electronics

MDPI

Article

Dual-Arm Cluster Tool Scheduling for Reentrant Wafer Flows

Tairan Song [1], Yan Qiao [1], Yunfang He [1], Naiqi Wu [1,2,*], Zhiwu Li [1] and Bin Liu [3]

[1] Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, Macao 999078, China; song.tairan@ikasinfo.com (T.S.); yqiao@must.edu.mo (Y.Q.); heyunfang.sch@gmail.com (Y.H.); zwli@must.edu.mo (Z.L.)
[2] State Key Laboratory of Precision Electronic Manufacturing Technology and Equipment, Guangdong University of Technology, Guangzhou 510006, China
[3] IKAS Industries Technology (Suzhou) Company Ltd., Suzhou 215000, China; liu.bin@ikasinfo.com
* Correspondence: nqwu@must.edu.mo

Abstract: Cluster tools are the key equipment in semiconductor manufacturing systems. They have been widely adopted for many wafer fabrication processes, such as chemical and physical vapor deposition processes. Reentrant wafer flows are commonly seen in cluster tool operations for deposition processes. It is very complicated to schedule cluster tools with reentrant processes. For a dual-arm cluster tool with two-time reentering, the existing studies point out that a *one-wafer periodical* (1-WP) schedule can be found, and it is optimal in terms of productivity. However, for some wafer fabrication processes, wafers should be processed at some PMs more than two times. This gives rise to a question of whether there still exists a 1-WP schedule for dual-arm cluster tools with the number of reentering times being more than two such that the cycle time of a tool can reach the lower bound. This problem is still open, and this is what this work wants to tackle. For a dual-arm cluster tool with the number of reentering times being k (>2) times, if there does not exist a value $f \in \{1, 2 \dots\}$ such that $k = 3f$, theoretical proofs are given to show that a 1-WP schedule can be found, otherwise it does not exist. For cases with a 1-WP schedule, the cycle time can be obtained by analytical expressions. For the cases without a 1-WP schedule, two new methods for a three-wafer periodical schedule are proposed to improve the system productivity by comparing it with an existing three-wafer periodical schedule. The applications of the obtained results are demonstrated by examples. Wafer residency time constraints are required for some wafer fabrication processes. Note that the results obtained in this work cannot be directly applied to cluster tools with both reentrant wafer flows and wafer residency time constraints. Nevertheless, schedulablity and scheduling analyses for that applications can be conducted based on the obtained results in this work.

Keywords: cluster tools; reentrant flows; scheduling; semiconductor manufacturing

check for updates

Citation: Song, T.; Qiao, Y.; He, Y.; Wu, N.; Li, Z.; Liu, B. Dual-Arm Cluster Tool Scheduling for Reentrant Wafer Flows. *Electronics* **2023**, *12*, 2411. https://doi.org/10.3390/electronics12112411

Academic Editor: Katarzyna Antosz

Received: 16 April 2023
Revised: 22 May 2023
Accepted: 22 May 2023
Published: 26 May 2023

1. Introduction

In semiconductor manufacturing, cluster tools have been widely adopted for wafer fabrication. Such a tool compactly integrates several *process modules* (PMs), a robot, and two *loadlocks* (LLs). Moreover, it adopts a single-wafer processing technology such that wafers are processed one by one in a PM. With a single- or dual-arm robot, a tool is called a *single-arm cluster tool* (SACT) or *dual-arm cluster tool* (DACT), as shown in Figure 1a,b.

In an SACT or a DACT as shown in Figure 1, PMs are filled with chemicals for wafer processing, and the internal temperature in PMs is required to be high for some wafer fabrication processes. The robot in the center of a tool is used to transport wafers among the PMs. LLs have two doors. One faces the internal chamber in which there are several PMs, while the other faces outside. LLs are used to ensure the internal vacuum processing environment.

In semiconductor fabs, 25 wafers are grouped into a lot and held in a front opening unified pod (FOUP) [1]. FOUPs are transported to the right areas for wafer fabrication according to recipes. The recipes of wafers in a lot are identical. In a cluster tool, after a wafer lot is loaded into an LL, the LL is pumped into a vacuum environment. Then, the wafers can be delivered to PMs to be processed according to their recipes. After wafers in an LL are completed, then the raw wafers from the other LL are fed to PMs to be processed such that the tool can operate under a steady state without interruption [1,2].

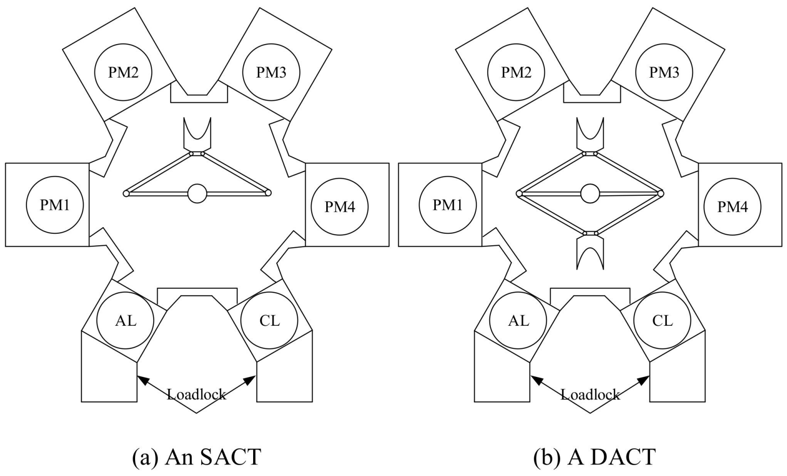

Figure 1. Cluster tools: (**a**) an SACT with a single-arm robot and four PMs; (**b**) a DACT with a dual-arm robot and four PMs.

For some wafer fabrication processes, wafers are required to visit some PMs for more than one time. Such a wafer fabrication process is called a reentering process. Atomic layer deposition (ALD) and plasma-enhanced chemical vapor deposition (PECVD) are such typical processes. For the ALD process, deposition processes should be repeatedly performed several times, even more than five. The first deposition layer requires three process steps: Al_2O_3 deposition, Ta_2O_5 deposition, and oxidation process. Each subsequent deposition layer repeats the last two process steps [3]. For a PECVD process, PECVD tools are used for depositing thin films onto silicon wafer substrates, which is one of the crucial steps in the manufacturing of microelectronic circuits and solar cells [4]. For reentering processes, the reentering operations should be performed under identical processing conditions. This presents a requirement that only one PM is configured to serve a processing step.

For a DACT with reentrant wafer flows, Qiao et al. [5] developed a one-wafer periodical (1-WP) scheduling method by adjusting the processing progresses of wafers at different steps such that a 1-WP schedule can be obtained. Moreover, theoretical proofs for its optimality are provided in [5]. However, the study is done for a 2-time reentering process, and it is not known if it is applicable to processes with the reentering times being more than two. In practice, for some reentering processes, wafers are required to visit some PMs for k ($\geq$2) times. Recently, it has been found that a 1-WP schedule cannot be obtained for a DACT with k-time reentrant processes if there exists a value $f \in \mathbb{N} = \{1, 2 \ldots \}$ such that $k = 3f$ holds. This implies that the method in [5] is not applicable to such cases, which motivates us to do this work. Further, this work aims at solving the scheduling problem of DACTs with k-time ($k \geq 3$) reentrant processes such that the solutions to DACTs with reentrant processes are complete.

2. Literature Review

In SACTs and DACTs, the time taken for robot tasks, including placing, picking, and moving, is actually quite short in comparison with wafer processing time in PMs. Thus, for SACTs, a backward strategy is optimal in terms of productivity [6,7], while for DACTs, a swap strategy is optimal [8]. Further, studies on modeling and scheduling SACTs and DACTs were conducted in [8–10]. With two wafer types and one shared PM, the scheduling analysis was done for DACTs in [11]. The results obtained in the above-mentioned studies are based on the assumption that there are no *wafer residency time constraints* (WRTCs).

For some wafer fabrication processes, WRTCs are imposed, which requires that after a wafer is processed in a PM, it should be removed from the PM within a limited time interval, otherwise it would be damaged by the high temperature and chemical gas in the PM. In [12,13], methods were proposed to find optimal periodical schedules for DACTs with WRTCs. Further, based on Petri nets, a *mixed integer programming* (MIP) method was presented in [14] to get optimal cyclic schedules for both SACTs and DACTs with various wafer processing flows. To improve the computational efficiency in finding an optimal solution for SACTs and DACTs with WRTCs, schedulability conditions under which a feasible schedule exists were established in [15,16]. If schedulable checked by such conditions, the authors established analytical expressions to find optimal solutions. With multiple wafer types, wafer delay analysis and workload balancing of parallel PMs were conducted in [17]. Moreover, PM configuring problem of residency time-constrained DACTs was investigated, and a polynomial-complexity algorithm was developed to find optimal cyclic schedules in [18].

In practice, the time required for robot activities and wafer processing might be disturbed. Such a time variation may result in a feasible schedule obtained under the deterministic activity time assumption becoming infeasible. Thus, a robust scheduling method is necessary for cluster tools with activity time variation. To do so, in [19], based on Petri nets, a real-time control policy was proposed for DACTs with WRTCs and activity time variation to offset the activity time disturbance as much as possible by adjusting the robot waiting time in real-time. Then, an optimal real-time scheduling method consisting of an off-line schedule and a real-time control policy was presented in [20]. Since the robot task sequences for DACTs and SACTs are different, the methods for DACTs in [19,20] are not applicable to SACTs. Thus, in [21], for SACTs with a backward strategy, a real-time control policy was proposed to reduce the impact caused by activity time variation as much as possible and analytical expressions were given to calculate the upper bound of wafer sojourn time delay. Then, based on the results obtained in [21], an optimal real-time scheduling method was proposed in [22] for SACT with WRTCs and activity time variation. In [23], a class of schedules was proposed for cluster tools to keep timing patterns as steady as possible and adopt timing of tasks in response to process time variation so as to satisfy WRTCs robustly. To ensure the consistency of wafer sojourn time, ref. [24] examined the conditions under which a feedback controller proposed in their previous work can stabilize the wafer sojourn time in a stochastic processing environment with unexpected random time disturbances.

As the wafer size becomes larger, while the circuit width shrinks down, the wafer fabrication constraints are more and more strict. Periodical chamber cleaning operations are normally required for a PM in cluster tools after a PM processes a specified number of wafers so as to ensure the processing environment. Let h denote the number of wafers that a PM processes at most before it requires a chamber cleaning operation. When $h = 1$, the chamber cleaning operation is called a purge cleaning operation. To improve the productivity of SACTs and DACTs with purge operations, a backward(z) strategy is presented in [25] for SACTs based on the conventional backward strategy, while a swap(a, z) strategy is presented in [26,27] for DACTs based on the conventional swap strategy. By considering more general cases with $h \geq 1$, Qiao et al. [28] presented a virtual wafer-based method for DACTs to deal with chamber cleaning requirements. Besides, in [29,30], efficient scheduling approaches were proposed to deal with the chamber cleaning operation issue caused by the processing environment detection.

All the above-mentioned studies were conducted for cluster tools without reentrant processes. In fact, for an SACT or a DACT without reentrant processes, it can be treated as a flow-shop system. However, with reentrant processes, it is not, therefore, the above-mentioned studies are not applicable. Thus, for cluster tools with reentrant processes, system behavior was modeled by *Petri nets* (PNs) in [31] for performance analysis without tackling their optimization problems. Then, in [3], these problems were addressed for SACTs based on a developed PN model and formulated by an MIP model. To improve the computational efficiency, based on a *resource-oriented PN* (ROPN) model, an analytical method was proposed to schedule the overall system with reentrant processes in [32]. For DACTs with reentrant processes, Wu et al. [33,34] pointed out that the tool may operate under a transient process for some cases all the time based on a *three-wafer periodical* (3-WP) scheduling method. This means that the obtained results under the steady state in [31] are not applicable. Further, ref. [35] presented the cycle time analysis for DACTs with k-time ($k \geq 3$) reentrant processes based on the 3-WP scheduling method. Then, Qiao et al. [5] developed a 1-WP scheduling method by adjusting the processing progresses of wafers at different steps and theoretically proved its optimality. Furthermore, for time-constrained DACTs with reentrant processes, the schedulability and scheduling analysis was carried out in [36,37] based on such a 1-WP schedule. By taking the activity time variation into account, for time-constrained DACTs with reentrant processes, efficient algorithms were developed to calculate the upper bound of wafer sojourn time delay in [38] and an optimal real-time scheduling method was proposed in [31] to operate DACTs.

The existing studies for cluster tools with reentrant wafer flows are summarized in Table 1. In [31], it did not present a method to obtain optimal schedules. In [5], although the proposed MIP model can find optimal solutions, it would take a long time to solve the model as the number of reentering times increases. In [32], deadlock control policies were presented, and efficient scheduling methods were proposed for SACTs. In [5,33,34,36–39], the scheduling problem of DACTs with two-time reentrant processes was fully investigated. Moreover, by the swap strategy, a 1-WP scheduling method was found in [5] to achieve the lower bound of cycle time. However, as mentioned in the Introduction, in a case study for DACTs with three-time reentrant processes, it was found that a 1-WP schedule cannot be obtained by the swap strategy. Thus, it gives rise to a question of how to schedule DACTs with three-time reentrant processes so as to maximize productivity. With such motivation, this work is conducted.

Table 1. The existing studies for cluster tools with reentrant wafer flows.

References	Number of Reentrant Times	Other Constraints	The Addressed Problem	Methods	Results
[31]	$k \geq 2$	None	Deadlock analysis	PNs	No optimality analysis
[3]	$k \geq 2$	None	Scheduling	PNs and MIP	Optimal
[32]	$k = 2$	None	Scheduling	PNs	Optimal for SACTs
[33]	$k = 2$	None	Scheduling	PNs and 3-WP scheduling	Optimal for some cases
[34]	$k = 2$	None	Scheduling	PNs and 2-WP scheduling	Optimal for some cases
[5]	$\boldsymbol{k = 2}$	**None**	**Scheduling**	**PNs and 1-WP**	**Optimal**
[36,37]	$k = 2$	WRTCs	Scheduling	1-WP	Optimal
[38,39]	$k = 2$	WRTCs, time variation	Control and Scheduling	PNs and 1-WP	Optimal
[35]	$\boldsymbol{k \geq 2}$	**None**	**Cycle time analysis**	**PNs and 3-WP**	**Optimal for some cases**

In this work, if there does not exist a value $f \in \{1, 2 \ldots\}$ such that $k = 3f$, theoretical proofs are given to show that a 1-WP schedule can be found for DACTs with k-time reentrant processes. Furthermore, for cases with a 1-WP schedule, simple expressions are given to calculate the cycle time. For the cases without a 1-WP schedule, two new 3-WP scheduling methods are proposed to improve the system productivity by comparing it with an existing 3-WP scheduling method presented in [35]. In summary, compared with the existing studies, this work aims to tackle the open problem in this research field and makes significant improvements.

In the next section, the reentrant processes and the 1-WP schedule are introduced. Then, for k-time reentrant processes with $k \geq 3$, Section 4 shows that a 1-WP schedule cannot be found if there exists a value $f \in \mathbb{N}$ such that $k = 3f$ holds. In Section 5, two novel methods are proposed to improve the productivity of DACTs for the cases where a 1-WP schedule cannot be found. The applications of the obtained results are demonstrated by examples in Section 6, and this work is concluded in Section 7.

3. The Reentrant Process and Periodical Schedules

3.1. Reentrant Process

PMs in a cluster tool are divided into different groups. The PMs in the same group serve to perform the same fabrication operation called a step. For serial wafer flows, raw wafers sequentially visit several steps according to their processing recipes. For reentrant wafer flows (i.e., reentrant processes), wafers should revisit some steps multiple times. Notice that, for the steps involving reentrant processes, only one PM is configured so as to ensure processing consistency. For example, ALD and PECVD have such a requirement. Thus, if a wafer is repeatedly processed at a reentrant step, the processing environment is exactly identical such that the processing quality can be ensured.

There are three processing steps for the ALD process. After a raw wafer is moved out of an LL by the robot, the wafer is delivered to Step 1 to be processed, then Step 2, and followed by Step 3. When the processing of the wafer in Step 3 is completed, it revisits Steps 2 and 3 again such that a wafer visits Steps 2 and 3 totally $k \geq 2$ times. Note that each step of Steps 2 and 3 has only one PM. In fact, the workloads at the reentrant steps are much greater than that in Step 1. Thus, more than one PM used for Step 1 cannot improve the productivity of a cluster tool. Therefore, only one PM is used to serve for Step 1 as well. Moreover, it can save cost to operate such a tool in this way since a PM (i.e., a chamber) is quite expensive. PM_i, $i \in \{1, 2, 3\}$, is used for Step i. Then, the wafer flow pattern is denoted as $(PM_1, (PM_2, PM_3)^k)$, with $(PM_2, PM_3)^k$ being the reentrant process.

Another commonly seen reentrant process is PECVD. For PECVD, it has two steps (i.e., Steps 1 and 2), and both are reentrant ones. PM_1 is used to complete plasma-enhanced chemical vapor deposition in Step 1, while PM_2 in Step 2 is used to complete a cure operation so as to ensure wafer quality. Then, $(PM_1, PM_2)^k$, $k \geq 2$, is used to represent the wafer flow pattern of PECVD. By observing the wafer flow patterns of ALD and PECVD, the reentrant pattern of PECVD is a special case of ALD from the perspective of scheduling. Thus, this work focuses on the scheduling analysis of DACTs with $(PM_1, (PM_2, PM_3)^k)$ with $k \geq 2$, which is commonly seen in cluster tool scheduling with reentrant processes. Note that the results in [5] are conducted for DACTs with $(PM_1, (PM_2, PM_3)^k)$ as well.

3.2. Activity Description

In a DACT, the two robot arms are named Arm-1 and Arm-2, respectively. Robot activities include picking a wafer from a PM, moving between two PMs, placing a wafer into a PM, rotating, and waiting. For a DACT, a swap strategy is efficient. At a state, assume that Arm-1 is empty and stays at PM_i, while Arm-2 carries a wafer. At this state, a swap operation at PM_i, $i \in \{1, 2, 3\}$, includes the following activities: Picking a processed wafer from PM_i by Arm-1 $\rightarrow$ rotating $\rightarrow$ placing the wafer held by Arm-2 into PM_i. The robot picking and placing activities at PM_i are denoted as PI_i and PL_i, respectively. A swap operation at PM_i is denoted as SWP_i. Thus, SWP_i includes PI_i, robot rotation, and PL_i. Besides, M_{ij} is used to denote the robot moving from Steps i to j. Note that LLs are denoted as Step 0.

To model the time aspect, we assume that the time needed to execute each of the above-mentioned robot activities is constant. The activities of executing PI_i, PL_i, and M_{ij} spend α, β, and μ time units, respectively. In practice, the time taken for SWP_i is often less than the sum of the time for performing PI_i, robot rotation, and PL_i. Thus, symbol λ is introduced to represent the time needed for SWP_i. Except for the robot activities, the wafer processing time at PM_i, $i \in \{1, 2, 3\}$, is denoted as ρ_i. The meanings of the notations are summarized in Table 2.

Table 2. Robot and processing activities.

Notations	Robot Tasks	Time
PI_i	Picking a wafer in Step *i*	α
PL_i	Placing a wafer in Step *i*	β
M_{ij}	Moving from Steps *i* to *j*	μ
SWP_i	Swapping in Step *i*	λ

3.3. Periodical Schedules

In a cluster tool, raw wafers enter the system one by one. The *d*-th wafer entering the system is denoted as W_d, $d \in \mathbb{N}$. To find a periodical schedule, it is necessary to analyze the state evolution of the system. Let $\Theta_i = \{W_d(q)\}$, $i \in \{1, 2, 3\}$, denote the state of PM_i (Step *i*), indicating that W_d is processed in PM_i for the *q*-th operation. Furthermore, let $\Theta_4 = \{R_i(W_d(q)\}$ denote the state of the robot, representing that the robot is staying at PM_i, $i \in \{1, 2, 3\}$, and at the same time, carrying W_d with its *q*-th operation to be processed in the PM. Then, the state of the system is denoted as $S = \{\Theta_1, \Theta_2, \Theta_3, \Theta_4\}$.

For a DACT with (PM_1, $(PM_2, PM_3)^2$), by a swap strategy, Wu et al. [33] present a three-wafer periodical schedule called a 3-WP schedule in short. With a 3-WP schedule, starting from $S_1 = \{W_3(1), W_2(2), W_1(3), R_1(W_4(1))\}$, a DACT evolves as follows: $S_1 = \{W_3(1), W_2(2), W_1(3), R_1(W_4(1))\} \rightarrow S_2 = \{W_4(1), W_3(2), W_1(3), R_3(W_2(3))\} \rightarrow S_3 = \{W_4(1), W_1(4), W_2(3), R_3(W_3(3))\} \rightarrow S_4 = \{W_4(1), W_2(4), W_3(3), R_3(W_1(5))\} \rightarrow S_5 = \{W_4(1), W_3(4), W_1(5), R_3(W_2(5))\} \rightarrow S_6 = \{W_4(1), W_3(4), W_2(5), R_1(W_5(1))\} \rightarrow S_7 = \{W_5(1), W_4(2), W_3(5), R_1(W_6(1))\} \rightarrow S_8 = \{W_6(1), W_5(2), W_4(3), R_1(W_7(1))\} \rightarrow S_9 = \{W_7(1), W_6(2), W_4(3), R_3(W_5(3))\}$. Notice that S_1 and S_8 are equivalent. It implies that evolution from S_1 to S_8 forms a period.

Note that, to transfer S_1 to S_2 and S_8 to S_9, robot task sequence $\sigma_1 = \langle SWP_1 \rightarrow M_{12} \rightarrow SWP_2 \rightarrow M_{23}\rangle$ is performed, i.e., the robot sequentially performs the following robot tasks: Swaps at PM_1, moves to PM_2 from PM_1, swaps at PM_2, and moves to PM_3 from PM_2. To transfer S_2 to S_3, $\sigma_2 = \langle SWP_3 \rightarrow M_{32} \rightarrow SWP_2 \rightarrow M_{23}\rangle$ is performed, i.e., the robot sequentially performs the following robot tasks: Swaps at PM_3, moves to PM_2 from PM_3, swaps at PM_2, and moves to PM_3 from PM_2. σ_2 is repeated for S_3 to S_4 and S_4 to S_5. Note that σ_2 forms a robot task cycle involving the reentrant process $(PM_2, PM_3)^2$, and it is called a local cycle. $\sigma_3 = \langle SWP_3 \rightarrow M_{30} \rightarrow PL_0 \rightarrow PI_0 \rightarrow M_{01}\rangle$ is for S_5 to S_6. This means that the robot sequentially performs the following robot tasks for σ_3: Swaps at PM_3, moves to an LL from PM_3, place a wafer into the LL, pick a wafer from the LL, and moves to PM_1 from the LL. $\sigma_4 = \langle SWP_1 \rightarrow M_{12} \rightarrow SWP_2 \rightarrow M_{23} \rightarrow SWP_3 \rightarrow M_{30} \rightarrow PL_0 \rightarrow PI_0 \rightarrow M_{01}\rangle$ is for S_6 to S_7 and S_7 to S_8. This means that the robot sequentially performs the following robot tasks for σ_4: Swaps at PM_1, moves to PM_2 from PM_1, swaps at PM_2, moves to PM_3 from PM_2, swaps at PM_3, moves to an LL from PM_3, place a wafer into the LL, pick a wafer from the LL, and moves to PM_1 from the LL. Obviously, σ_4 is a robot cycle involving all PMs, and it is called a global cycle. Moreover, σ_1 and σ_3 together form a global cycle as well. Thus, as shown in Figure 2, a period from S_1 to S_8 (or S_2 to S_9) contains three local and three global cycles. Notice that, in each global cycle, one wafer with all operations being completed is returned to LLs. Thus, three wafers are returned to LLs in this period.

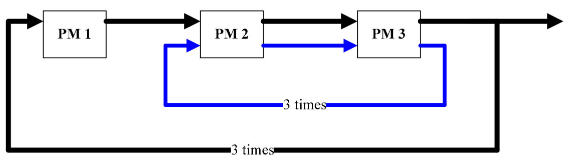

Figure 2. A 3-WP schedule for a DACT with (PM_1, $(PM_2, PM_3)^2$).

For a DACT with (PM_1, $(PM_2, PM_3)^2$), by a 3-WP schedule, as shown in Figure 2, Wu et al. [33] point out that the tool may operate under a transient process in some cases all the time. In such cases, once the robot enters the global cycles from local cycles, there might be a time delay when the robot arrives at the PMs involving reentrant processes. Such time delay makes that the lower bound of the system cycle time cannot be reached, i.e., a 1-WP schedule may not be optimal in such cases. This also implies that productivity reduction is caused by multiple local and global cycles in a period. Thus, it raises the question of whether the performance of the system can be improved by reducing the number of local and global cycles. To answer this question, for a DACT with (PM_1, $(PM_2, PM_3)^2$), Qiao et al. [5] present a 1-WP schedule.

By a 1-WP schedule, the tool should start to operate from a steady state, i.e., S_1 = {$W_3(1)$, $W_1(4)$, $W_2(3)$, $R_1(W_4(1))$}. Then, by performing σ_1, the system enters state S_2 = {$W_4(1)$, $W_3(2)$, $W_2(3)$, $R_3(W_1(5))$}. Further, by executing σ_2, it reaches S_3 = {$W_4(1)$, $W_2(4)$, $W_1(5)$, $R_3(W_3(3))$}. Finally, by executing σ_3, S_4 = {$W_4(1)$, $W_2(4)$, $W_3(3)$, $R_1(W_5(1))$} is reached. At this time, S_1 and S_4 are equivalent. Thus, as shown in Figure 3, a period with a local and a global cycle is formed. Moreover, during such a period, a wafer is returned to LLs.

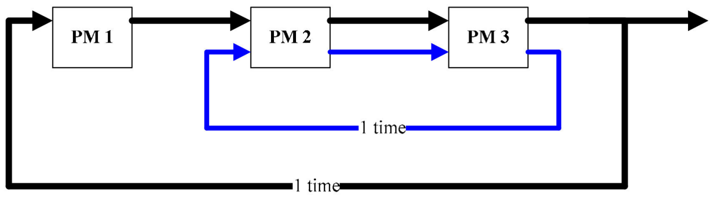

Figure 3. A 1-WP schedule for a DACT with (PM_1, $(PM_2, PM_3)^2$).

For a DACT with (PM_1, $(PM_2, PM_3)^2$), Qiao et al. [5] proved that the 1-WP schedule is optimal in terms of cycle time. However, in the cases where wafers are required to visit some PMs for k (>2) times, if there exists a value $f \in \mathbb{N} = \{1, 2 \ldots \}$ such that $k = 3f$ holds, it is found that a 1-WP schedule cannot be obtained for a DACT, resulting in that the method in [5] is not applicable to such cases. It motivates us to conduct this work. Next, this work gives theoretical proofs for the above findings and provides the analytical expressions of the system cycle time if a 1-WP schedule exists for a DACT with (PM_1, $(PM_2, PM_3)^k$).

4. Scheduling Analysis by One-Wafer Cyclic Schedule

For a DACT with (PM_1, $(PM_2, PM_3)^k$), $k \geq 3$, with a 3-WP schedule, the cycle time analysis has been conducted in [35]. It is found that there are $(3k - 3)$ local cycles and three global cycles in a period. Thus, before a completed wafer goes back to LLs in a global cycle, it should undergo $(3k - 3)$ local cycles to complete the reentrant process. Furthermore, when this wafer is returned to LLs, it has completed its $(2k + 1)$-th operation. Assume that a 1-WP schedule exists for a DACT with (PM_1, $(PM_2, PM_3)^k$). Then, the 1-WP schedule should result in a one-wafer period with multiple local cycles and a global cycle. Since only one completed wafer is returned to LLs in such a period, this wafer has already experienced $(3k - 3)$ local cycles in fact. Notice that if $(3k - 3)$ local cycles are consecutively performed, three wafers should be returned to LLs by three consecutive global cycles. This is a 3-WP schedule. If a 1-WP schedule exists, an obtained period should have $(k - 1)$ local cycles first and then a global cycle such that a wafer is returned to LLs during each global cycle. Further, with a 1-WP schedule, there should be a state, i.e., S_1 = {$W_1(1)$, $W(\vartheta_1)$, $W(2k + 1)$, $R_3(W(\vartheta_2))$} at which the last local cycle in a period is just completed.

Theorem 1. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), if there exists a value $f \in \mathbb{N} \cup \{0\}$ such that $k = 3f + 2$ holds, then the system can be scheduled by a 1-WP schedule.*

Proof. Starting from $S1$ = {$W_1(1)$, $W(\vartheta_1)$, $W(2k + 1)$, $R_3(W(\vartheta_2))$}, S_2 = {$W_2(1)$, $W_1(2)$, $W(\vartheta_2)$, $R_3(W(\vartheta_1 + 1))$} is reached after a global cycle. Thus, the tool then evolves in local cycles. After $(k - 1)$ local cycles, the robot stays at PM_3 and prepares to place W_1 into PM_3 for processing the (2f + 3)-th operation. Then, the tool undergoes a global cycle such that W_1 is processed at PM_3 for the (2f + 3)-th operation. Further, after (k − 1) local cycles, W_1 is processed at PM_2 for the (4f + 4)-th operation. In the next global cycle, the robot picks W_1 from PM_2 and moves to PM_3, implying that the robot is going to place W_1 into PM_3 for processing the (4f + 5)-th operation. The following evolution also undergoes (k − 1) local cycles. After that, W_1 is processed at PM_3 for the (6f + 5)-th operation. Note that, the robot is staying at PM_3 at this time. With $k = 3f + 2$, $(6f + 5) = 2k + 1$ holds. This means that W_1 can be returned to LLs in the next global cycle. Further, by repeatedly performing (k − 1) local cycles and a global cycle, wafers (i.e., W_d, $d \in \mathbb{N}\backslash\{1\}$) are continuously completed and returned to LLs. Hence, the system can be scheduled by a 1-WP schedule. □

In this case, for a DACT with (PM_1, $(PM_2, PM_3)^k$), $k \geq 2$, if k is known and there exists a value $f \in \mathbb{N} \cup \{0\}$ such that $k = 3f + 2$ holds, Theorem 1 provides a simple way to find a state (i.e., S_2 = {$W_2(1)$, $W_1(2)$, $W(2f + 3)$, $R_3(W(4f + 5))$}) starting from which a 1-WP schedule exists.

Theorem 2. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), if there exists a value $f \in \mathbb{N}$ such that $k = 3f$ holds, then the system cannot be scheduled by a 1-WP schedule.*

Proof. Starting from marking S_1 = {$W_1(1)$, $W(\vartheta_1)$, $W(2k + 1)$, $R_3(W(\vartheta_2))$}, S_2 = {$W_2(1)$, $W_1(2)$, $W(\vartheta_2)$, $R_3(W(\vartheta_1 + 1))$} is reached after a global cycle. Thus, the tool then evolves in local cycles. After (k − 1) local cycles, the robot is at PM_3, and wafer W_1 is just being processed at the PM for the (2f + 1)-th operation. By a 1-WP schedule, a global cycle should be performed next. In this global cycle, W_1 should be delivered to LLs without completing all operations. Hence, the theorem holds. □

In this case, a 1-WP schedule is not applicable. Thus, this work presents two novel scheduling methods for this case in the next section to improve the system productivity.

Theorem 3. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), if there exists a value $f \in \mathbb{N}$ such that $k = 3f + 1$ holds, then the system can be scheduled by a 1-WP schedule.*

Proof. Starting from S_1 = {$W_1(1)$, $W(\vartheta_1)$, $W(2k + 1)$, $R_3(W(\vartheta_2))$}, S_2 = {$W_2(1)$, $W_1(2)$, $W(\vartheta_2)$, $R_3(W(\vartheta_1 + 1))$} is reached after a global cycle. Thus, the tool then evolves in local cycles. After (k − 1) local cycles, wafer W_1 is just being processed at PM_2 for the (2f + 2)-th operation. Then, after a global cycle, the robot picks W_1 from PM_2, moves to PM_3, and stays there. By a 1-WP schedule, the following state evolution is for local cycles. After (k − 1) local cycles, the robot is at PM_3 with W_1 being held to be placed into the PM for the (4f + 3)-th operation. After the wafer is placed into the PM, W_1 is processed in PM_3 for the (4f + 3)-th operation, and then the system just enters the next global cycle. Then, after a global cycle, the system evolves for (k − 1) local cycles. After that, W_1 is processed at PM_3 for the (6f + 3)-th operation. Due to $k = 3f + 1$, $(6f + 3) = 2k + 1$ holds, implying that W_1 can be returned to LLs in the next global cycle. Further, by repeatedly performing (k − 1) local cycles and a global cycle, wafers (i.e., W_d, $d \in \mathbb{N}\backslash\{1\}$) are continuously completed and returned to LLs. Hence, the system can be scheduled by a 1-WP schedule. □

In this case, for a DACT with (PM_1, $(PM_2, PM_3)^k$), $k \geq 2$, if k is known and there exists a value $f \in \mathbb{N}$ such that $k = 3f + 1$ holds, Theorem 3 provides a simple way to find a state (i.e., S_2 = {$W_2(1)$, $W_1(2)$, $W(4f + 3)$, $R_3(W(2f + 3))$}) starting from which a 1-WP schedule exists. Then, for the cases where a 1-WP schedule is applicable to a DACT with (PM_1, $(PM_2, PM_3)^k$), $k \geq 2$, the cycle time is analyzed next.

For DACTs, in Step i (PM_i), a swap operation makes a processed wafer removed from the PM and a new wafer placed into the PM. Then, the PM starts to process the wafer. When the robot comes to the PM again to perform a swap operation, the wafer in the PM is picked up, and a new wafer is placed into the PM again. Thus, the time taken to complete a wafer in Step *i* (i.e., the workload) is

$$\Pi_i = \rho_i + \lambda, i \in \{1, 2, 3\} \tag{1}$$

Let φ and ψ denote the time taken for a local and a global cycle without considering the robot waiting time, respectively. Then, they can be calculated as follows.

$$\varphi = 2\lambda + 2\mu \tag{2}$$

$$\psi = \alpha + \beta + 3\lambda + 4\mu \tag{3}$$

Further, let $\Pi_{local} = max\{\Pi_2, \Pi_3, \varphi\}$ and $\Pi_{1\text{-}WP}$ be the cycle time of a DACT with (PM_1, $(PM_2, PM_3)^k$), $k \geq 2$, if a 1-*WP* schedule exists. Then, according to the cycle time analysis for a 1-WP schedule in [5], if $\Pi_1 \leq (k-1)\Pi_{local} + \psi$, Corollaries 1 and 2 are given below.

Corollary 1. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), when there does not exist a value $f \in \mathbb{N}$ such that $k = 3f$ holds, by a 1-WP schedule, if $\Pi_1 \leq (k-1)\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, the cycle time is*

$$\Pi_{1\text{-}WP} = (k-1)\Pi_{local} + \psi \tag{4}$$

In fact, the time taken for the $(k-1)$ local cycles is $(k-1)\Pi_{local}$. In this case, due to $\Pi_1 \leq (k-1)\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, when the robot comes to PM_i, $i \in \{1, 2, 3\}$, the PM has completed the wafer processing. Thus, the robot can perform a swap operation immediately such that there is no robot waiting in the global cycle. Therefore, the time taken for the global cycle is ψ. Hence, in this case, (4) holds.

Corollary 2. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), when there does not exist a value $f \in \mathbb{N}$ such that $k = 3f$ holds, by a 1-WP schedule, if $\Pi_1 \leq (k-1)\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} > \psi$, the cycle time is*

$$\Pi_{1\text{-}WP} = k\Pi_{local} \tag{5}$$

In this case, due to $\Pi_1 \leq (k-1)\Pi_{local} + \psi$, it implies that when the robot comes to PM_1, PM_1 has completed the wafer processing. Thus, the robot can perform a swap operation immediately. However, due to $max\{\Pi_2, \Pi_3\} > \psi$, it implies that the time taken for the global cycle is $max\{\Pi_2, \Pi_3\}$. Therefore, (5) holds. Further, if $\Pi_1 > (k-1)\Pi_{local} + \psi$, according to the cycle time analysis for a 1-WP schedule in [5], Corollaries 3–5 are given below.

Corollary 3. *For a DACT with (PM_1, $(PM_2, PM_3)^k$), when there does not exist a value $f \in \mathbb{N}$ such that $k = 3f$ holds, by a 1-WP schedule, if $k\Pi_{local} \geq \Pi_1 > (k-1)\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} > \psi$, the cycle time is*

$$\Pi_{1\text{-}WP} = k\Pi_{local} \tag{6}$$

In this case, due to $\Pi_1 > (k-1)\Pi_{local} + \psi$, it implies that when the robot comes to PM_1, the robot has to wait for some time since the PM has not completed the wafer processing yet at this time. The robot waiting time at PM_1 should be $\Pi_1 - [(k-1)\Pi_{local} + \psi]$. Without loss of generality, in this case, let $\Pi_3 = max\{\Pi_2, \Pi_3\} = \Pi_{local} > \psi$. This means that in a global cycle, the total robot waiting time should be $\Pi_{local} - \psi$ at least. Note that $\Pi_1 - [(k-1)\Pi_{local} + \psi] \leq k\Pi_{local} - [(k-1)\Pi_{local} + \psi] = \Pi_{local} - \psi$, indicating that, in a global cycle, although the robot has waited at PM_1 for $\Pi_1 - [(k-1)\Pi_{local} + \psi]$ time units, it still needs to wait at PM_3 such that the time taken for the global cycle is Π_{local}. Therefore, (6) holds.

Corollary 4. *For a DACT with (PM_1, (PM_2, PM_3)^k), when there does not exist a value $f \in \mathbb{N}$ such that $k = 3f$ holds, by a 1-WP schedule, if $\Pi_1 > k\Pi_{local}$ and $\max\{\Pi_2, \Pi_3\} > \psi$, the cycle time is*

$$\Pi_{1\text{-}WP} = \Pi_1 \tag{7}$$

Corollary 5. *For a DACT with (PM_1, (PM_2, PM_3)^k), when there does not exist a value $f \in \mathbb{N}$ such that $k = 3f$ holds, by a 1-WP schedule, if $\Pi_1 > (k-1)\Pi_{local} + \psi$ and $\max\{\Pi_2, \Pi_3\} \le \psi$, the cycle time is*

$$\Pi_{1\text{-}WP} = \Pi_1 \tag{8}$$

For the cases given by Corollaries 4 and 5, in a global cycle, when the robot comes to PM_1, the robot must wait there since the PM has not completed the wafer processing yet. However, when the robot comes to PM_i, $i \in \{2, 3\}$, the wafer in the PM has been completed, meaning that the workload at PM_1 dominates the system cycle time. Therefore, (7) and (8) for Corollaries 4 and 5 hold, respectively.

Besides, in the cases given by Corollaries 1–5, a 1-WP schedule can achieve the lower bound of the cycle time, i.e., it is optimal in terms of productivity. Up to now, the cycle time analysis has been done in all cases where a 1-WP schedule is applicable to a DACT with (PM_1, (PM_2, PM_3)^k). Based on the above analysis, to apply a 1-WP schedule, the key is to get the desired state of the system shown above, then, by starting from this state, the system can evolve with the 1-WP schedule.

5. Two Novel Scheduling Methods

As above-discussed, given the number k of revisiting times, if there is an $f \in \mathbb{N}$ such that $k = 3f$, then no 1-WP schedule can be found. Thus, in this case, there is an issue of how to schedule the system so as to improve productivity. This section aims to tackle this issue by proposing two novel scheduling methods.

For a DACT with (PM_1, (PM_2, PM_3)^k), $k \ge 2$, when the system evolves from local cycles to a global cycle, there might be wafer delay time in PMs involved in the reentrant process in the global cycle such that the system cycle time cannot reach its lower bound [33]. Thus, after the local cycles, if the number of global cycles can be decreased, the system cycle time can be improved. With this idea, the first scheduling method is proposed.

5.1. Scheduling Method One

Without loss of generality, the scheduling analysis is conducted for a DACT with (PM_1, (PM_2, PM_3)^3). Then, by the proposed method, a DACT with (PM_1, (PM_2, PM_3)^3) evolves as follows: $S_1 = \{W_4(1), W_3(2), W_1(5), R_3(W_2(5))\} \rightarrow S_2 = \{W_4(1), W_1(6), W_2(5), R_3(W_3(3))\} \rightarrow S_3 = \{W_4(1), W_2(6), W_3(3), R_3(W_1(7))\} \rightarrow S_4 = \{W_4(1), W_3(4), W_1(7), R_3(W_2(7))\} \rightarrow S_5 = \{W_5(1), W_4(2), W_2(7), R_3(W_3(5))\} \rightarrow S_6 = \{W_6(1), W_5(2), W_3(5), R_3(W_4(3))\} \rightarrow S_7 = \{W_6(1), W_3(6), W_4(3), R_3(W_5(3))\} \rightarrow S_8 = \{W_6(1), W_4(4), W_5(3), R_3(W_3(7))\} \rightarrow S_9 = \{W_6(1), W_5(4), W_3(7), R_3(W_4(5))\} \rightarrow S_{10} = \{W_7(1), W_6(2), W_4(5), R_3(W_5(5))\}$ with S_1 and S_{10} being equivalent. Thus, a period from S_1 to S_{10} is formed. Note that to reach S_2 from S_1, S_3 from S_2, S_4 from S_3, S_7 from S_6, S_8 from S_7, and S_9 from S_8, robot task sequence σ_2 for a local cycle is performed, respectively. To reach S_5 from S_4, S_6 from S_5, and S_{10} from S_9, $\sigma_5 = \langle SWP_3 \rightarrow M_{30} \rightarrow PL_0 \rightarrow PI_0 \rightarrow M_{01} \rightarrow SWP_1 \rightarrow M_{12} \rightarrow SWP_2 \rightarrow M_{23}\rangle$ is executed, respectively. Note that σ_5 is a global cycle.

During a period from S_1 to S_{10}, before two global cycles (i.e., the first global cycle from S_4 to S_5 and the second one from S_5 to S_6), three local cycles are performed at first.

Then, there are three local cycles again, and they are followed by one global cycle (i.e., the third global cycle from S_9 to S_{10}). Thus, after the local cycles, the number of global cycles followed is decreased by comparing with the 3-WP schedule presented in [33,35]. Moreover, in each global cycle, one wafer is returned to LLs. Totally, three wafers are returned to LLs in a period. Therefore, it is also a 3-WP schedule. However, it is different from the 3-WP schedule presented in [33,35], by which three wafers are completed in three consecutive global cycles in a period. Thus, it is a new 3-WP schedule, called an N3-WP1 schedule in short. Let $\Pi_{\text{N3-WP1}}$ denote the system cycle time by an N3-WP1 schedule.

Theorem 4. *For a DACT with (PM_1, (PM_2, PM_3)^3), by an N3-WP1 schedule, if $\Pi_1 \leq 3\Pi_{local} + \psi$ and max{Π_2, Π_3} $\leq \psi$, the cycle time is*

$$\Pi_{\text{N3-WP1}} = \begin{cases} 2\Pi_{local} + \psi,\ if\ \psi\ \geq\ \Pi_1 \\ \frac{6\Pi_{local} + 2\psi + \Pi_1}{3},\ if\ \psi < \Pi_1 \end{cases} \tag{9}$$

Proof. From S_1 to S_4, there are three local cycles that take $3\Pi_{local}$ time units. After that, the system performs two global cycles. Due to $\Pi_1 \leq 3\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, when the robot arrives at PM_i, $i \in \{1, 2, 3\}$, to pick up a wafer in the first global cycle from S_4 to S_5, the wafer should have been processed and can be picked up from the PM immediately. Therefore, the robot does not need to wait at PM_i, $i \in \{1, 2, 3\}$, such that ψ time units are needed for the first global cycle. Then, if $\psi < \Pi_1$, the next global cycle from S_5 to S_6 takes Π_1 time units, and otherwise, it takes ψ time units. From S_6 to S_9, the three local cycles take $3\Pi_{local}$ time units. For the global cycle from S_9 to S_{10}, ψ time units are required due to $\Pi_1 \leq 3\Pi_{local} + \psi$. Thus, by an N3-WP1 schedule, if $\psi \geq \Pi_1$, a period takes $6\Pi_{local} + 3\psi$ time units, and otherwise it takes $6\Pi_{local} + 2\psi + \Pi_1$ time units. Therefore, (9) holds. □

In this case, when $\psi < \Pi_1$, the result obtained by a 3-WP schedule is improved. Let ω_{1i}, ω_{2i}, and ω_{3i}, $i \in \{1, 2, 3\}$, denote the robot waiting time before a swap operation at PM_1, PM_2 and PM_3 in the i-th global cycle in a period, respectively. Moreover, let $\chi = \Pi_1 - \Pi_{local}$. The following result is achieved.

Theorem 5. *For a DACT with (PM_1, (PM_2, PM_3)^3), by an N3-WP1 schedule, if $\Pi_1 \leq 3\Pi_{local} + \psi$ and max{Π_2, Π_3} $> \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP1} = \begin{cases} 3\Pi_{local},\ if\ \Pi_{local} - \psi\ \geq\ \chi \\ 3\Pi_{local} + \frac{\chi + \psi - \Pi_{local}}{3},\ if\ \Pi_{local} - \psi < \chi \end{cases} \tag{10}$$

Proof. By Theorem 4, for the three local cycles from S_1 to S_4, $3\Pi_{local}$ time units are required. After that, the system undergoes two global cycles. Due to $\Pi_1 \leq 3\Pi_{local} + \psi$, when the robot arrives at PM_1 to pick up a wafer in the first global cycle from S_4 to S_5, the wafer should have been processed and can be picked up from the PM immediately. Therefore, $\omega_{11} = 0$. Further, due to $max\{\Pi_2, \Pi_3\} > \psi$, $\omega_{21} + \omega_{31} = \Pi_{local} - \psi$. In this way, the wafer in a PM with a higher workload of PM_2 and PM_3 can be picked up once it is processed. Thus, Π_{local} time units are needed for the first global cycle. In the second global cycle from S_5 to S_6, when the robot comes to PM_1, there are two cases: (1) $\Pi_{local} - \psi \geq \chi$ and (2) $\Pi_{local} - \psi < \chi$. In the first case, before the robot can pick a wafer up from PM_1, it should wait at the PM for $\omega_{12} = \Pi_1 - (\psi + \omega_{21} + \omega_{31}) = \Pi_1 - \Pi_{local} = \chi$ time units if $\chi \geq 0$, or $\omega_{12} = 0$ if $\chi < 0$. Then, due to $max\{\Pi_2, \Pi_3\} > \psi$, $\omega_{22} + \omega_{32} = \Pi_{local} - \psi - \omega_{12} = \Pi_{local} - \psi - max(\chi, 0) \geq 0$, implying that a wafer in a PM with the higher workload of PM_2 and PM_3 can be picked up once it is processed. Thus, the time taken for the second global cycle is Π_{local} in this case. In Case (2), before the robot can pick a wafer up from PM_1, it should wait at the PM for ω_{12} =

$\Pi_1 - (\psi + \omega_{21} + \omega_{31}) = \Pi_1 - \Pi_{local} = \chi > \Pi_{local} - \psi > 0$ time units. Further, when the robot comes to PM$_2$ or PM$_3$ to pick up a wafer, the wafer has already been processed due to $\Pi_{local} - \psi < \chi = \omega_{12}$ (i.e., $\Pi_{local} < \omega_{12} + \psi$). Thus, for the second global cycle, it spends $\omega_{12} + \psi = \chi + \psi$ time units. Similarly, the time taken for the process from S_6 to S_{10} is $4\Pi_{local}$. Hence, (10) holds. □

In this case, when $\Pi_{local} - \psi \geq \chi$, the cycle time is optimal. Furthermore, when $\Pi_{local} - \psi < \chi$, the result obtained by a 3-WP schedule is improved.

Theorem 6. *For a DACT with (PM$_1$, (PM$_2$, PM$_3$)^3), by an N3-WP1 schedule, if $4\Pi_{local} \geq \Pi_1 > 3\Pi_{local} + \psi$ and max{Π_2, Π_3} > ψ, the cycle time is*

$$\Pi_{N3\text{-}WP1} = (\Pi_1 + 7\Pi_{local} + \psi + max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\})/3 \quad (11)$$

Proof. Note that in the second global cycle from S_5 to S_6, due to $\Pi_1 > 3\Pi_{local} + \psi$, $\omega_{22} + \omega_{32} = 0$ should hold. Thus, from S_6 to S_{10}, it takes $4\Pi_{local}$ time units due to $4\Pi_{local} \geq \Pi_1 > 3\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} > \psi$. Further, in the third global cycle, $\omega_{13} = \Pi_1 - (3\Pi_{local} + \psi)$ and $\omega_{23} + \omega_{33} = \Pi_{local} - \omega_{13} - \psi = 4\Pi_{local} - \Pi_1$.

By Theorem 4, from S_1 to S_4, it takes $3\Pi_{local}$ time units. Further, with $\omega_{23} + \omega_{33} = 4\Pi_{local} - \Pi_1$, during the first global cycle from S_4 to S_5, $\omega_{11} = max(\Pi_1 - (3\Pi_{local} + \psi) - (\omega_{23} + \omega_{33}), 0) = max(\Pi_1 - (3\Pi_{local} + \psi) - (4\Pi_{local} - \Pi_1), 0) = max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\}$ and $\omega_{21} + \omega_{31} = \Pi_{local} - \psi - max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\}$. If $2\Pi_1 - \psi - 7\Pi_{local} > 0$, then $\Pi_{local} - \psi - max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\} = \Pi_{local} - \psi - 2\Pi_1 + \psi + 7\Pi_{local} = 8\Pi_{local} - 2\Pi_1 \geq 0$, i.e., $\omega_{21} + \omega_{31} \geq 0$. Therefore, the time taken for the first global cycle is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \Pi_{local}$.

In the second global cycle from S_5 to S_6, $\omega_{12} = \Pi_1 - \Pi_{local} + max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\} > 2\Pi_{local} + \psi + max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\}$ holds due to $\Pi_1 > 3\Pi_{local} + \psi$. Thus, $\omega_{22} + \omega_{32} = 0$. Therefore, the time taken for this global cycle is $\omega_{12} + \psi = \Pi_1 - \Pi_{local} + max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\} + \psi$.

Thus, the time taken for a period from S_1 to S_{10} is $\Pi_1 + 7\Pi_{local} + \psi + max\{2\Pi_1 - \psi - 7\Pi_{local}, 0\}$. With three wafers being completed in a period, the cycle time can be obtained by (11). □

Similar to Corollaries 4 and 5, the following two theorems are presented. Due to the space limit, their proofs are omitted.

Theorem 7. *For a DACT with (PM$_1$, (PM$_2$, PM$_3$)^3), by an N3-WP1 schedule, if $\Pi_1 > 4\Pi_{local}$ and max{Π_2, Π_3} > ψ, the cycle time is*

$$\Pi_{\text{N3-WP1}} = \Pi_1 \quad (12)$$

Theorem 8. *For a DACT with (PM$_1$, (PM$_2$, PM$_3$)^3), by an N3-WP1 schedule, if $\Pi_1 > 3\Pi_{local} + \psi$ and max{Π_2, Π_3} $\leq \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP1} = \Pi_1 \quad (13)$$

Note that in the cases presented by Theorems 7 and 8, the cycle time is optimal.

5.2. Scheduling Method Two

Up to now, the cycle time analysis for a DACT with (PM_1, (PM_2, PM_3)^3) by an N3-WP1 schedule has been done. With the N3-WP1 schedule, in some cases, the productivity of the system is improved over the existing 3-WP schedule. However, in some other cases, the optimal cycle time cannot be achieved. Thus, we present another scheduling method by which a DACT with $(PM_1, (PM_2, PM_3)^3)$ evolves as follows: $S_1 = \{W_4(1), W_3(2), W_2(3), R_3(W_1(7))\} \rightarrow S_2 = \{W_4(1), W_2(4), W_1(7), R_3(W_3(3))\} \rightarrow S_3 = \{W_5(1), W_4(2), W_3(3), R_3(W_2(5))\} \rightarrow S_4 = \{W_5(1), W_3(4), W_2(5), R_3(W_4(3))\} \rightarrow S_5 = \{W_5(1), W_2(6), W_4(3), R_3(W_3(5))\} \rightarrow S_6 = \{W_5(1), W_4(4), W_3(5), R_3(W_2(7))\} \rightarrow S_7 = \{W_5(1), W_3(6), W_2(7), R_3(W_4(5))\} \rightarrow S_8 = \{W_6(1), W_5(2), W_4(5), R_3(W_3(7))\} \rightarrow S_9 = \{W_6(1), W_4(6), W_3(7), R_3(W_5(3))\} \rightarrow S_{10} = \{W_7(1), W_6(2), W_5(3), R_3(W_4(7))\}$. Since S_1 and S_{10} are equivalent, a period is formed by this state evolution. In the above state evolution, to realize the process from S_1 to S_2, S_3 to S_4, S_4 to S_5, S_5 to S_6, S_6 to S_7, and S_8 to S_9, robot task sequence σ_1 for a local cycle is executed, respectively, while to reach S_3 from S_2, S_8 from S_7, and S_{10} from S_9, robot task sequence σ_5 for a global cycle is executed, respectively.

During the period from S_1 to S_{10}, at first, there is a local cycle followed by a global cycle (i.e., the first global cycle from S_2 to S_3). Then, there are four local cycles and then a global cycle (i.e., the second global cycle from S_7 to S_8) is followed. After that, there is a local cycle followed by a global cycle (i.e., the third global cycle from S_9 to S_{10}) again. In each global cycle, one wafer is completed. Thus, during the period, it is also a 3-WP schedule that is different from the one presented in [33,35]. In this work, the second scheduling method is called an N3-WP2 schedule in short. Let $\Pi_{\text{N3-WP2}}$ denote the system cycle time under an N3-WP2 schedule. Based on Corollary 1 and Theorem 4, we present Theorem 9, while based on Corollary 2 and Theorem 5, we can get Theorem 10. The proofs of Theorems 9 and 10 are omitted due to the space limit.

Theorem 9. *For a DACT with $(PM_1, (PM_2, PM_3)^3)$, by an N3-WP2 schedule, if $\Pi_1 \leq \Pi_{local} + \psi$ and $\max\{\Pi_2, \Pi_3\} \leq \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP2} = 2\Pi_{local} + \psi \tag{14}$$

Theorem 10. *For a DACT with $(PM_1, (PM_2, PM_3)^3)$, by an N3-WP2 schedule, if $\Pi_1 \leq \Pi_{local} + \psi$ and $\max\{\Pi_2, \Pi_3\} > \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP2} = 3\Pi_{local} \tag{15}$$

In these two cases, as presented in Theorems 9 and 10, the cycle time cannot be shortened anymore, implying that this is the lower bound. For an N3-WP2 schedule, ω_{1i}, ω_{2i}, and ω_{3i}, $I \in \{1, 2, 3\}$, are also used to denote the robot waiting time before a swap operation at PM_1, PM_2, and PM_3 in the *i*-th global cycle during a period, respectively. Then, the following result is presented.

Theorem 11. *For a DACT with $(PM_1, (PM_2, PM_3)^3)$, by an N3-WP2 schedule, if $2\Pi_{local} \geq \Pi_1 > \Pi_{local} + \psi$ and $\max\{\Pi_2, \Pi_3\} > \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP2} = 3\Pi_{local} \tag{16}$$

Proof. Due to $2\Pi_{local} \geq \Pi_1 > \Pi_{local} + \psi$, the time taken for the state evolution from S_3 to S_8 is $5\Pi_{local}$. Further, $\omega_{22} + \omega_{32} = 0$ holds. For the local cycle from S_8 to S_9, it takes Π_{local} time units. In the global cycle from S_9 to S_{10}, when the robot comes to PM_1, due to $\Pi_1 > \Pi_{local} + \psi$, it has to wait there for $\omega_{13} = \Pi_1 - (\Pi_{local} + \psi)$ time units. Further, we have $\omega_{23} + \omega_{33} = \Pi_{local} - \psi - \omega_{13} = 2\Pi_{local} - \Pi_1 \geq 0$. Then, the time taken for the global cycle from S_9 to S_{10} is $\psi + \omega_{13} + \omega_{23} + \omega_{33} = \Pi_{local}$. Similarly, for the local cycle from S_1 to S_2, it takes Π_{local} time units. For the global cycle from S_2 to S_3, when the robot comes to PM_1, due to $\Pi_1 >$

$\Pi_{local} + \psi$, it has to wait there for $\omega_{11} = max(\Pi_1 - (\Pi_{local} + \psi) - (\omega_{23} + \omega_{33}), 0) = max(\Pi_1 - (\Pi_{local} + \psi) - (2\Pi_{local} - \Pi_1), 0) = max(2\Pi_1 - 3\Pi_{local} - \psi, 0)$ time units. Then, there are two cases: (1) $\omega_{11} = 0$ and (2) $\omega_{11} = 2\Pi_1 - 3\Pi_{local} - \psi$. In Case (1), $\omega_{21} + \omega_{31} = \Pi_{local} - \psi$, leading to that the time taken for the global cycle from S_2 to S_3 is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \Pi_{local}$. In Case (2), $\omega_{21} + \omega_{31} = \Pi_{local} - \psi - \omega_{11} = \Pi_{local} - \psi - (2\Pi_1 - 3\Pi_{local} - \psi) = 4\Pi_{local} - 2\Pi_1 \geq 0$ due to $2\Pi_{local} \geq \Pi_1$. Thus, in this case, the time taken for the global cycle from S_2 to S_3 is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \Pi_{local}$ as well. Therefore, it takes $9\Pi_{local}$ time units for the period from S_1 to S_{10}. With three wafers completed in the period, (16) holds. □

The cycle time cannot be shortened in this case either, i.e., it is optimal. Further, the following result for another case can be obtained.

Theorem 12. *For a DACT with (PM_1, (PM_2, PM_3)^3), by an N3-WP2 schedule, if* $4\Pi_{local} \geq \Pi_1 > 2\Pi_{local}$ *and* $max\{\Pi_2, \Pi_3\} > \psi$*, the cycle time is*

$$\Pi_{N3\text{-}WP2} = \begin{cases} 3\Pi_{local},\ if\ 5\Pi_{local} - 2\Pi_1 - \psi \geq 0 \\ \frac{4\Pi_{local} + \psi + 2\Pi_1}{3},\ if\ 5\Pi_{local} - 2\Pi_1 - \psi < 0 \end{cases} \tag{17}$$

Proof. For the four local cycles from S_3 to S_7, it takes $4\Pi_{local}$ time units. With $4\Pi_{local} \geq \Pi_1 > 2\Pi_{local}$, $4\Pi_{local} + \psi > \Pi_1$ holds. Thus, during the global cycle from S_7 to S_8, when the robot comes to PM_1, the robot does not need to wait before swapping at the PM, i.e., $\omega_{12} = 0$. Further, $\omega_{22} + \omega_{32} = \Pi_{local} - \psi$, resulting in that the time taken for the global cycle from S_7 to S_8 is $\psi + \omega_{12} + \omega_{22} + \omega_{32} = \Pi_{local}$. For the local cycle from S_8 to S_9, it takes Π_{local} time units. In the global cycle from S_9 to S_{10}, when the robot comes to PM_1, it has to wait there for $\omega_{13} = \Pi_1 - \Pi_{local} - \psi - (\omega_{22} + \omega_{32}) = \Pi_1 - 2\Pi_{local} > 0$ time units due to $4\Pi_{local} \geq \Pi_1 > 2\Pi_{local}$. Further, we have $\omega_{23} + \omega_{33} = max(\Pi_{local} - (\omega_{13} + \psi), 0) = max(\Pi_{local} - (\Pi_1 - 2\Pi_{local} + \psi), 0) = max(3\Pi_{local} - \Pi_1 - \psi, 0)$. Then, there are two cases as follows.

Case 1: $max(3\Pi_{local} - \Pi_1 - \psi, 0) = 3\Pi_{local} - \Pi_1 - \psi$. In this case, the time taken for the global cycle from S_9 to S_{10} is $\psi + \omega_{13} + \omega_{23} + \omega_{33} = \psi + \Pi_1 - 2\Pi_{local} + 3\Pi_{local} - \Pi_1 - \psi = \Pi_{local}$. Then, the time taken for the local cycle from S_1 to S_2 is Π_{local}. Furthermore, in the global cycle from S_2 to S_3, when the robot arrives at PM_1, it has to wait there for $\omega_{11} = \Pi_1 - \Pi_{local} - \psi - (\omega_{23} + \omega_{33}) = \Pi_1 - \Pi_{local} - \psi - (3\Pi_{local} - \Pi_1 - \psi) = 2\Pi_1 - 4\Pi_{local} > 0$ time units due to $4\Pi_{local} \geq \Pi_1 > 2\Pi_{local}$. Thus, $\omega_{21} + \omega_{31} = max(\Pi_{local} - (\omega_{11} + \psi), 0) = max(\Pi_{local} - (2\Pi_1 - 4\Pi_{local} + \psi), 0) = max(5\Pi_{local} - 2\Pi_1 - \psi, 0)$. Then, there are two cases as follows again. Case 1.1: $max(5\Pi_{local} - 2\Pi_1 - \psi, 0) = 5\Pi_{local} - 2\Pi_1 - \psi$. In this case, the time taken for the global cycle S_2 to S_3 is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \psi + 2\Pi_1 - 4\Pi_{local} + 5\Pi_{local} - 2\Pi_1 - \psi = \Pi_{local}$. Hence, the time taken for a period from S_1 to S_{10} is $9\Pi_{local}$ in this case. Case 1.2: $max(5\Pi_{local} - 2\Pi_1 - \psi, 0) = 0$. In this case, the time taken for the global cycle from S_2 to S_3 is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \psi + 2\Pi_1 - 4\Pi_{local}$. Hence, the time taken for a period from S_1 to S_{10} is $4\Pi_{local} + \psi + 2\Pi_1$ in this case.

Case 2: $max(3\Pi_{local} - \Pi_1 - \psi, 0) = 0$. In this case, the time taken for the global cycle from S_9 to S_{10} is $\psi + \omega_{13} + \omega_{23} + \omega_{33} = \psi + \Pi_1 - 2\Pi_{local}$. Then, the time taken for the local cycle from S_1 to S_2 is Π_{local}. Furthermore, in the global cycle from S_2 to S_3, when the robot arrives at PM_1, it has to wait there for $\omega_{11} = \Pi_1 - \Pi_{local} - \psi - (\omega_{23} + \omega_{33}) = \Pi_1 - \Pi_{local} - \psi > 0$ time units. Thus, $\omega_{21} + \omega_{31} = max(\Pi_{local} - (\omega_{11} + \psi), 0) = max(2\Pi_{local} - \Pi_1, 0) = 0$. Therefore, the time taken for the global cycle from S_2 to S_3 is $\psi + \omega_{11} + \omega_{21} + \omega_{31} = \Pi_1 - \Pi_{local}$. Hence, the time taken for a period from S_1 to S_{10} is $4\Pi_{local} + \psi + 2\Pi_1$ in this case.

In summary, we conclude that the theorem holds. □

Therefore, by an N3-WP2 schedule, we ensure that, in the case with $5\Pi_{local} - 2\Pi_1 - \psi \geq 0$, the cycle time is optimal. Further, by Theorem 12, the following result can be obtained.

Theorem 13. *For a DACT with (PM_1, $(PM_2, PM_3)^3$), by an N3-WP2 schedule, if $3\Pi_{local} + \psi \geq \Pi_1 > \Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, the cycle time is*

$$\Pi_{N3\text{-}WP2} = (4\Pi_{local} + 2\Pi_1 + \psi)/3 \tag{18}$$

Proof. For the four local cycles from S_3 to S_7, it takes $4\Pi_{local}$ time units. With $3\Pi_{local} + \psi \geq \Pi_1$, $4\Pi_{local} + \psi > \Pi_1$ holds. Thus, during the global cycle from S_7 to S_8, when the robot comes to PM_1, the robot does not need to wait before swapping at the PM, i.e., $\omega_{12} = 0$. Further, $\omega_{22} + \omega_{32} = 0$ due to $max\{\Pi_2, \Pi_3\} \leq \psi$. Therefore, the time taken for the global cycle from S_7 to S_8 is ψ. For the local cycle from S_8 to S_9, it takes Π_{local} time units. During the global cycle from S_9 to S_{10}, when the robot comes to PM_1, it has to wait there for $\omega_{13} = \Pi_1 - \Pi_{local} - \psi - (\omega_{22} + \omega_{32}) = \Pi_1 - \Pi_{local} - \psi > 0$ time units. Further, $\omega_{23} + \omega_{33} = 0$ due to $max\{\Pi_2, \Pi_3\} \leq \psi$. Thus, the time taken for the global cycle from S_9 to S_{10} is $\psi + \omega_{13} + \omega_{23} + \omega_{33} = \Pi_1 - \Pi_{local}$. Similarly, the time taken for the local cycle from S_1 to S_2 and the global cycle from S_2 to S_3 is Π_{local} and $\Pi_1 - \Pi_{local}$, respectively. Thus, the time taken for the period from S_1 to S_{10} is $4\Pi_{local} + \psi + 2\Pi_1$. Therefore, with three wafers being completed in such a period, the cycle time can be obtained by (18), or the theorem holds. □

Based on the above development, Algorithm 1 is presented to determine one of the schedules N3-WP1 and N3-WP2 to be applied to the system to maximize productivity.

Algorithm 1: For a DACT with (PM_1, $(PM_2, PM_3)^3$), the following algorithm is applied to choose one of the N3-WP1 and N3-WP2 schedules for the tool.

```
Input: $\rho_1$, $\rho_2$, $\rho_3$, $\alpha$, $\mu$, $\beta$, and $\lambda$
Output: The adopted schedule
1.   Calculate $\psi$, $\Pi_1$, $\Pi_2$, $\Pi_3$, and $\Pi_{local}$;
2.   If $max\{\Pi_2, \Pi_3\} \leq \psi$
3.      If $\Pi_1 \leq \Pi_{local} + \psi$
4.         The N3-WP2 schedule is applied, and the cycle time is calculated by Theorem 9;
5.      If $\Pi_{local} + \psi < \Pi_1 \leq 3\Pi_{local} + \psi$
6.         Calculate $\Pi_{N3\text{-}WP1}$ by Theorem 4 and $\Pi_{N3\text{-}WP2}$ by Theorem 13;
7.         If $\Pi_{N3\text{-}WP1} < \Pi_{N3\text{-}WP2}$
8.            The N3-WP1 schedule is applied;
9.         Else
10.           The N3-WP2 schedule is applied;
11.     If $\Pi_1 > 3\Pi_{local} + \psi$
12.        The N3-WP1 schedule is applied, and the cycle time is calculated by Theorem 8;
13.  Else
14.     If $\Pi_1 \leq \Pi_{local} + \psi$
15.        The N3-WP2 schedule is applied, and the cycle time is calculated by Theorem 10;
16.     If $\Pi_{local} + \psi < \Pi_1 \leq 2\Pi_{local}$
17.        The N3-WP2 schedule is applied, and the cycle time is calculated by Theorem 11;
18.     If $2\Pi_{local} < \Pi_1 \leq 2.5\Pi_{local} - 0.5\psi$
19.        The N3-WP2 schedule is applied, and the cycle time is calculated by Theorem 12;
20.     If $2.5\Pi_{local} - 0.5\psi < \Pi_1 \leq 3\Pi_{local} + \psi$
21.        Calculate $\Pi_{N3\text{-}WP1}$ by Theorem 5 and $\Pi_{N3\text{-}WP2}$ by Theorem 12;
22.        If $\Pi_{N3\text{-}WP1} < \Pi_{N3\text{-}WP2}$
23.           The N3-WP1 schedule is applied;
24.        Else
25.           The N3-WP2 schedule is applied;
26.     If $3\Pi_{local} + \psi < \Pi_1 \leq 4\Pi_{local}$
```

```
27.         Calculate $\Pi_{\text{N3-WP1}}$ by Theorem 6 and $\Pi_{\text{N3-WP2}}$ by Theorem 12.
28.         If $\Pi_{\text{N3-WP1}} < \Pi_{\text{N3-WP2}}$
29.             The N3-WP1 schedule is applied;
30.         Else
31.             The N3-WP2 schedule is applied;
32.     If $\Pi_1 > 4\Pi_{local}$
33.         The N3-WP1 schedule is applied, and the cycle time can be obtained by Theorem 7;
```

Note that, in the case with $4\Pi_{local} < \Pi_1$ and $max\{\Pi_2, \Pi_3\} > \psi$, by Theorem 7, an N3-WP1 schedule can achieve the minimum system cycle time, while in the case with $\Pi_1 > 3\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, by Theorem 8, an N3-WP1 schedule can achieve the minimum system cycle time as well. Thus, we do not present the cycle time analysis in both cases under an N3-WP2 schedule. Now, we have completed the cycle time analysis for a DACT with $(PM_1, (PM_2, PM_3)^3)$ by the N3-WP1 and N3-WP2 schedules.

Based on Theorems 10–12, in the case with $2.5\Pi_{local} - 0.5\psi \geq \Pi_1$ and $max\{\Pi_2, \Pi_3\} > \psi$, an N3-WP2 schedule is optimal. Based on Theorem 7, in the case with $\Pi_1 > 4\Pi_{local}$ and $max\{\Pi_2, \Pi_3\} > \psi$, an N3-WP1 schedule is optimal. Based on Theorem 9, in the case with $\Pi_1 \leq \Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, an N3-WP2 schedule is optimal. Based on Theorem 8, in the case with $\Pi_1 > 3\Pi_{local} + \psi$ and $max\{\Pi_2, \Pi_3\} \leq \psi$, an N3-WP1 schedule is optimal. In other cases, for the N3-WP1 and N3-WP2 schedule, the one with the minimum cycle time can be applied to a DACT with $(PM_1, (PM_2, PM_3)^3)$. This is what Algorithm 1 does. Besides, in such cases, the adopted scheduling method cannot ensure optimality in terms of the cycle time. This is also the limitation of this work.

Note that by extending the N3-WP1 and N3-WP2 schedules, similar schedules can be developed for a DACT with $(PM_1, (PM_2, PM_3)^k)$, $k \in \{3f \mid f \in \mathbb{N}\backslash\{1\}\}$. Furthermore, the cycle time can be analyzed in a similar way. Besides, according to the investigation from enterprises of integrated circuit high-end technological equipment, for DACTs with k-time reentrant processes, k is normally no greater than five. Therefore, the obtained results in this work match the practical demand well.

6. Implementation of the Proposed Methods and Illustrative Examples

6.1. Implementation of the Proposed Methods

For a DACT with $(PM_1, (PM_2, PM_3)^k)$, $k \geq 2$ and $k \neq 3f$, $f \in \{1, 2 \ldots\}$, based on Theorems 1 and 3, a 1-WP schedule exists for the tool such that the lower bound of the system cycle time can be achieved. Let W_0 denote a virtual wafer and assume that a tool starts from its idle state. To implement the 1-WP schedule for a DACT, virtual wafers are introduced into the tool, and we assume that at the initial state, each PM is processing a virtual wafer and also the robot is holding a wafer.

For the case that a DACT with $(PM_1, (PM_2, PM_3)^k)$, $k = 3f + 2$ and $f \in \mathbb{N} \cup \{0\}$, Theorem 1 provides a simple way to find a state (i.e., $\{W_2(1), W_1(2), W(2f + 3), R_3(W(4f + 5))\}$) starting from which a 1-WP schedule exists. By introducing virtual wafers into the tool, let $S_0 = \{W_0(1), W_0(2), W_0(3), R_3(W_0(5))\}$ be the initial (idle) state of the tool. Then, by the swap strategy, the tool can evolve into a state in which all virtual wafers are removed from the tool. At this time, the wafers in the tool are all real ones. In this way, the 1-WP schedule is implemented.

For the case that a DACT with $(PM_1, (PM_2, PM_3)^k)$, $k = 3f + 1$ and $f \in \mathbb{N}$, Theorem 3 provides a simple way to find a state (i.e., $\{W_2(1), W_1(2), W(4f + 3), R_3(W(2f + 3))\}$) starting from which a 1-WP schedule exists. By introducing virtual wafers into the tool, let $S_0 = \{W_0(1), W_0(2), W_0(7), R_3(W_0(5))\}$ be the initial (idle) state of the tool. Then, by the swap strategy, the tool can evolve into a state in which all virtual wafers are removed from the tool, and the wafers in the tool are all real ones. In this way, the 1-WP schedule is implemented.

For a DACT with $(PM_1, (PM_2, PM_3)^k)$, $k = 3f$ and $f \in \mathbb{N}$, based on Theorem 2, a 1-WP schedule does not exist. In this case, this work presents two methods (an N3-WP1 schedule and an N3-WP2 schedule) to operate the tool for productivity improvement. To implement the N3-WP1 schedule, by introducing virtual wafers, let $S_0 = \{W_0(1), W_0(2), W_0(5), R_3(W_0(5))\}$ be the initial (idle) state of the tool. Then, by the swap strategy, the tool can evolve into a state in which all virtual wafers are removed from the tool, and the wafers in the tool are all real ones. To implement the N3-WP2 schedule, by introducing virtual wafers, let $S_0 = \{W_0(1), W_0(2), W_0(3), R_3(W_0(7))\}$ be the initial (idle) state of the tool. Then, by the swap strategy, the tool can evolve into a state in which all virtual wafers are removed from the tool, and the wafers in the tool are all real ones. In this way, the N3-WP1 schedule and N3-WP2 schedule are implemented.

6.2. Illustrative Examples

Now, several examples are presented to demonstrate the obtained results in this work. Among them, Examples 2–4 come from [35]. Note that $\Pi_{\text{3-WP}}$ represents the system cycle time obtained by a 3-WP schedule.

Example 1. *For a DACT with (PM_1, (PM_2, PM_3)^5), the wafer processing time at PM_1, PM_2, and PM_3 is 80 s, 35 s, and 50 s (i.e., ρ_1 = 80 s, ρ_2 = 35 s, and ρ_3 = 50 s), respectively, and $\alpha = \mu = \beta$ = 3 s and λ = 8 s.*

For this example, $k = 3 \times 1 + 2$ with $f = 1$, Π_1 = 88 s, Π_2 = 43 s, Π_3 = 58 s, and ψ = 42 s. By applying a 3-WP schedule, $\Pi_{\text{3-WP}}$ = (290 + 44/3) s ≈ 304.67 s. However, if a 1-WP schedule is applied, it follows from Corollary 2 that $\Pi_{\text{1-WP}} = 5\Pi_3$ = 290 s. Obviously, the 1-WP schedule outperforms the 3-WP schedule.

Example 2. *For a DACT with (PM_1, (PM_2, PM_3)^3), ρ_1 = 37 s, ρ_2 = 22 s, ρ_3 = 32 s, $\alpha = \mu = \beta$ = 4 s, and λ = 8 s.*

For this example, Π_1 = 45 s, Π_2 = 30 s, Π_3 = 40 s, and ψ = 48 s. By using a 3-WP schedule, we get $\Pi_{\text{3-WP}}$ = 128 s. If an N3-WP1 schedule is applied, by Theorem 4, we get $\Pi_{\text{N3-WP1}}$ = 128 s, while if an N3-WP2 schedule is applied, by Theorem 9, we also get $\Pi_{\text{N3-WP2}}$ = 128 s. Thus, the three scheduling methods can obtain the same results.

Example 3. *For a DACT with (PM_1, (PM_2, PM_3)^3), ρ_1 = 50 s, ρ_2 = 22 s, ρ_3 = 32 s, $\alpha = \mu = \beta$ = 4 s, and λ = 8 s.*

For this example, Π_1 = 58 s, Π_2 = 30 s, Π_3 = 40 s, and ψ = 48 s. By using a 3-WP schedule, $\Pi_{\text{3-WP}}$ = 404/3 s ≈ 134.67 s. However, if an N3-WP1 schedule is applied, by Theorem 4, $\Pi_{\text{N3-WP1}}$ = 394/3 s ≈ 131.33 s. If an N3-WP2 schedule is applied, by Theorem 9, $\Pi_{\text{N3-WP2}}$ = 128 s. Thus, in this case, an N3-WP2 schedule should be applied since it can achieve the best performance among the three schedules.

Example 4. *For a DACT with (PM_1, (PM_2, PM_3)^3), ρ_1 = 450 s, ρ_2 = 200 s, ρ_3 = 250 s, $\alpha = \mu = \beta$ = 3 s, and λ = 8 s.*

For this example, Π_1 = 458 s, Π_2 = 208 s, Π_3 = 258 s, and ψ = 42 s. By using a 3-WP schedule, $\Pi_{\text{3-WP}}$ = (774 + 184/3) s ≈ 835.33 s. However, if an N3-WP1 schedule is applied, by Theorem 5, $\Pi_{\text{N3-WP1}}$ = 774 s. If an N3-WP2 schedule is applied, by Theorem 11, $\Pi_{\text{N3-WP2}}$ = 774 s. Thus, in this case, we can choose either N3-WP1 or N3-WP2 to improve the cycle time by 7.34% in comparison with a 3-WP schedule.

Example 5. *For a DACT with (PM_1, (PM_2, PM_3)^3), ρ_1 = 200 s, ρ_2 = 45 s, ρ_3 = 50 s, $\alpha = \mu = \beta$ = 2 s, and λ = 5 s.*

In this example, Π_1 = 205 s, Π_2 = 50 s, Π_3 = 55 s, and ψ = 27 s. By using a 3-WP schedule, we have $\Pi_{\text{3-WP}}$ = (165 + 272/3) s ≈ 255.67 s. However, if an N3-WP1 schedule is applied, by Theorem 6, $\Pi_{\text{N3-WP1}}$ = 617/3 s ≈ 205.67 s. If an N3-WP2 schedule is applied, by

Theorem 12, $\Pi_{\text{N3-WP2}}$ = 219 s. Thus, in this case, the N3-WP1 schedule should be applied, and it improves the cycle time by 19.56% in comparison with a 3-WP schedule.

In Examples 1–5, with the parameters of the cluster tool provided, we can quickly obtain the cycle time of the system under any one of the 3-WP schedule, N3-WP1 schedule, N3-WP2 schedule, and 1-WP schedule (if existing). With the system cycle time obtained, one can determine the best scheduling method to be applied to DACTs with reentrant wafer flows.

Besides, more examples are given in Table 3 to show the superiority of the proposed methods by comparing them with the 3-WP schedule in terms of the system cycle time. In the cases in Table 3, the time for the robot activities and wafer processing is set according to real applications in a semiconductor equipment vendor in China. With these given parameters of cluster tools, Algorithm 1 can be used to obtain the best scheduling method immediately in the cases shown in Table 3. Furthermore, by using the best one of the N3-WP1 and N3-WP2 schedules, the cycle time of the system is improved by 16.8% on average by comparing with the 3-WP schedule. Besides, in Cases 1–6, the adopted scheduling method can achieve a minimal cycle time. However, in Cases 7–11, the optimality of the adopted scheduling method cannot be ensured. Nevertheless, the adopted method significantly outperforms the existing 3-WP schedule.

Table 3. Comparison results.

No.	ρ_1	ρ_2	ρ_3	N3-WP1		N3-WP2		3-WP	The Adopted Scheduling Method	Improvement
				$\Pi_{\text{N3-WP1}}$	Theorem	$\Pi_{\text{N3-WP2}}$	Theorem	$\Pi_{\text{3-WP}}$		
1	250	35	50	**258**	7	/	/	302	N3-WP1	14.57%
2	150	25	30	**158**	8	/	/	195(1/3)	N3-WP1	19.11%
3	70	25	30	130	4	**118**	9	142	N3-WP2	16.90%
4	70	25	35	140(1/3)	5	**129**	10	152	N3-WP2	15.13%
5	95	40	50	183(2/3)	5	**174**	11	198(2/3)	N3-WP2	12.42%
6	110	40	50	188(2/3)	5	**174**	12	208(2/3)	N3-WP2	16.61%
7	140	25	30	**153(1/3)**	4	163(1/3)	13	188(2/3)	N3-WP1	18.73%
8	100	25	30	140	4	**136(2/3)**	13	162	N3-WP2	15.64%
9	210	35	50	**222**	6	236(2/3)	12	275(1/3)	N3-WP1	19.37%
10	200	35	50	**218(2/3)**	5	230	12	268(2/3)	N3-WP1	18.61%
11	120	35	50	192	5	**176(2/3)**	12	215(1/3)	N3-WP2	17.96%

7. Conclusions

Wafer reentrant flows normally exist for some processes, such as chemical vapor and physical vapor deposition. Moreover, cluster tools are widely applied to wafer fabrication for such processes. Such a tool compactly integrates several PMs, a wafer transport robot, and two LLs. It can provide a highly precise vacuum environment for wafer fabrication. Since there is no buffer between PMs, it is important to effectively schedule such a tool with reentrant processes. For a DACT with two-time reentrant processes, it is found that a 1-WP schedule can achieve the optimal cycle time. However, for some wafer fabrication processes, wafers should be processed at some processing steps more than two times. Up to now, the problem is open for the issue if a 1-WP schedule exists for a DACT with wafer reentrant processes for more than two times. If not, it raises the question of whether a better schedule exists by comparing it with an existing 3-WP schedule. This motivates us to do this work to answer this question.

In this work, theorical analysis is conducted on it. In Section 4, it is shown that if there exists a value $f \in \mathbb{N}$ such that $k = 3f$ holds, then a 1-WP schedule does not exist, and otherwise, the system can be scheduled by a 1-WP schedule. Further, analytical expressions are given to calculate the system cycle time if a 1-WP schedule exists. In the cases where a 1-WP schedule is not applicable, this work presents two novel scheduling methods in Section 5 to improve the cycle time by comparing it with an existing 3-WP schedule. Meanwhile, by these two scheduling methods, analytical expressions are given to obtain the system cycle time. Further, an algorithm is given to display the conditions under which one of the two novel scheduling methods can be used to operate cluster tools. Besides, this work presents the conditions under which the two novel scheduling methods may not achieve the minimal cycle time, i.e., the limitation of the methods. Thus, given a case, by simply calculating the cycle time, one can decide which method should be applied to the system. In Section 6, by introducing virtual wafers, a way to implement the proposed scheduling methods is given, and the application of the proposed scheduling methods is demonstrated by examples.

In the future, we aim to deal with the scheduling problems of cluster tools handling multi-wafer types with different wafer flow patterns as well as multi-cluster tools. Moreover, the proposed scheduling methods cannot achieve the minimal system cycle time in some cases. Thus, another future work needs to be done for optimal scheduling of such cases.

Author Contributions: Conceptualization, T.S. and Y.Q.; methodology, T.S. and Y.Q.; validation, T.S. and Y.H.; formal analysis, T.S., Y.Q. and Y.H.; investigation, T.S. and Y.Q.; resources, N.W. and B.L.; data curation, T.S., Y.H. and B.L.; writing—original draft preparation, T.S. and Y.Q.; writing—review and editing, N.W.; supervision, N.W. and Z.L.; project administration, Y.Q. and N.W.; funding acquisition, N.W. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by Science and Technology development fund (FDCT), Macau SAR (file Nos. 0018/2021/A1, 0083/2021/A2, and 0015/2020/AMJ), and in part by Research Fund of Guangdong-Hong Kong-Macao Joint Laboratory for Intelligent Micro-Nano Optoelectronic Technology (No. 2020B1212030010).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kim, W.; Yu, T.-S.; Lee, T.-E. Integrated Scheduling of a Dual-Armed Cluster Tool for Maximizing Steady Schedule Patterns. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 7282–7294. [CrossRef]
2. Kim, Y.; Lee, G.; Jeong, J. ML-Based JIT1 Optimization for Throughput Maximization in Cluster Tool Automation. *Appl. Sci.* **2022**, *12*, 7519. [CrossRef]
3. Lee, H.Y.; Lee, T.E. Scheduling single-armed cluster tools with reentrant wafer flows. *IEEE Trans. Semicond. Manuf.* **2006**, *19*, 226–240. [CrossRef]
4. Bleakie, A.; Djurdjanovic, D. Feature extraction, condition monitoring, and fault modeling in semiconductor manufacturing systems. *Comput. Ind.* **2013**, *64*, 203–213. [CrossRef]
5. Qiao, Y.; Wu, N.; Zhou, M. A Petri Net-Based Novel Scheduling Approach and Its Cycle Time Analysis for Dual-Arm Cluster Tools with Wafer Revisiting. *IEEE Trans. Semicond. Manuf.* **2013**, *26*, 100–110. [CrossRef]
6. Lopez, M.J.; Wood, S.C. Systems of multiple cluster tools: Configuration, reliability, and performance. *IEEE Trans. Semicond. Manuf.* **2003**, *16*, 170–178. [CrossRef]
7. Lee, T.E.; Lee, H.Y.; Shin, Y.H. Workload balancing and scheduling of a single-amred cluster tool. In Proceedings of the 5th APIEMS Conference, Gold Coast, Australia, 12–15 December 2004; pp. 1–15.
8. Venkatesh, S.; Davenport, R.; Foxhoven, P.; Nulman, J. A steady-state throughput analysis of cluster tools: Dual-blade versus single-blade robots. *IEEE Trans. Semicond. Manuf.* **1997**, *10*, 418–424. [CrossRef]
9. Perkinson, T.L.; McLarty, P.K.; Gyurcsik, R.S.; Cavin, R.K. Single-wafer cluster tool performance: An analysis of throughput. *IEEE Trans. Semicond. Manuf.* **1994**, *7*, 369–373. [CrossRef]
10. Kim, D.K.; Kim, H.J.; Lee, T.E. Optimal scheduling for sequentially connected cluster tools with dual-armed robots and a single input and output module. *Int. J. Prod. Res.* **2017**, *55*, 3092–3109. [CrossRef]
11. Lee, J.H.; Kim, H.J.; Lee, T.E. Scheduling cluster tools for concurrent processing of two wafer types with PM sharing. *Int. J. Prod. Res.* **2015**, *53*, 6007–6022. [CrossRef]
12. Rostami, S.; Hamidzadeh, B.; Camporese, D. An optimal periodic scheduler for dual-arm robots in cluster tools with residency constraints. *IEEE Trans. Robot. Autom.* **2001**, *17*, 609–618. [CrossRef]
13. Kim, J.H.; Lee, T.E.; Lee, H.Y.; Park, D.B. Scheduling analysis of time-constrained dual-armed cluster tools. *IEEE Trans. Semicond. Manuf.* **2003**, *16*, 521–534. [CrossRef]

14. Jung, C.; Lee, T.E. An Efficient Mixed Integer Programming Model Based on Timed Petri Nets for Diverse Complex Cluster Tool Scheduling Problems. *IEEE Trans. Semicond. Manuf.* **2012**, *25*, 186–199. [CrossRef]
15. Wu, N.; Chu, C.; Chu, F.; Zhou, M.C. A Petri Net Method for Schedulability and Scheduling Problems in Single-Arm Cluster Tools with Wafer Residency Time Constraints. *IEEE Trans. Semicond. Manuf.* **2008**, *21*, 224–237. [CrossRef]
16. Wu, N.; Zhou, M. 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.* **2010**, *7*, 303–315. [CrossRef]
17. Ko, S.G.; Yu, T.S.; Lee, T.E. Wafer delay analysis and workload balancing of parallel chambers for dual-armed cluster tools with multiple wafer types. *IEEE Trans. Autom. Sci. Eng.* **2021**, *18*, 1516–1526. [CrossRef]
18. Wang, J.F.; Liu, C.F.; Zhou, M.C.; Leng, T.T.; Albeshri, A. Optimal cyclic scheduling of wafer-residency-time-constrained dual-arm cluster tools by configuring processing modules and robot waiting time. *IEEE Trans. Semicond. Manuf.* **2023**, *36*, 251–259. [CrossRef]
19. Wu, N.; Zhou, M. 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.* **2012**, *9*, 446–454. [CrossRef]
20. Wu, N.; Zhou, M. Schedulability Analysis and Optimal Scheduling of Dual-Arm Cluster Tools with Residency Time Constraint and Activity Time Variation. *IEEE Trans. Autom. Sci. Eng.* **2012**, *9*, 203–209. [CrossRef]
21. Qiao, Y.; Wu, N.; Zhou, M. 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.* **2012**, *25*, 432–446. [CrossRef]
22. Qiao, Y.; Wu, N.; Zhou, M. Real-Time Scheduling of Single-Arm Cluster Tools Subject to Residency Time Constraints and Bounded Activity Time Variation. *IEEE Trans. Autom. Sci. Eng.* **2012**, *9*, 564–577. [CrossRef]
23. Lim, Y.; Yu, T.S.; Lee, T.E. Adaptive scheduling of cluster tools with wafer delay constraints and process time variation. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 375–388. [CrossRef]
24. Kim, C.; Yu, T.S.; Lee, T.E. Feedback control of cluster tools: Stability against random time disruptions. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 2008–2015. [CrossRef]
25. Yu, T.S.; Kim, H.J.; Lee, T.E. Scheduling Single-Armed Cluster Tools with Chamber Cleaning Operations. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 705–716. [CrossRef]
26. Yu, T.S.; Lee, T.E. Scheduling Dual-Armed Cluster Tools with Chamber Cleaning Operations. *IEEE Trans. Autom. Sci. Eng.* **2019**, *16*, 218–228. [CrossRef]
27. Yu, T.S.; Lee, T.E. Wafer delay analysis and control of dual-armed cluster tools with chamber cleaning operations. *Int. J. Prod. Res.* **2020**, *58*, 434–447. [CrossRef]
28. Qiao, Y.; Lu, Y.; Li, J.; Zhang, S.; Wu, N.; Liu, B. An Efficient Binary Integer Programming Model for Residency Time-Constrained Cluster Tools with Chamber Cleaning Requirements. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 1757–1771. [CrossRef]
29. Li, C.; Yang, F.; Zhen, L. Efficient scheduling approaches to time-constrained single-armed cluster tools with condition-based chamber cleaning operations. *Int. J. Prod. Res.* **2022**, *60*, 3555–3568. [CrossRef]
30. Li, H.P.; Zhu, Q.H.; Hou, Y. Scheduling a single-arm robotic cluster tool with a condition-based cleaning operation. In Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics, Clarion Congress Hotel, Prague, Czech Republic, 9–12 October 2022.
31. Zuberek, W.M. Cluster tools with chamber revisiting-modeling and analysis using timed Petri nets. *IEEE Trans. Semicond. Manuf.* **2004**, *17*, 333–344. [CrossRef]
32. Wu, N.; Chu, F.; Chu, C.; Zhou, M. Petri Net-Based Scheduling of Single-Arm Cluster Tools with Reentrant Atomic Layer Deposition Processes. *IEEE Trans. Autom. Sci. Eng.* **2011**, *8*, 42–55. [CrossRef]
33. Wu, N.; Chu, F.; Chu, C.; Zhou, M. Petri Net Modeling and Cycle-Time Analysis of Dual-Arm Cluster Tools with Wafer Revisiting. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 196–207. [CrossRef]
34. Wu, N.; Zhou, M.; Chu, F.; Chu, C. A Petri-Net-Based Scheduling Strategy for Dual-Arm Cluster Tools with Wafer Revisiting. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 1182–1194. [CrossRef]
35. Qiao, Y.; Wu, N.; Zhu, Q.; Bai, L. Cycle time analysis of dual-arm cluster tools for wafer fabrication processes with multiple wafer revisiting times. *Comput. Oper. Res.* **2015**, *53*, 252–260. [CrossRef]
36. Qiao, Y.; Wu, N.; Zhou, M. Scheduling of Dual-Arm Cluster Tools with Wafer Revisiting and Residency Time Constraints. *IEEE Trans. Ind. Inform.* **2014**, *10*, 286–300. [CrossRef]
37. Qiao, Y.; Wu, N.; Zhou, M. 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.* **2015**, *45*, 472–484. [CrossRef]
38. Qiao, Y.; Wu, N.; Yang, F.; Zhou, M.; Zhu, Q. 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.* **2018**, *48*, 622–636. [CrossRef]
39. Qiao, Y.; Wu, N.; Yang, F.; Zhou, M.; Zhu, Q.; Qu, T. Robust Scheduling of Time-Constrained Dual-Arm Cluster Tools with Wafer Revisiting and Activity Time Disturbance. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *49*, 1228–1240. [CrossRef]