloading raw W_7 from the loadlock → moving to PM$_1$ → waiting ($\omega_{1,3k+4}$) for $W_6(1)$ at PM$_1$⟩.

According to the system state revolution process as well as corresponding operation sequences, we can build the PERT model (shown in Fig. 4) of the DACT with k-WRP from the idle state to the first target steady state. In this PERT model, Nodes 0, 1_L, 3_L, 6_L, and 6_U correspond to states M_0, M_1, M_2, M_3, and M_4 during the first stage, respectively. Since M_4, the system starts to execute the revisiting process (i.e., the local work cycle) multiple times. Accordingly, in the local work cycles, states $M_5, M_6, \cdots, M_{3k}$, and M_{3k+1} correspond to Nodes 9_U, $11_U, \cdots, (6k-1)_U$, and $(6k+1)_U$. In the final stage containing two global work cycles, states M_{3k+2}, M_{3k+3}, and M_{3k+4} correspond to Nodes 7_U, $(6k+4)_U$, and $(6k+7)_U$. For the DACT with k-WRP under $k > 2$, similar to *Algorithm* 1, we have the following algorithm.

IV. CLOSE-DOWN TRANSIENT PROCESS SCHEDULING

The CDTP starts when the last wafer of a batch is loaded into PM$_1$ and terminates after the last wafer is completed and loaded into the loadlock. For the DACT with 2-WRP, according to the paradigm of 1-WCS, the system reaches $\{W_n(1), W_{n-3}(4), W_{n-2}(3), R_1(W_{n-1}(2))\}$ when the last wafer W_n is loaded into PM$_1$. When W_n is completed and delivered to the loadlock, the system reaches $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$. Consequently, the CDTP of the DACT with 2-WRP starts from $\{W_n(1), W_{n-3}(4), W_{n-2}(3), R_1(W_{n-1}(2))\}$ and ends up with $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$. For the DACT with k-WRP, $k > 2$, based on the 3-WCS, when W_n is loaded into PM$_1$, the system reaches $\{W_n(1), W_{n-2}(2), W_{n-3}(3), R_1(W_{n-1}(2))\}$. Therefore, the CDTP of the DACT with k-WRP, $k > 2$, is determined from $\{W_n(1), W_{n-2}(2), W_{n-3}(3), R_1(W_{n-1}(2))\}$ to $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$. Note that there are four wafers in the DACT during the CDTP. Consequently, the CDTP of DACTs with WRP can be divided into four stages according to the time point when the completed wafer is loaded into the loadlock.

A. 2-WRP

In the first stage, i.e., after the beginning of CDTP while before W_{n-3} is completed and loaded into the loadlock, the system state evolves as: $M_1 = \{W_n(1), W_{n-3}(4), W_{n-2}(3), R_1(W_{n-1}(2))\} \rightarrow M_2 = \{W_n(1), W_{n-1}(2), W_{n-2}(3), R_3(W_{n-3}(5))\} \rightarrow M_3 = \{W_n(1), W_{n-2}(4), W_{n-3}(5), R_3(W_{n-1}(3))\} \rightarrow M_4 = \{W_n(1), W_{n-2}(4), W_{n-1}(3), R_0(\Theta)\}$. During such a state transition, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,2}$) for $W_{n-3}(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,2}$) for $W_{n-2}(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,3}$) for $W_{n-1}(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3}$) for $W_{n-3}(5)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_{n-3} into the loadlock⟩.

In the second stage, i.e., after M_4 while before W_{n-2} is loaded into the loadlock, the system state evolves as: $M_4 \rightarrow M_5 = \{\Theta, W_n(2), W_{n-1}(3), R_3(W_{n-2}(4))\} \rightarrow M_6 = \{\Theta, W_{n-1}(4), W_{n-2}(5), R_3(W_n(3))\} \rightarrow M_7 = \{\Theta, W_{n-1}(4), W_n(3), R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_1$ → waiting ($\omega_{1,5}$) for $W_n(1)$ at PM$_1$ → unloading W_n from PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,5}$) for $W_{n-2}(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,5}$) for $W_{n-1}(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,6}$) for W_n at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,6}$) for W_{n-2} at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading W_{n-2} into the loadlock⟩.

In the third stage, i.e., after M_7 while before W_{n-1} is loaded into the loadlock, the system state evolves as: $M_7 \rightarrow M_8 = \{\Theta, \Theta, W_n(3), R_3(W_{n-1}(5))\} \rightarrow M_9 = \{\Theta, W_n(4), \Theta, R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,8}$) for $W_{n-1}(4)$ at PM$_2$ → unloading W_{n-1} from PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,8}$) for $W_n(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → loading $W_n(4)$ into PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,9}$) for W_{n-1} at PM$_3$ → unloading W_{n-1} from PM$_3$ → moving to the loadlock → loading completed W_{n-1} into the loadlock⟩.

In the final stage, i.e., after M_9 while before W_n is loaded into the loadlock, the system state evolves as: $M_9 \rightarrow M_{10} = \{\Theta, \Theta, \Theta, R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,9}$)

Algorithm 1 ϕ_O Computation for 2-WRP

```
    I. Initialization
1:  $T_0$, the SUTP starts;
2:  $T_{1_L} = T_0 + \alpha + \mu + \beta$;
3:  $T_{1_U} = \max\{T_{1_L} + a_1, 2\mu + \alpha\}$;
4:  $T_{2_L} = T_{1_U} + \lambda$;
5:  $T_{3_L} = T_{2_L} + \mu + \beta$;
6:  $T_{2_U} = \max\{T_{2_L} + a_1, T_{3_L} + 2\mu + \alpha\}$;
7:  $T_{4_L} = T_{2_U} + \lambda$;
8:  $T_{3_U} = \max\{T_{3_L} + a_2, T_{4_L} + \mu\}$;
9:  $T_{5_L} = T_{3_U} + \lambda$;
10: $T_{6_L} = T_{5_L} + \mu + \beta$;
    II. Calculating $T_{9_{1U}}$
11: $T_{6_{1U}} = T_{6_L} + a_3$;
12: $T_{5_{1U}} = \max\{T_{5_L} + a_2, T_{6_{1U}} + \alpha + \mu\}$;
13: $T_{7_{1L}} = T_{5_{1U}} + \lambda$;
14: $T_{8_{1L}} = T_{7_{1L}} + \mu + \beta$;
15: $T_{4_{1U}} = \max\{T_{4_L} + a_1, T_{8_{1L}} + 2\mu + \alpha\}$;
16: $T_{9_{1L}} = T_{4_{1U}} + \lambda$;
17: $T_{7_{1U}} = \max\{T_{7_{1L}} + a_2, T_{9_{1L}} + \mu\}$;
18: $T_{10_{1L}} = T_{7_{1U}} + \lambda$;
19: $T_{8_{1U}} = \max\{T_{8_{1L}} + a_3, T_{10_{1L}} + \mu\}$;
20: $T_{11_{1L}} = T_{8_{1U}} + \lambda$;
21: $T_{10_{1U}} = \max\{T_{10_{1L}} + a_2, T_{11_{1L}} + \mu\}$;
22: $T_{11_{1U}} = \max\{T_{11_{1L}} + a_3, T_{10_{1U}} + \mu + \lambda\}$;
23: $T_{9_{1U}} = \max\{T_{9_{1L}} + a_1, T_{11_{1U}} + 2\mu + \alpha + \beta + \lambda\}$;
    III. Calculating $T_{9_{2U}}$
24: $T_{5_{2U}} = \max\{T_{5_L} + a_2, T_{6_L} + \mu\}$;
25: $T_{6_{2U}} = \max\{T_{6_L} + a_3, T_{5_{2U}} + \alpha + \mu\}$;
26: $T_{7_{2L}} = T_{6_{2U}} + \lambda$;
27: $T_{8_{2L}} = T_{7_{2L}} + \mu + \beta$;
28: $T_{4_{2U}} = \max\{T_{4_L} + a_1, T_{8_{2L}} + 2\mu + \alpha\}$;
29: $T_{9_{2L}} = T_{4_{2U}} + \lambda$;
30: $T_{8_{2U}} = \max\{T_{8_{2L}} + a_2, T_{9_{2L}} + \mu\}$;
31: $T_{10_{2L}} = T_{8_{2U}} + \lambda$;
32: $T_{7_{2U}} = \max\{T_{7_{2L}} + a_3, T_{10_{2L}} + \mu\}$;
33: $T_{11_{2L}} = T_{7_{2U}} + \lambda$;
34: $T_{10_{2U}} = \max\{T_{10_{2L}} + a_2, T_{11_{2L}} + \mu\}$;
35: $T_{11_{2U}} = \max\{T_{11_{2L}} + a_3, T_{10_{2U}} + \mu + \lambda\}$;
36: $T_{9_{2U}} = \max\{T_{9_{2L}} + a_1, T_{11_{2U}} + 2\mu + \alpha + \beta + \lambda\}$;
    IV. Final result
37: $T_{12} = \min\{T_{9_{1U}}, T_{9_{2U}}\} + \lambda$.
    V. Stop.
```

Algorithm 2 ϕ_O Computation for k-WRP With $k > 2$

```
1:  $T_0$, the SUTP starts;
2:  $T_{1_L} = T_0 + \alpha + \mu + \beta$;
3:  $T_{1_U} = \max\{T_{1_L} + a_1, 2\mu + \alpha\}$;
4:  $T_{2_L} = T_{1_U} + \lambda$;
5:  $T_{3_L} = T_{2_L} + \mu + \beta$;
6:  $T_{2_U} = \max\{T_{2_L} + a_1, T_{3_L} + 2\mu + \alpha\}$;
7:  $T_{4_L} = T_{2_U} + \lambda$;
8:  $T_{3_U} = \max\{T_{3_L} + a_2, T_{4_L} + \mu\}$;
9:  $T_{5_L} = T_{3_U} + \lambda$;
10: $T_{6_L} = T_{5_L} + \mu + \beta$;
11: $T_{4_U} = \max\{T_{4_L} + a_1, T_{6_L} + 2\mu + \alpha\}$;
12: $T_{7_L} = T_{4_U} + \lambda$;
13: $T_{5_U} = \max\{T_{5_L} + a_2, T_{7_L} + \mu\}$;
14: $T_{8_L} = T_{5_U} + \lambda$;
15: $T_{6_U} = \max\{T_{6_L} + a_3, T_{8_L} + \mu\}$;
16: $T_{9_L} = T_{6_U} + \lambda$;
17: $T_{8_U} = \max\{T_{8_L} + a_2, T_{9_L} + \mu\}$;
18: for $i \in \mathbb{N}_9^{6k+2}$ do
19:     $T_{i_L} = T_{(i-2)_U} + \lambda$;
20:     if $i = 2j, j \in \mathbb{N}_n$ then
21:         $T_{(i-1)_U} = \max\{T_{(i-1)_L} + a_3, T_{i_L} + \mu\}$;
22:     else if $i = 2j + 1, j \in \mathbb{N}_n$ then
23:         $T_{(i-1)_U} = \max\{T_{(i-1)_L} + a_2, T_{i_L} + \mu\}$;
24:     end if
25: end for
26: $T_{(6k+3)_L} = T_{(6k+1)_U} + \lambda$;
27: $T_{7_U} = \max\{T_{7_L} + a_1, T_{(6k+3)_L} + 2\mu + \alpha + \beta\}$;
28: $T_{(6k+4)_L} = T_{7_U} + \lambda$;
29: $T_{(6k+2)_U} = \max\{T_{(6k+2)_L} + a_2, T_{(6k+4)_L} + \mu\}$;
30: $T_{(6k+5)_L} = T_{(6k+2)_U} + \lambda$;
31: $T_{(6k+3)_U} = \max\{T_{(6k+3)_L} + a_3, T_{(6k+5)_L} + \mu\}$;
32: $T_{(6k+6)_L} = T_{(6k+3)_U} + \lambda$;
33: $T_{(6k+4)_U} = \max\{T_{(6k+4)_L} + a_1,$
                  $T_{(6k+6)_L} + 2\mu + \alpha + \beta\}$;
34: $T_{(6k+7)_L} = T_{(6k+4)_U} + \lambda$;
35: $T_{(6k+5)_U} = \max\{T_{(6k+5)_L} + a_2, T_{(6k+7)_L} + \mu\}$;
36: $T_{(6k+6)_U} = \max\{T_{(6k+6)_L} + a_3, T_{(6k+5)_U} + \mu + \lambda\}$;
37: $T_{(6k+7)_U} = \max\{T_{(6k+7)_L} + a_1,$
                  $T_{(6k+6)_U} + 2\mu + \alpha + \beta + \lambda\}$;
38: $T_{6k+8} = T_{(6k+7)_U} + \lambda$;
39: Stop.
```

for $W_n(4)$ at PM$_2$ → unloading $W_n(4)$ from PM$_2$ → moving to PM$_3$ → loading $W_n(4)$ into PM$_3$ → waiting ($\omega_{3,10}$) for $W_n(5)$ at PM$_3$ → unloading $W_n(5)$ from PM$_3$ → moving to the loadlock → loading completed W_n into the loadlock⟩.

Up to now, we have provided the operation sequence across the entire CDTP. This allows us to easily build the PERT model for the DACT with 2-WRP during the CDTP, which is shown as Fig. 5. During the first stage, states M_1 and M_3 respectively correspond to Nodes 1_L and 3_U, state M_2 corresponds to the succeeding node of 2_L, and state M_4 corresponds to 1_U's preceding node. In the second stage, states M_5 and M_6 respectively correspond to Nodes 1_L and 3_U, and state M_7 corresponds to 8_U's preceding node. In the third stage, state M_8 and state M_9 correspond to Node 9_U and Node 11_U's preceding node, respectively. Finally, state M_{10} corresponds to the terminal Node 13. We use Γ_k to indicate the time taken from the initial state of CDTP (represented by Node 1), i.e., $\{W_n(1), W_{n-3}(4), W_{n-2}(3), R_1(W_{n-1}(2))\}$ for 2-WRP and $\{W_n(1), W_{n-2}(2), W_{n-3}(3), R_1(W_{n-1}(2))\}$ for k-WRP with $k > 2$ $\{\Theta, \Theta, \Theta, R_0(\Theta)\}$, to the terminal state (Node k). Let Φ_O indicate the time taken to complete the CDTP of the DACT with WRP. Similarly, the same time

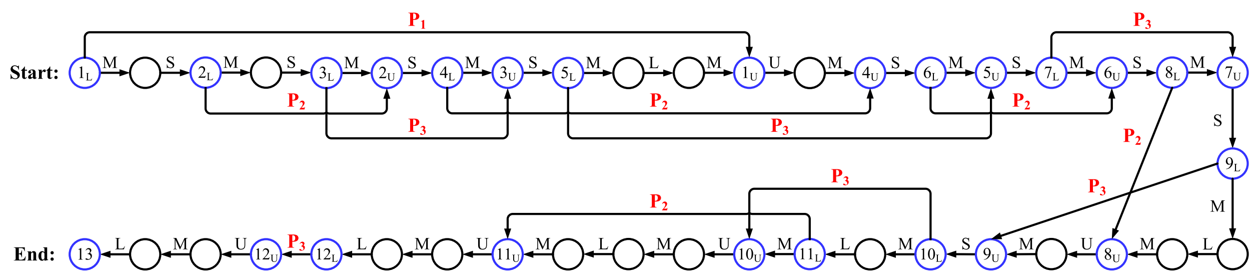


FIGURE 5. PERT model for the CDTP with 2-WRP.

Algorithm 3 Φ_O Computation for 2-WRP

```
1: $\Gamma_{1_L}$, the system starts to switch from the steady state to the CDTP;
2: $\Gamma_{2_L} = \Gamma_{1_L} + \mu + \lambda$;
3: $\Gamma_{3_L} = \Gamma_{2_L} + \mu + \lambda$;
4: $\Gamma_{2_U} = \max\{\Gamma_{2_L} + a_2, \Gamma_{3_L} + \mu\}$;
5: $\Gamma_{4_L} = \Gamma_{2_U} + \lambda$;
6: $\Gamma_{3_U} = \max\{\Gamma_{3_L} + a_3, \Gamma_{4_L} + \mu\}$;
7: $\Gamma_{5_L} = \Gamma_{3_U} + \lambda$;
8: $\Gamma_{1_U} = \max\{\Gamma_{1_L} + a_1, \Gamma_{5_L} + 2\mu + \beta\}$;
9: $\Gamma_{4_U} = \max\{\Gamma_{4_L} + a_2, \Gamma_{1_U} + \alpha + \mu\}$;
10: $\Gamma_{6_L} = \Gamma_{4_U} + \lambda$;
11: $\Gamma_{5_U} = \max\{\Gamma_{5_L} + a_3, \Gamma_{6_L} + \mu\}$;
12: $\Gamma_{7_L} = \Gamma_{5_U} + \lambda$;
13: $\Gamma_{6_U} = \max\{\Gamma_{6_L} + a_2, \Gamma_{7_L} + \mu\}$;
14: $\Gamma_{8_L} = \Gamma_{6_U} + \lambda$;
15: $\Gamma_{7_U} = \max\{\Gamma_{7_L} + a_3, \Gamma_{8_L} + \mu\}$;
16: $\Gamma_{9_L} = \Gamma_{7_U} + \lambda$;
17: $\Gamma_{8_U} = \max\{\Gamma_{8_L} + a_2, \Gamma_{9_L} + 2\mu + \beta\}$;
18: $\Gamma_{9_U} = \max\{\Gamma_{9_L} + a_3, \Gamma_{8_U} + \alpha + \mu\}$;
19: $\Gamma_{10_L} = \Gamma_{9_U} + \lambda$;
20: $\Gamma_{11_L} = \Gamma_{10_L} + \mu + \beta$;
21: $\Gamma_{10_U} = \max\{\Gamma_{10_L} + a_3, \Gamma_{11_L} + \mu\}$;
22: $\Gamma_{11_U} = \max\{\Gamma_{11_L} + a_2, \Gamma_{10_U} + 2\mu + \alpha + \beta\}$;
23: $\Gamma_{12_L} = \Gamma_{11_U} + \mu + \alpha + \beta$;
24: $\Gamma_{12_U} = \Gamma_{12_L} + a_3$;
25: $\Gamma_{13} = \Gamma_{12_U} + \mu + \alpha + \beta$;
26: **Stop**.
```

based on the VWM is denoted as Φ_V. Therefore, we can calculate Φ_O for the DACT with 2-WRP according to the following algorithm.

B. k-WRP WITH $k > 2$

In the first stage, i.e., after the beginning of CDTP while before W_{n-3} is completed and loaded into the loadlock, the system state evolves as: $M_1 = \{W_n(1), W_{n-2}(2), W_{n-3}(3), R_1(W_{n-1}(2))\} \rightarrow M_2 = \{W_n(1), W_{n-1}(2), W_{n-3}(3), R_3(W_{n-2}(3))\} \rightarrow M_3 = \{W_n(1), W_{n-3}(4), W_{n-2}(3), R_3(W_{n-1}(3))\} \rightarrow M_4 = \{W_n(1), W_{n-2}(4), W_{n-1}(3), R_3(W_{n-3}(5))\} \rightarrow \cdots \rightarrow M_{3k-2} = \{W_n(1), W_{n-2}(2k), W_{n-1}(2k-1), R_3(W_{n-3}(2k+1))\} \rightarrow M_{3k-1} = \{W_n(1), W_{n-1}(2k), W_{n-3}(2k+1), R_3(W_{n-2}(2k+1))\} \rightarrow M_{3k} = \{W_n(1), W_{n-1}(2k), W_{n-2}(2k+1), R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,2}$) for $W_{n-2}(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,2}$) for $W_{n-3}(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,3}$) for $W_{n-1}(2)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3}$) for $W_{n-2}(3)$ at PM$_3$ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,4}$) for $W_{n-3}(4)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,4}$) for $W_{n-1}(3)$ at PM$_3$ → ⋯ → swapping at PM$_3$ → moving to PM$_2$ → waiting ($\omega_{2,3k-2}$) for $W_{n-2}(2k)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k-1}$) for W_{n-3} at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_{n-3} into the loadlock⟩.

In the second stage, i.e., after M_{3k} while before W_{n-2} is loaded into the loadlock, the system state evolves as: $M_{3k} \rightarrow M_{3k+1} = \{\Theta, W_n(2), W_{n-1}(2k+1), R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_1$ → waiting ($\omega_{1,3k+1}$) for $W_n(1)$ at PM$_1$ → unloading W_n from PM$_1$ → moving to PM$_2$ → waiting ($\omega_{2,3k+1}$) for $W_{n-1}(2k)$ at PM$_2$ → swapping at PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k+1}$) for $W_{n-2}(2k+1)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_{n-2} into the loadlock⟩.

In the third stage, i.e., after M_{3k+1} while before W_{n-1} is loaded into the loadlock, the system state evolves as: $M_{3k+1} \rightarrow M_{3k+2} = \{\Theta, \Theta, W_n(3), R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_2$ → waiting ($\omega_{2,3k+2}$) for $W_n(2)$ at PM$_2$ → unloading $W_n(2)$ from PM$_2$ → moving to PM$_3$ → waiting ($\omega_{3,3k+2}$) for $W_{n-1}(2k+1)$ at PM$_3$ → swapping at PM$_3$ → moving to the loadlock → loading completed W_{n-1} into the loadlock⟩.

In the final stage, i.e., after M_{3k+2} while before W_n is loaded into the loadlock, the system state evolves as: $M_{3k+2} \rightarrow M_{3k+3} = \{\Theta, \Theta, W_n(3), R_3(W_n(4))\} \rightarrow M_{3k+4} = \{\Theta, \Theta, W_n(5), R_3(W_n(6))\} \rightarrow \cdots \rightarrow M_{4k+1} = \{\Theta, \Theta, W_n(2k-1), R_3(W_n(2k))\} \rightarrow M_{4k+2} = \{\Theta, \Theta, W_n(2k+1), R_3(W_n(0))\} \rightarrow M_{4k+3} = \{\Theta, \Theta, \Theta, R_0(\Theta)\}$. During this process, the robot performs the following activities: ⟨moving to PM$_3$ → waiting ($\omega_{3,3k+3}$) for $W_n(3)$ at PM$_3$ → unloading $W_n(3)$ from PM$_3$ → moving to PM$_2$ → loading $W_n(4)$ into PM$_2$ → waiting ($\omega_{2,3k+4}$) for $W_n(4)$ at PM$_2$ → unloading $W_n(4)$ from PM$_2$ → moving to PM$_3$ → loading $W_n(5)$ into PM$_3$ → waiting ($\omega_{3,3k+4}$) for $W_n(5)$ at PM$_3$ → ⋯ → unloading $W_n(2k-1)$ from PM$_3$ → moving to PM$_2$ → loading $W_n(2k)$ into PM$_2$ → waiting ($\omega_{2,4k+2}$) for $W_n(2k)$ at PM$_2$ → unloading $W_n(2k)$

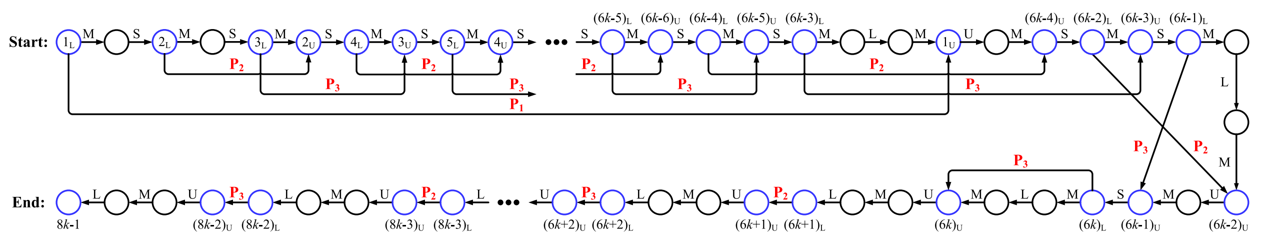


FIGURE 6. PERT model for the CDTP with k-WRP and $k > 2$.

from PM$_2$ → moving to PM$_3$ → loading $W_n(2k+1)$ into PM$_3$ → waiting ($\omega_{3,4k+2}$) for $W_n(2k+1)$ at PM$_3$ → unloading $W_n(2k+1)$ from PM$_3$ → moving to the loadlock → loading completed W_n into the loadlock⟩.

Based on the above operation sequence, we can build the PERT model as shown in Fig. 6. In the first stage, state M_1, M_3, M_4, $\cdots$, M_{3k-2}, and M_{3k-1} respectively correspond to Nodes 1_L, 3_U, 5_U, $\cdots$, $(6k-7)_U$, $(6k-5)_L$ M_2, while M_2 and M_{3k} correspond to 2_L's succeeding node and 1_U's preceding node, respectively. During the rest stages, sates M_{3k+1}, M_{3k+2}, M_{3k+3}, M_{3k+4}, $\cdots$, M_{4k+1}, M_{4k+2}, and M_{4k+3} correspond to $(6k-2)_U$'s preceding node, $(6k)_U$'s preceding node, Nodes $(6k)_U$, $(6k+2)_U$, $\cdots$, $(8k-4)_U$, $(8k-2)_U$, and $8k-1$, respectively. To calculate Φ_O of the DACT with k-WRP under $k > 2$, we have the following algorithm.

Algorithm 4 Φ_O Computation for k-WRP With $k > 2$

```
1: $\Gamma_{1_L}$, the system starts to switch from the steady state to the CDTP;
2: $\Gamma_{2_L} = \Gamma_{1_L} + \mu + \lambda$;
3: $\Gamma_{3_L} = \Gamma_{2_L} + \mu + \lambda$;
4: $\Gamma_{2_U} = \max\{\Gamma_{2_L} + a_2, \Gamma_{3_L} + \mu\}$;
5: for $i \in \mathbb{N}_3^{6k-4}$ do
6:     $\Gamma_{i_L} = \Gamma_{(i-2)_U} + \lambda$;
7:     if $i = 2j, j \in \mathbb{N}_n$ then
8:         $\Gamma_{(i-1)_U} = \max\{\Gamma_{(i-1)_L} + a_3, \Gamma_{i_L} + \mu\}$;
9:     else if $i = 2j+1, j \in \mathbb{N}_n$ then
10:        $\Gamma_{(i-1)_U} = \max\{\Gamma_{(i-1)_L} + a_2, \Gamma_{i_L} + \mu\}$;
11:    end if
12: end for
13: $\Gamma_{(6k-3)_L} = \Gamma_{(6k-5)_U} + \lambda$;
14: $\Gamma_{1_U} = \max\{\Gamma_{1_L} + a_1, \Gamma_{(6k-3)_L} + 2\mu + \beta\}$;
15: $\Gamma_{(6k-4)_U} = \max\{\Gamma_{(6k-4)_L} + a_2, \Gamma_{1_U} + \alpha + \mu\}$;
16: $\Gamma_{(6k-2)_L} = \Gamma_{(6k-4)_U} + \lambda$;
17: $\Gamma_{(6k-3)_U} = \max\{\Gamma_{(6k-3)_L} + a_3, \Gamma_{(6k-2)_L} + \mu\}$;
18: $\Gamma_{(6k-1)_L} = \Gamma_{(6k-3)_U} + \lambda$;
19: $\Gamma_{(6k-2)_U} = \max\{\Gamma_{(6k-2)_L} + a_2, \Gamma_{(6k-1)_L} + 2\mu + \beta\}$;
20: $\Gamma_{(6k-1)_U} = \max\{\Gamma_{(6k-1)_L} + a_3, \Gamma_{(6k-2)_U} + \alpha + \mu\}$;
21: $\Gamma_{(6k)_L} = \Gamma_{(6k-1)_U} + \lambda$;
22: $\Gamma_{(6k)_U} = \max\{\Gamma_{(6k)_L} + a_3, \Gamma_{(6k)_L} + 2\mu + \beta\}$;
23: for $i \in \mathbb{N}_{6k}^{8k-2}$ do
24:     $\Gamma_{i_L} = \Gamma_{(i-1)_U} + \mu + \alpha + \beta$;
25:     if $i = 2j, j \in \mathbb{N}_n$ then
26:         $\Gamma_{i_U} = \Gamma_{i_L} + a_3$;
27:     else if $i = 2j+1, j \in \mathbb{N}_n$ then
28:         $\Gamma_{i_U} = \Gamma_{i_L} + a_2$;
29:     end if
30: end for
31: $\Gamma_{8k-1} = \Gamma_{(8k-2)_U} + \mu + \alpha + \beta$;
32: Stop.
```

V. ILLUSTRATIVE EXAMPLES

In this section, several examples are provided to demonstrate the preceding section' results. In the following examples, the time unit is second, abbreviated as s.

Example 1: The wafer processing time at process steps 1-3 are 76 s, 43 s, and 32 s, i.e., $a_1 = 76$ s, $a_2 = 43$ s, and $a_3 = 32$ s. The robot task times of unloading, loading, moving, and swap operation are $\alpha = \beta = \mu = 4$ s, and $\lambda = 10$ s, respectively.

In this case, the bottleneck of the revisiting process is the second process step. For 2-WRP, by combining the VWM-based 1-WCS in [32], it will take 446 s and 410 s to reach the first target steady state and complete the CDTP. However, it takes 402 s and 341 s to do so via *Algorithms* 1 and 3, respectively. For k-WRP with $k > 2$, with the VWM-based 3-WCS in [31], it will take 1106 s, 1424 s, and 1742 s to reach the first target steady state and 898 s, 1216 s, and 1534 s to complete the CDTP, when k is 3, 4, and 5, respectively. By using *Algorithms* 2 and 4, it takes 814 s, 963 s, 1122 s, and 688 s, 946 s, 1204 s, respectively. Algorithms presented in this paper obtain a great reduction in transient processes schedule.

Example 2: The wafer processing time at process steps 1-3 are 110 s, 45 s, and 80 s, respectively; and $\alpha = \beta = 4$ s, $\mu = 2$ s, and $\lambda = 8$ s.

In this case, different from *Example* 1, $a_3 > a_2$, i.e., the third process step is the bottleneck of the revisiting process. For 2-WRP, it takes 515 s and 468 s to reach the first target steady state and finish the CDTP, respectively, according to *Algorithms* 1 and 3, whereas it takes 614 s and 551 s respectively by using the VWM-based 1-WCS. For k-WRP when k are respectively 3, 4, and 5, it takes 1536 s, 1986 s, and 2436 s to reach the first target steady state by the VWM-based 3-WCS, whereas it merely needs 1098 s, 1280 s, and 1551 s respectively by *Algorithm* 2. As for the CDTP, according to *Algorithm* 4, it will take 961 s, 1318 s, and 1675 s, respectively, while it takes 1312 s, 1762 s, and 2212 s by the

TABLE 1. Comparison of the experimental results.

No.	Manufacturing Parameters							Start-Up Process			Close-Down Process		
	α/β	μ	λ	a_1	a_2	a_3	k	ϕ_V	ϕ_O	Reduction (%)	Φ_V	Φ_O	Reduction (%)
1	4	4	10	76	43	32	2	446	402	9.87	410	341	16.83
							3	1106	814	26.4	898	688	23.39
							4	1424	963	32.37	1216	946	22.2
							5	1742	1122	35.59	1534	1204	21.51
2	4	2	8	110	45	67	2	614	515	16.12	551	486	11.8
							3	1536	1098	28.52	1312	961	26.75
							4	1986	1280	35.55	1762	1318	25.2
							5	2436	1551	36.33	2212	1675	24.28
3	3	3	8	320	180	210	2	1758	1356	22.87	1554	1344	13.51
							3	4378	2999	31.5	3518	2801	20.38
							4	5686	3653	35.75	4826	3863	19.95
							5	6994	4299	38.53	6134	4925	19.71
4	4	3	10	450	260	190	2	2177	1851	14.97	1923	1579	17.89
							3	5637	3981	29.38	4460	3325	25.45
							4	7257	4781	34.12	6080	4607	24.23
							5	8877	5601	36.9	7700	5889	23.52

VWM-based 3-WCS, respectively. Algorithms proposed in this paper outperform appreciably beyond the VWM-based scheduling approach for the transient processes with WRP.

We also provide two cases (No. 3 and 4 in TABLE 1) with significant differences between the robot task time and processing times. Details of these cases provided in this section can refer to TABLE 1. Compared with the VWM-based scheduling approach in [31], [32], the PERT-based method achieves significant time reduction. Especially as the increase of k, the time taken for the SUTP decreases more significantly than the VWM-based schedule. As a whole, in terms of time, the PERT-based scheduling method for the transient process is superior to the VWM-based cyclic scheduling approach, even though the latter one is easier to implement.

VI. CONCLUSION

As the growing tendency of high-mix and low-volume production, the wafer lot size decreases steadily, leading to an increasing number of the transient process. In particular, wafer revisiting makes this scheduling problem more complicated. This paper is concerned about the transient process scheduling problems for DACTs with WRP. For the sake of the simplicity of cyclic scheduling in implementation, existing research tends to adopt a virtual wafer scheduling method to operate transient processes for DACTs with WRP, resulting in extensive redundant activities. To resolve such a problem, we adopt a PERT-based model to analyze transient processes of DACTs with WRP comprehensively. Moreover, we present computationally efficient algorithms to optimize transient processes. The numerical experimental results indicate that the proposed approach performs much better than the virtual wafer scheduling method. Future studies are expected to extend the proposed method to transient process scheduling for cluster tools with WRP and WRTCs. It is also meaningful to investigate parallel PMs and other complex cases caused by disruptive events reported in [17]–[21].

REFERENCES

[1] C. Pan, M. Zhou, Y. Qiao, and N. Wu, "Scheduling cluster tools in semiconductor manufacturing: Recent advances and challenges," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 586–601, Apr. 2018.

[2] J.-H. Kim, T.-E. Lee, H.-Y. Lee, and D.-B. Park, "Scheduling analysis of time-constrained dual-armed cluster tools," *IEEE Trans. Semicond. Manuf.*, vol. 16, no. 3, pp. 521–534, Aug. 2003.

[3] T.-E. Lee and S.-H. Park, "An extended event graph with negative places and tokens for time window constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 4, pp. 319–332, Oct. 2005.

[4] N. Wu, C. 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.

[5] N. Wu and M. Zhou, "A closed-form solution for schedulability and optimal scheduling of dual-arm cluster tools with wafer residency time constraint based on steady schedule analysis," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 2, pp. 303–315, Apr. 2010.

[6] Y. Lim, T.-S. Yu, and T.-E. Lee, "A new class of sequences without interferences for cluster tools with tight wafer delay constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 1, pp. 392–405, Jan. 2019.

[7] F. Yang, N. Wu, Y. Qiao, M. Zhou, R. Su, and T. Qu, "Modeling and optimal cyclic scheduling of time-constrained single-robot-arm cluster tools via Petri nets and linear programming," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 3, pp. 871–883, Mar. 2020.

[8] Q. Zhu, M. Zhou, Y. Qiao, N. Wu, and Y. Hou, "Multiobjective scheduling of dual-blade robotic cells in wafer fabrication," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 12, pp. 5015–5023, Dec. 2020.

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

[10] W. Xiong, C. Pan, Y. Qiao, N. Wu, M. Chen, and P. Hsieh, "Reducing wafer delay time by robot idle time regulation for single-arm cluster tools," *IEEE Trans. Autom. Sci. Eng.*, early access, Aug. 17, 2020, doi: 10.1109/TASE.2020.3014078. [Online]. Available: https://ieeexplore.ieee.org/document/9169798

[11] N. Du, H. Hu, and M. Zhou, "A survey on robust deadlock control policies for automated manufacturing systems with unreliable resources," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 389–406, Jan. 2020.

[12] N. Du, H. Hu, and M. Zhou, "Robust deadlock avoidance and control of automated manufacturing systems with assembly operations using Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 4, pp. 1961–1975, Oct. 2020.

[13] N. Du and H. Hu, "Robust deadlock detection and control of automated manufacturing systems with multiple unreliable resources using Petri nets," *IEEE Trans. Autom. Sci. Eng.*, early access, Sep. 10, 2020, doi: 10.1109/TASE.2020.3019684. [Online]. Available: https://ieeexplore.ieee.org/document/9194275

[14] X. Wang and H. Hu, "A robust control approach to automated manufacturing systems allowing multitype and multiquantity of resources with Petri nets," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 10, pp. 3499–3514, Oct. 2020.

[15] X. Wang and H. Hu, "A robust control approach to automated manufacturing systems combining absorbing and distributing characteristics," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 12, pp. 5024–5036, Dec. 2020.

[16] J.-H. Kim and T.-E. Lee, "Schedulability analysis of time-constrained cluster tools with bounded time variation by an extended Petri net," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 3, pp. 490–503, Jul. 2008.

[17] N. Wu and M. Zhou, "Analysis of wafer sojourn time in dual-arm cluster tools with residency time constraint and activity time variation," *IEEE Trans. Semicond. Manuf.*, vol. 23, no. 1, pp. 53–64, Feb. 2010.

[18] N. Wu and M. Zhou, "Schedulability analysis and optimal scheduling of dual-arm cluster tools with residency time constraint and activity time variation," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 1, pp. 203–209, Jan. 2012.

[19] N. Q. Wu and M. C. Zhou, "Modeling, analysis and control of dual-arm cluster tools with residency time constraint and activity time variation based on Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 3, pp. 564–577, Jul. 2012.

[20] Y. Qiao, N. Q. Wu, and M. C. Zhou, "Real-time scheduling of single-arm cluster tools subject to residency time constraints and bounded activity time variation," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 1, pp. 564–577, Jan. 2012.

[21] Y. Qiao, N. Wu, and M. Zhou, "Petri net modeling and wafer sojourn time analysis of single-arm cluster tools with residency time constraints and activity time variation," *IEEE Trans. Semicond. Manuf.*, vol. 25, no. 3, pp. 432–446, Aug. 2012.

[22] C. Pan, Y. Qiao, N. Wu, and M. 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.

[23] F. Yang, X. Tang, N. Wu, C. 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.

[24] Y. Lim, T.-S. Yu, and T.-E. Lee, "Adaptive scheduling of cluster tools with wafer delay constraints and process time variation," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 375–388, Jan. 2020.

[25] W. M. Zuberek, "Cluster tools with chamber revisiting—Modeling and analysis using timed Petri nets," *IEEE Trans. Semicond. Manuf.*, vol. 17, no. 3, pp. 333–344, Aug. 2004.

[26] H.-Y. Lee and T.-E. Lee, "Scheduling single-armed cluster tools with reentrant wafer flows," *IEEE Trans. Semicond. Manuf.*, vol. 19, no. 2, pp. 226–240, May 2006.

[27] N. Wu, F. Chu, C. Chu, and M. Zhou, "Petri net-based scheduling of single-arm cluster tools with reentrant atomic layer deposition processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 1, pp. 42–55, Jan. 2011.

[28] F. Yang, N. Wu, Y. Qiao, M. Zhou, and Z. 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.

[29] N. Wu, F. Chu, C. Chu, and M. Zhou, "Petri net modeling and cycle-time analysis of dual-arm cluster tools with wafer revisiting," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 1, pp. 196–207, Jan. 2013.

[30] N. Wu, M. Zhou, F. Chu, and C. Chu, "A Petri-net-based scheduling strategy for dual-arm cluster tools with wafer revisiting," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 5, pp. 1182–1194, Sep. 2013.

[31] Y. Qiao, N. Wu, Q. Zhu, and L. Bai, "Cycle time analysis of dual-arm cluster tools for wafer fabrication processes with multiple wafer revisiting times," *Comput. Oper. Res.*, vol. 53, pp. 252–260, Jan. 2015.

[32] Y. Qiao, N. Wu, and M. Zhou, "A Petri net-based novel scheduling approach and its cycle time analysis for dual-arm cluster tools with wafer revisiting," *IEEE Trans. Semicond. Manuf.*, vol. 26, no. 1, pp. 100–110, Feb. 2013.

[33] Y. Qiao, N. Wu, and M. C. Zhou, "Scheduling of dual-arm cluster tools with wafer revisiting and residency time constraints," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 286–300, Feb. 2014.

[34] Y. Qiao, N. Wu, and M. Zhou, "Schedulability and scheduling analysis of dual-arm cluster tools with wafer revisiting and residency time constraints based on a novel schedule," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 3, pp. 472–484, Mar. 2015.

[35] Y. Qiao, N. Wu, F. Yang, M. Zhou, and Q. Zhu, "Wafer sojourn time fluctuation analysis of time-constrained dual-arm cluster tools with wafer revisiting and activity time variation," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 4, pp. 622–636, Apr. 2018.

[36] Y. Qiao, N. Wu, F. Yang, M. Zhou, Q. Zhu, and T. Qu, "Robust scheduling of time-constrained dual-arm cluster tools with wafer revisiting and activity time disturbance," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 6, pp. 1228–1240, Jun. 2019.

[37] H.-J. Kim, J.-H. Lee, C. Jung, and T.-E. Lee, "Scheduling cluster tools with ready time constraints for consecutive small lots," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 1, pp. 145–159, Jan. 2013.

[38] J.-H. Lee, H.-J. Kim, and T.-E. Lee, "Scheduling lot switching operations for cluster tools," *IEEE Trans. Semicond. Manuf.*, vol. 26, no. 4, pp. 592–601, Nov. 2013.

[39] J.-H. Lee and H.-J. Kim, "Makespan analysis of lot switching period in cluster tools," *IEEE Trans. Semicond. Manuf.*, vol. 29, no. 2, pp. 127–136, May 2016.

[40] J.-H. Lee, H.-J. Kim, and T.-E. Lee, "Scheduling cluster tools for concurrent processing of two wafer types," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 525–536, Apr. 2014.

[41] J.-H. Lee, H.-J. Kim, and T.-E. Lee, "Scheduling cluster tools for concurrent processing of two wafer types with PM sharing," *Int. J. Prod. Res.*, vol. 53, no. 19, pp. 6007–6022, Oct. 2015.

[42] S.-G. Ko, T.-S. Yu, and T.-E. Lee, "Scheduling dual-armed cluster tools for concurrent processing of multiple wafer types with identical job flows," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1058–1070, Jul. 2019.

[43] J. Wang, C. Pan, H. Hu, L. Li, and Y. Zhou, "A cyclic scheduling approach to single-arm cluster tools with multiple wafer types and residency time constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1373–1386, Jul. 2019.

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

[45] T.-K. Kim, C. Jung, 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.

[46] 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.*, early access, Mar. 20, 2020, doi: 10.1109/TSMC.2020.2978486. [Online]. Available: https://ieeexplore.ieee.org/document/9043734

[47] Y. Qiao, M. Zhou, N. Wu, and Q. Zhu, "Scheduling and control of startup process for single-arm cluster tools with residency time constraints," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1243–1256, Jul. 2017.

[48] Q. Zhu, M. Zhou, Y. Qiao, and N. 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.

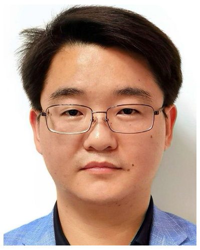
[49] D.-K. Kim, T.-E. Lee, and H.-J. Kim, "Optimal scheduling of transient cycles for single-armed cluster tools with parallel chambers," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 1165–1175, Apr. 2016.

[50] F. Yang, Y. Qiao, K. Gao, N. Wu, Y. Zhu, I. W. Simon, and R. Su, "Efficient approach to scheduling of transient processes for time-constrained single-arm cluster tools with parallel chambers," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 10, pp. 3646–3657, Oct. 2020.

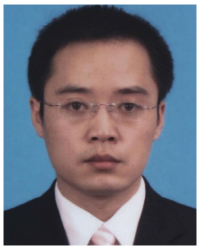
[51] Y. Qiao, M. Zhou, N. Wu, Z. Li, and Q. Zhu, "Closing-down optimization for single-arm cluster tools subject to wafer residency time constraints," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Jan. 22, 2020, doi: 10.1109/TSMC.2020.2964032. [Online]. Available: https://ieeexplore.ieee.org/document/8966604

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

[53] C. Rong Pan, Y. Qiao, M. Chu Zhou, and N. Qi Wu, "Scheduling and analysis of start-up transient processes for dual-arm cluster tools with wafer revisiting," *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 2, pp. 160–170, May 2015.



JIPENG WANG (Student Member, IEEE) received the B.S. degree from the Hubei University of Technology, Wuhan, China, in 2011, and the M.S. degree from the Jiangxi University of Science and Technology, Ganzhou, China, in 2016, both in mechanical engineering. He is currently pursuing the Ph.D. degree in control theory and control engineering with the School of Mechano-Electronic Engineering, Xidian University, Xi'an, China. His current research interests include manufacturing system modeling and scheduling, Petri nets, and discrete-event systems.



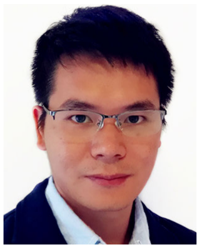
HESUAN HU (Senior Member, IEEE) received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electro-mechanical engineering from Xidian University, Xi'an, China, in 2003, 2005, and 2010, respectively.

He is currently a Full Professor with Xidian University, and also a Researcher with Xi'an Jiaotong University, China. He has more than 140 publications in journals, book chapters, and conference proceedings in the below areas and holds more than 40 issued and filed patents in his fields of expertise. His current research interests include discrete event systems and their supervisory control techniques, Petri nets, automated manufacturing systems, multimedia streaming systems, autonomous vehicles, cyber security, and artificial intelligence.

Dr. Hu was a recipient of many national and international awards, including the Franklin V. Taylor Outstanding Paper Award from the IEEE SMC Society, in 2010, and the Finalist of the Best Automation Paper from the IEEE ICRA Society, in 2013, 2016, and 2017. He has been an Associate Editor of the *IEEE Control Systems Magazine*, *IEEE Robotics and Automation Magazine*, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, IEEE ROBOTICS AND AUTOMATION LETTERS, *Journal of Intelligent Manufacturing*, and so on.

CHUNRONG PAN (Senior Member, IEEE) received the M.S. degree in mechatronics engineering from Shantou University, Shantou, China, in 2006, and the Ph.D. degree in mechanical engineering from the Guangdong University of Technology, Guangzhou, China, in 2010.

From 1997 to 2011, he was with Shantou University. In 2011, he joined the Jiangxi University of Science and Technology, Ganzhou, China, where he is currently a Full Professor with the Department of Mechatronics Engineering. He was a Visiting Scholar with the New Jersey Institute of Technology, Newark, NJ, USA, from 2013 to 2014, and Bournemouth University, Poole, U.K., from 2018 to 2019. He has published over ten research papers in international journals and conferences. His current research interests include modeling, simulation, scheduling, and control of integrated manufacturing equipment.



LIANG LI received the M.S. degree in mechanical engineering from the Jiangxi University of Science and Technology, Ganzhou, China, in 2015, the Ph.D. degree in control theory and control engineering from Xidian University, Xi'an, China, in 2020, and the Ph.D. degree in computer science and information engineering from the University of Salerno, Fisciano, Italy, in 2020. He is currently a Lecturer with the Wuhan University of Science and Technology. His current research interests include Petri nets, and control and optimization of timed discrete-event systems.