<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Efficient Approach to Cyclic Scheduling of Single-arm Cluster Tools with Chamber Cleaning Operations and Wafer Residency Time Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Yang, Fa Jun; Wu, Nai Qi; Gao, Kai Zhou; Zhang, Chun Jiang; Zhu, Yu Ting; Su, Rong.; Qiao, Yan</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2018</td>
</tr>
<tr>
<td><strong>URL</strong></td>
<td><a href="http://hdl.handle.net/10220/44649">http://hdl.handle.net/10220/44649</a></td>
</tr>
<tr>
<td><strong>Rights</strong></td>
<td>© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: [<a href="http://dx.doi.org/10.1109/TSM.2018.2811125">http://dx.doi.org/10.1109/TSM.2018.2811125</a>].</td>
</tr>
</tbody>
</table>
Efficient Approach to Cyclic Scheduling of Single-arm Cluster Tools with Chamber Cleaning Operations and Wafer Residency Time Constraint

FaJun Yang, NaiQi Wu, Senior Member, IEEE, KaiZhou Gao, Member, IEEE, ChunJiang Zhang, YuTing Zhu, Rong Su, Senior Member, IEEE, Yan Qiao, Member, IEEE

Abstract—In semiconductor manufacturing, with the shrinking down of wafer circuit widths, a strict quality control is required for wafer fabrication processes, resulting in that after a wafer being processed and removed from a chamber, a cleaning operation that takes significant time is performed for eliminating the chemical residual. Such a cleaning operation makes a traditional backward strategy for single-arm cluster tools inefficient. By the existing studies, it is shown that the productivity can be improved if some numbers of chambers at a step are kept empty. With this idea, an extended backward strategy is proposed by deciding the optimal number of empty chambers. Based on such a strategy, this work studies the challenging problem for scheduling a single-arm cluster tool with both chamber cleaning operations and wafer residency time constraint for the first time. By building a timed Petri net model for the system, two linear programs are proposed to determine the minimal cycle time and test the existence of a feasible schedule and find it if existing. At last, two industrial examples are used to demonstrate the obtained results.

Index Terms—Cluster tools, wafer manufacturing, Petri nets (PN), cleaning, scheduling.

I. INTRODUCTION

Cluster tools are widely adopted for wafer processing in semiconductor fabrication. Generally, a cluster tool is composed of a transfer chamber (also called a robot), two loadlocks, and several process chambers (PCs). Depending on the number of the arms for the robot, a tool is named single- or dual-arm cluster tool whose architecture is shown in Fig. 1.

Fig. 1. (a) single-arm cluster tool and (b) dual-arm cluster tool.

Wafers to be processed enter a tool through the loadlocks in a cassette-by-cassette way with a cassette holding 25 wafers. Then, they are transported to the process chambers by the robot one by one for processing in a predefined sequence. After visiting all the processing steps, a wafer returns to the cassette eventually [27].

In comparison with the scheduling problem of complex industrial processes [4, 23, 30, 31, 37], it seems that a cluster tool is simple. However, various process constraints make its scheduling problem very complicated. Thus, extensive studies on cluster tool scheduling have been done [1, 7-18, 20, 21, 27-29, 32-35] and a review on this subject can be found in [16]. Some wafer manufacturing processes require that a processed wafer should be unloaded from a chamber in a limited time due to that the residual chemical gases and high temperature could damage its surface [7, 12, 18]. Such a requirement results in wafer residency time constraint (WRT constraint) and complicates the scheduling problem of cluster tools. Moreover, the activity time variation further complicates its scheduling problem [28, 29].

For dual-arm cluster tools with WRT constraint, Rostami et al. [18] use heuristic algorithms to search an optimal schedule without analyzing the existence of a feasible schedule. Kim et al. [7] develop sufficient conditions to check if there is a feasible schedule and find an optimal schedule if it exists.

For both dual- and single-arm cluster tools with WRT constraint, necessary and sufficient conditions are developed...
in [12] to test if there is a feasible schedule. Also, they present algorithms to find it. However, it involves complex computation. More efficient ways are proposed in [20, 27] to find a feasible and optimal cyclic schedule. Also, efficient methods are developed to deal with activity time variation [28, 29, 41, 42].

In [20], it is assumed that, for a single-arm cluster tool, the wafer processing time at a chamber is much longer than robot task time and thus a backward strategy is optimal. This may not be true for some cases [2]. Thus, in [35], with the WRT constraint being considered, computationally efficient techniques are derived to obtain a feasible and optimal cyclic schedule for single-arm cluster tools whose scheduling strategies are not limited to the backward one. For hybrid and single-arm multi-cluster tools with WRT constraint, Yang et al. [32] and Zhu et al. [39] establish necessary and sufficient conditions for the existence of a feasible schedule and propose algorithms to find an optimal one if a feasible solution exists.

Recently, to accommodate diversified customer needs, the wafer lot size is becoming smaller and smaller, leading to frequent lot switching operations [8, 13, 14], and start-up and close-down processes of tools [9, 10, 17, 40].

As the wafer circuit line width continuously shrinks down, a stricter quality control is required for wafer fabrication processes. To do so, a chamber cleaning operation is needed after a chamber finishes processing m wafers. It is called “purge” if \( m = 1 \) [11, 15]. It is known that, in a leading fab, purge is adopted for 50%-80% of the tools that are used for etching and chemical/physical vapor deposition. The purge takes time from 30-300 seconds [36].

In [36], by keeping some numbers of chambers being idle, the authors propose an extended backward strategy. In comparison with the traditionally full loading backward strategy (i.e., each chamber is processing a wafer), such an extended backward strategy uses partial loading, i.e., some numbers of chambers in a step are kept empty. The authors in [36] proved that, with the extended backward strategy, the globally optimal cycle time can be obtained in most practical cases and, even if it is not globally optimal, the gap between the lower bound of the global cycle time and the cycle time obtained by their proposed method is not too big (about 10%). Furthermore, the extended backward strategy is quite simple and very close to the traditional backward strategy, which is prevalently used in practice. Hence, in this work, we adopt this extended backward strategy.

In practice, due to that the chamber cleaning operations could significantly increase the wafer sojourn time delays in a chamber such that it is challenging to make the WRT constraint satisfied. Motivated by this, based on the extended backward strategy, this work conducts a study on scheduling a single-arm cluster tool with both chamber cleaning operations and WRT constraint for the first time. Consequently, an efficient method is presented to test if a feasible schedule exists and find it if existing.

In comparison with the prior work, this work has the following primary contributions:

1) A linear program is formulated to obtain the minimal system cycle time; and

2) Based on the minimal cycle time, we derive an efficient linear program to test if there is a feasible schedule and find it if existing.

In Section II, a timed PN model is developed and, based on the model, Section III proposes two linear programs to calculate the system cycle time and check whether a feasible schedule exists and find it if existing. Examples are given to demonstrate the power of the derived approach in Section IV. Finally, conclusions are summarized in Section V.

II. SYSTEM MODELING

Assume that, except the loadlocks, there are \( n \) processing steps for a single-arm cluster tool to process wafers. Then, we let \( \Omega_n = \{0, 1, 2, \ldots, n\} \) and \( \mathbf{N}_n = \Omega_n \{0\} \). For Step \( i, \) \( i \in \mathbf{N}_n, \) \( m_i \) is used to denote the number of its parallel PCs. By following Yu et al. [36], the number of empty chambers at Step \( i, i \in \mathbf{N}_n, \) is denoted by \( z_i \). Thus, at Step \( i, \) at most \( m_i - z_i \) chambers are processing wafers.

In [36], a method is proposed to determine \( z_i, i \in \mathbf{N}_n, \) and based on the determined \( z_i, \) an extended backward strategy is adopted.

To explain the meaning of the extended backward strategy, we first present the traditional backward strategy that is determined by the robot task sequence: \( \langle \) unloads a finished wafer from the \( m_i \)th chamber at Step \( n \rightarrow \) goes to the loadlocks and drops it there \( \rightarrow \) goes to Step \( n-1 \) and unloads a completed wafer from its \( m_i \)th chamber \( \rightarrow \) moves to Step \( n \) and drops it into the \( m_i \)th chamber \( \rightarrow \) goes to Step \( n-2 \) and picks up a completed wafer from its \( m_i \)th chamber \( \rightarrow \ldots \rightarrow \) goes to Step 1 and drops a wafer into the \( m_i \)th chamber \( \rightarrow \) goes to Step \( n \) again and picks up a finished wafer from its \( (m_i-1) \)th chamber \( \ldots \rangle. \) Note that, for such a traditional backward strategy, after removing a processed wafer from the \( m_i \)th chamber at Step \( i, i \in \mathbf{N}_n, 1 \leq m_i \leq m, \) the robot loads another wafer into the \( m_i \)th chamber within one cycle, and in the next cycle, it unloads a wafer from the \( (m_i-1) \)th (or \( m_i \)th if \( m_i = 1 \)) chamber at Step \( i. \) However, for the extended backward strategy proposed in [36], after unloading a wafer from the \( m_i \)th chamber at Step \( i, \) the robot loads another wafer into the earliest emptied chamber at Step \( i \) rather than the \( m_i \)th chamber, unless \( m_i = 1. \) This is the difference between them.

With the above assumption, this work intends to answer whether there is a feasible schedule with WRT constraint and how to find it if existing.

A. Finite Capacity Petri Net

For modeling, analysis, and control of manufacturing systems, PNs are widely used as an efficient approach [5-7, 20-22]. Based on the work in [19, 20, 24-26, 38], a PN is defined as follows.

Definition 1: A PN is a directed graph with six sets of elements defined as follows.

1) \( P \): a finite set of places.

2) \( T \): a finite set of transitions, which is disjoint with \( P. \)

3) \( I: (P \times T) \rightarrow \mathbf{N} = \{0, 1, 2, \ldots\} \) is an input function. If \( \ell(p, i) > 0, \) there is a directed arc from \( p \) to \( t \) with weight \( \ell(p, i), \)
Step 0

4) \( O: (P \times T) \rightarrow \mathbb{N} \) is an output function. If \( O(p, t) > 0 \), there is a directed arc from \( t \) to \( p \) with weight \( O(p, t) \); if \( O(p, t) = 0 \), there is no such an arc.

5) \( M(P) = (M(p_1), \ldots, M(p_n))^T \) is a marking with \( M_i \) being the initial one, where \( M(p_i) \) denotes the number of tokens in \( p_i \).

6) \( K: \) a capacity function with \( K(p) \) representing the maximum number of tokens that \( p \) can hold.

For a transition \( t \), its preset is denoted by \( \pi_t = \{ p: p \in P \text{ and } I(p, t) > 0 \} \); ant its postset is represented by \( \pi^*_t = \{ p: p \in P \text{ and } O(p, t) > 0 \} \). Then, we can give the following definition [19, 24, 26].

**Definition 2:** In a finite capacity PN, a transition \( t \) is said to be enabled if \( \forall p \in \pi_p \),

\[
M(p) \geq I(p, t) \tag{2.1}
\]

and

\[
K(p) \geq M(p) - I(p, t) + O(p, t) \tag{2.2}
\]

When an enabled \( t \) fires at \( M \), one can obtain a new marking

\[
M'(p) = M(p) - I(p, t) + O(p, t) \tag{2.3}
\]

By this definition, to enable \( t \), we need enough tokens in \( \forall p \in \pi^*_t \) and enough free spaces in \( \forall p \in \pi_t \). Next, PN is used to model the system.

### B. Petri Net for the System

As there are \( n \) processing steps (except the loadlocks), we number them as Steps 1 to \( n \). For concise presentation, the loadlocks are seen as Step 0 and \( n + 1 \). By adopting the extended backward strategy, at Step \( i \), \( i \in \mathbb{N}_n \), there are \( z_i \) empty parallel chambers. Thus, at a time, at most \( z_i + 1 \) parallel chambers are in the state of cleaning (it occurs for a time duration from the time instant when a wafer is removed from Step \( i \) to the time instant when another one is loaded into Step \( i \)) and, at most \( e_i = m_i - z_i \) wafers are being processed. With such a fact, the PN of the model is as follows.

![Fig. 2. PN model for system with \( n \) processing steps.](image)

Place \( p_i \) with \( K(p_i) = e_i = m_i - z_i \) models the PCs that are processing wafers at Step \( i \), \( i \in \mathbb{N}_n \), and \( p_0 \) the loadlocks with \( K(p_0) = \infty \). Places \( q_{i1} \) and \( q_{i2} \) with \( K(q_{i1}) = K(q_{i2}) = 1 \) model that the robot waits there before loading/unloading a wafer into/from \( p_i, i \in \mathbb{N}_n \). Place \( c_i \) with \( K(c_i) = z_i + 1 \) models the chambers that are being cleaned at Step \( i, i \in \mathbb{N}_n \). As chamber cleaning operation is not needed in the loadlocks, \( c_0 \) does not exist. Pictorially, a circle is used to denote these places. A special place \( r \) denoted by an ellipse is added to model the robot and we have \( K(r) = 1 \), implying that the robot has a single arm.

Transition \( l_i \) models the loading activity of the robot at \( p_i, i \in \mathbb{N}_n \). The robot task of unloading a wafer from a chamber at \( p_i, i \in \mathbb{N}_n \), and moving to Step \( i + 1 \) is modeled by transition \( u_i \). With no wafer being carried, the robot moving from Steps \( i + 2 \) to \( i \) is modeled by transition \( y_i, i \in \mathbb{N}_n \). From Steps 0 to \( n - 1 \) by \( y_{n-1} \), and from Steps 1 to \( n \) by \( y_n \). Pictorially, a bar is used to denote these transitions.

As no wafer can be dropped into a chamber before the chamber finishes its cleaning operation, and after a wafer is removed from a chamber its cleaning operation starts immediately. Thus, arcs \((c_i, l_i), (l_i, p_i), (p_i, u_i), (u_i, c_i)\), \( i \in \mathbb{N}_n \), are added. Note that if \( i = 0 \), \((c_0, l_0)\) and \((u_0, c_0)\) does not exist. Then, by adding arcs \((q_{i1}, l_i), (l_i, r), (r, y_i), (y_i, q_{i2}), (q_{i2}, u_i), (u_i, q_{i+1})\), \( i \in \mathbb{N}_n \), we can obtain the PN structure for the system as shown in Fig. 2.

For the above obtained PN structure, we set its initial marking \( M_0 \) as follows: \( M_0(q_{i1}) = M_0(q_{i2}) = 0, i \in \mathbb{N}_n \); \( M_0(p_i) = e_i = m_i - z_i, i \in \mathbb{N}_n \), and \( M_0(p_0) = m \) to model the fact that, in the loadlocks, there are always wafers to be processed; \( M_0(c_i) = z_i, i \in \mathbb{N}_n \); and \( M_0(r) = 1 \), as indicated by the number in the places shown in Fig. 2. By doing so, the PN model of the system is finally obtained.

With the PN model being built, we need to make sure that when the extended backward strategy is applied, the PN should be free of deadlock. To do so, the following control policy is presented.

**Definition 3:** For the above developed PN, at marking \( M \), if \( M(p_i) = m_i - z_i, i \in \mathbb{N}_n \), then transition \( y_n \) is control-enabled; and if \( M(p_{n+1}) = m_n - z_n - 1, i \in \mathbb{N}_n \), then \( y_n \) is control-enabled.

By Definition 3, at \( M_0 \) only \( y_n \) is enabled, meanwhile, Conditions (2.1) and (2.2) are satisfied for \( y_n \). Thus, it can fire and followed by \( u_n \) and then \( l_n \), leading to marking \( M_1 \) such that only \( y_{n-1} \) is enabled due to that, after the firing of \( u_n \), we have \( M(p_n) = m_n - z_n - 1 \). As Conditions (2.1) and (2.2) are also satisfied for \( y_{n-1} \), it can fire and followed by \( u_{n-1} \) and then \( l_{n-1} \). Next, \( y_{n-2} \) can fire. Consequently, after some time, \( y_0 \) can fire and followed by \( u_0 \) and then \( l_0 \). Then, marking \( M_n \) that is equivalent to \( M_0 \) is obtained, implying that a cycle is completed without deadlock, or the obtained PN model is deadlock-free.

### C. Modeling Activity Time

To describe the temporal behavior of the system, both transitions and places are associated with time as follows since both of them model activities that take time. If a transition \( t \) is associated with time \( \theta \), firing \( t \) takes \( \theta \) time units, while if a place \( p \) is associated with time \( \theta \), it implies that before a token in \( p \) can enable its output transition, it should stay in \( p \) for at least \( \theta \) time units. By following [3, 7], we have:

1) The time for loading/unloading a wafer at a step is a constant and denoted by \( \lambda \); and
2) The time for robot moving between two steps with or without holding a wafer is a known constant and denoted by \( \mu \).

For Step \( i, i \in \mathbb{N}_n \), let \( \alpha_i \) and \( \omega_i \) denote the wafer processing and chamber cleaning time, \( \delta \) the longest time for which a processed wafer can stay in Step \( i \), and \( \tau_i \) the wafer residency time (WRT) at Step \( i \).

Further, let \( \omega_{i1} \) and \( \omega_{i2} \), \( i \in \mathbb{N}_e \), represent the robot waiting time in \( q_{i1} \) and \( q_{i2} \), \( i \in \mathbb{N}_e \). In Table I, we summarize the time taken for different transitions and places.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Transition or place</th>
<th>Action</th>
<th>Time duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>( t_i \in T )</td>
<td>Loading a wafer into Step ( i, i \in \mathbb{N}_n )</td>
<td>( \lambda )</td>
</tr>
<tr>
<td>( \mu )</td>
<td>( y_i \in T )</td>
<td>Moving from Steps ( i + 2 ) to ( i, i \in \mathbb{N}_e )</td>
<td>( \mu )</td>
</tr>
<tr>
<td>( \tau_i )</td>
<td>( p_i \in P )</td>
<td>A wafer stays in a chamber of ( p_i, i \in \mathbb{N}_n )</td>
<td>( [\alpha_i, \alpha_i + \delta] )</td>
</tr>
<tr>
<td>( \alpha_0 )</td>
<td>( p_i \in P )</td>
<td>A chamber at Step ( i, i \in \mathbb{N}_n ) is being cleaned</td>
<td>( \alpha_i )</td>
</tr>
<tr>
<td>( \alpha_{i1} )</td>
<td>( q_{i1} \in P )</td>
<td>Waiting before loading a wafer into Step ( i, i \in \mathbb{N}_e )</td>
<td>( \alpha_{i1} )</td>
</tr>
<tr>
<td>( \alpha_{i2} )</td>
<td>( q_{i2} \in P )</td>
<td>Waiting before unloading a wafer from Step ( i, i \in \mathbb{N}_e )</td>
<td>( \alpha_{i2} )</td>
</tr>
</tbody>
</table>

Remark 1: Due to customization, a cluster tool may need to process different wafer types with different flow patterns. Note that the PN model developed in this work is structurally independent of wafer flow patterns. Thus, when there is a wafer type switch, what we need to do is changing the initial data without holding a wafer is a known constant and denoted by \( \delta \).

As pointed out in the above analysis, the main difference between the extended backward strategy given in [36] and the traditional one is stated as follows. By the traditional one, after unloading a wafer from the \( m_i \)th chamber at Step \( i \), \( i \in \mathbb{N}_n \), \( 1 \leq m_i \leq m_n \), the robot loads another wafer into this chamber. However, by the extended one, after a wafer is unloaded from the \( m_i \)th chamber at Step \( i \), the robot loads another wafer into the earliest emptied chamber at Step \( i \) rather than the \( m_i \)th chamber, unless \( m_i = 1 \). Hence, for both strategies, the robot cycle time is identical if no robot waiting time is taken into account. Thus, by Wu et al. [20], for the extended backward strategy, without robot waiting, the robot cycle time is

\[
\psi = 2(n + 1)(\mu + \lambda)
\]

At Step \( i, i \in \mathbb{N}_n \), to complete a wafer processing cycle, the robot activity sequence (unloads a processed wafer from Step \( i \) and moves to Step \( i + 1 \)) with time \( \mu \rightarrow \) waits at \( q_{i1} \) with time \( \alpha_{i1} \rightarrow \) loads a wafer into Step \( i + 1 \) with time \( \lambda \rightarrow \) moves to Step \( i + 2 \) with time \( \mu \rightarrow \) waits at \( q_{i2} \) with time \( \alpha_{i2} \rightarrow \) unloads a processed wafer from Step \( i + 1 \) and moves to Step \( i + 2 \) with time \( \lambda + \mu \rightarrow \) waits at \( q_{i1} \) with time \( \alpha_{i1} \rightarrow \) loads into Step \( i + 1 \) with time \( \lambda + \mu \rightarrow \) a wafer is being processed at Step \( i \) with time \( \alpha_i \rightarrow \) starts to unload from Step \( i \) again should be executed, which takes \( \alpha_i + 4 \lambda + 3 \mu + \alpha_{i1} + \alpha_{i2} \) time units. Since \( m_i - z_i \) wafers are being processed concurrently at this step, the wafer processing cycle time at Step \( i, i \in \mathbb{N}_n \), is

\[
\zeta_i = (\alpha_i + 4 \lambda + 3 \mu + \alpha_{i1} + \alpha_{i2})/(m_i - z_i), i \in \mathbb{N}_n
\]

By removing \( \alpha_{i1} + \alpha_{i2} + \alpha_i \) from (3.2), we can obtain the allowed shortest wafer cycle time at Step \( i, i \in \mathbb{N}_n \), as:

\[
\zeta_i = (\alpha_i + 4 \lambda + 3 \mu)/(m_i - z_i), i \in \mathbb{N}_n
\]

Let \( \Pi \) denote the system cycle time. We present how to calculate the WRT \( \tau_i \) at Step \( i, i \in \mathbb{N}_n \), as follows. With the PN model shown in Fig. 2, we can deduce that, by the \( k \)th firing of \( l_i \), the \( k \)th wafer is loaded into \( p_i \). Then, by the \( (k + m_i - z_i) \)th firing of \( u_i \), this wafer is unloaded from \( p_i \). During this time interval, \( (m_i - z_i) \) cycles are executed except that transitions \( u_i, l_{i+1}, y_{i+1}, u_{i+1}, l_i \), and the robot waiting at \( q_{i+1} \) and \( q_{i+2} \) are executed \( (m_i - z_i - 1) \) times. Hence, we have

\[
\tau_i = (m_i - z_i) \times \Pi - (4 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i), i \in \mathbb{N}_n
\]

After unloading a wafer from a chamber (let \( c_i \) denote this chamber) at Step \( i \), the cleaning operation of \( c_i \) starts immediately. Assume that \( \kappa_i \) time units later, the robot starts to drop another wafer into \( c_i \). Then, we present how to calculate \( \kappa_i \).

If \( z_i = 0 \), by the PN in Fig. 2, after unloading a wafer from \( c_i \), the robot sequentially performs the following activities: moves to Step \( i + 1 \) with time \( \mu \rightarrow \) waits in \( q_{i+1} \) with time \( \alpha_{i+1} \rightarrow \) fires \( l_{i+1} \) with time \( \lambda \rightarrow \) fires \( y_{i+1} \) with time \( \mu \rightarrow \) waits in \( q_{i+2} \) with time \( \alpha_{i+2} \rightarrow \) fires \( u_{i+1} \) with time \( \lambda + \mu \rightarrow \) waits in \( q_{i+1} \) with time \( \alpha_i \rightarrow \) for loading another wafer into \( c_i \) which takes \( 2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i \) time units. It means that \( \kappa_i = 2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i \).

If \( z_i = 1 \), after waiting in \( q_{i+1} \), the robot loads the wafer into the earliest emptied chamber at Step \( i \), and in the next cycle, it loads a new wafer into \( c_i \), which takes \( \Pi + 2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i \) time units, i.e., we have \( \kappa_i = \Pi + (2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i) \).

Similarly, if \( z_i = 2 \), we have \( \kappa_i = 2 \Pi + 2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i \). Thus, for the general case, we have

\[
\kappa_i = z_i \times \Pi + (2 \lambda + 3 \mu + \alpha_{i+1} + \alpha_{i+2} + \alpha_i), i \in \mathbb{N}_n
\]
With (3.1)-(3.5), we can analyze if a feasible schedule exists and present how to find it if existing.

B. Schedulability Condition

With WRT constraint, a critical issue is the existence of a feasible schedule. Before presenting the schedulability condition, we discuss how to calculate the system cycle time \( \Pi \) first.

By Wu et al. [20], we know that the robot cycle time with robot waiting time being included should be equal to \( \Pi \). Thus, we have Constraint \( a)\): \( \sum_{i=0}^{\infty} (\omega_{1i} + \omega_{2i}) + \psi = \Pi \), or
\[
\sum_{i=0}^{\infty} (\omega_{1i} + \omega_{2i}) = \Pi - \psi. \]
Hence, to find a schedule for the system, the key is to allocate \( \Pi - \psi \) to \( \omega_j \)'s, \( i \in \Omega_n \) and \( j \in \mathbb{N}_2 \).

For a chamber at Step \( i \), \( i \in \mathbb{N}_n \), no wafer can be loaded into it before its cleaning operation is finished. Thus, we have Constraint \( c)\): \( \kappa_i \geq \alpha_i \) where \( \alpha_i \) denotes the time for cleaning the chamber and \( \kappa_i \) is determined by (3.5). Meanwhile, the WRT \( \tau_i \) in a chamber at Step \( i \), \( i \in \mathbb{N}_n \), should be longer than or equal to the wafer processing time \( \alpha_i \), i.e., we have Constraint \( d)\): \( \tau_i \geq \alpha_i \) where \( \tau_i \) is determined by (3.4). Finally, we have Constraint \( e)\): \( \alpha_1 \geq 0 \) and \( \alpha_2 \geq 0 \), \( i \in \Omega_n \).

Note that, for Constraint \( b)\), it can be rewritten as
\[
\omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1} + (m_i - z_i) \times (\Pi - \xi_i) \leq (m_i - z_i) \times \Pi, \]
or
\[
\alpha_i \leq (m_i - z_i) \times \Pi - (\omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1} + 4\lambda + 3\mu + \alpha_i) \leq (m_i - z_i) \times \Pi, \]
where \( \xi_i \) is determined by (3.4). Thus, with minimizing \( \Pi \) as the objective, Linear Program (3.6) is developed to obtain the minimal \( \Pi \), and set \( \omega_1 \)'s and \( \omega_2 \)'s, \( i \in \Omega_n \) as well.

Minimize \( \Pi \)
\[
\sum_{i=0}^{\infty} (\omega_{1i} + \omega_{2i}) = \Pi - \psi
\]
\[
\omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1} \leq (m_i - z_i) \times (\Pi - \xi_i), \quad i \in \mathbb{N}_n
\]
\[
\alpha_i \leq (m_i - z_i) \times \Pi - (\omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1}) \leq (m_i - z_i) \times \Pi
\]
\[
\omega_{1i} \geq 0, \quad i \in \Omega_n
\]
\[
\alpha_i \geq 0, \quad i \in \Omega_n
\]

With the minimal \( \Pi \) being determined by solving (3.6), we need to check if a feasible schedule exists, or if the WRT constraint can be met. By Definition 4, for a feasible schedule, \( \tau_i \leq \alpha_i + \delta_i \) has to be satisfied, i.e., \( (m_i - z_i) \times (4\lambda + 3\mu + \omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1}) \leq \alpha_i + \delta_i \), \( i \in \mathbb{N}_n \). Then, based on (3.6), we present Linear Program (3.7) with minimizing \( \Theta \) as the objective to check if a feasible schedule exists, where \( \Theta \geq \Pi \).

Minimize \( \Theta \)
\[
\Theta \geq \Pi
\]
\[
\sum_{i=0}^{\infty} (\omega_{1i} + \omega_{2i}) = \Theta - \psi
\]
\[
\omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1} \leq (m_i - z_i) \times (\Theta - \xi_i), \quad i \in \mathbb{N}_n
\]
\[
\alpha_i \leq z_i \times \Theta + (2\lambda + 3\mu + \omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1}), \quad i \in \mathbb{N}_n
\]
\[
(m_i - z_i) \times (4\lambda + 3\mu + \omega_{(i+1)1} + \omega_{(i+1)2} + \omega_{1}) \leq \alpha_i + \delta_i, \quad i \in \mathbb{N}_n
\]
\[
\omega_{1i} \geq 0, \quad i \in \Omega_n
\]
\[
\alpha_i \geq 0, \quad i \in \Omega_n
\]

By (3.7), if a solution is found by determining \( \omega_1 \)'s and \( \omega_2 \)'s, \( i \in \Omega_n \), it must be a feasible schedule. On the contrary, if there is no solution for (3.7), we cannot find a feasible schedule, or the WRT constraint is violated with the extended backward strategy.

By using a commercial solver, a linear program can be efficiently solved. Thus, the proposed approach is computationally efficient. Since the WRT constraint is embedded into (3.7), we have the following schedulability result immediately.

Theorem 3.1: For a single-arm cluster tool with a chamber cleaning operation and WRT constraint, by using the extended backward strategy, if a solution can be found by solving Linear Program (3.7), the obtained schedule is feasible.

It should also be pointed out that the cycle time \( \Theta \) for the obtained schedule is minimized by Linear Program (3.7), the maximal productivity is achieved for a determined extended scheduling strategy.

C. Experimental Performance of the Proposed method

Next, we generate some problem instances randomly to compare the gap between the lower bound of the global cycle time and the cycle time obtained by the proposed method in this work.

Let \( \Theta \) denote the lower bound of the global cycle time. In Step \( i \), \( i \in \mathbb{N}_n \), after a wafer finishes its processing (taking \( \alpha_i \) time units), the robot removes this wafer from Step \( i \) (taking \( \lambda \) time units). Then, the cleaning operation begins (\( \alpha_i \)) immediately. From the start of the cleaning operation at Step \( i \), it takes at least \( 2\lambda + 3\mu \) time units for the robot to load another wafer into Step \( i \), i.e., moving to the next step (\( \mu \)) loading a wafer there (\( \lambda \)) moving to Step \( i - 1 \) (\( \mu \)) unloading another wafer and moving to Step \( i \) again (\( \lambda + \mu \)). If the cleaning operation finished, then, loading that wafer into Step \( i \).

It follows from the above analysis that the workload at Step \( i \) is greater than or equal to \( \left[ \alpha_i + 2\lambda + \max \{ \alpha_i, 2\lambda + 3\mu \} \right] / m_n \), where \( m_n \) denotes the number of the parallel chambers at Step \( i \).

Hence, \( \Theta \) can be calculated by
\[
\Theta = \max \left[ \frac{\left[ \alpha_i + 2\lambda + \max \{ \alpha_i, 2\lambda + 3\mu \} \right]}{m_n} \right], \quad i \in \mathbb{N}_n.
\]
With $\varTheta'$, we define the relative optimality gap between $\varTheta'$ and $\varTheta$ as

$$\text{Gap} (\%) = \frac{\varTheta - \varTheta'}{\varTheta} \times 100$$  \hspace{1cm} (3.8)$$

Next, we use two cases to show the variation of the gap. In the first case, a cluster tool has three processing steps and its configuration is $(m_1, m_2, m_3) = (2, 2, 1)$ and $(z_1, z_2, z_3) = (1, 0, 0)$. The robot loading/unloading takes six time units and, the robot moving with or without holding a wafer takes two time units, or we have $\lambda = 6$ and, $\mu = 2$.

For Step 1, wafer processing time $\alpha_1 = 120$ and, chamber cleaning time $\alpha_3 = 140$; For Step 3, $\alpha_5 = 60$, $\alpha_3 = 18$; For Step 2, we have $\alpha_2 \in [80, 220]$, and $\alpha_2 \in [30, 180]$, note that these ranges are most common in etching and chemical vapor deposition steps. After being processed, a wafer can stay at each step for no more than 30 time units, or $\delta_1 = \delta_2 = \delta_3 = 30$. The experimental optimality gap is shown in Fig. 3.

Fig. 3. Experimental optimality gap for Case 1.

In Fig. 3, the area with no color means that there is no feasible solution or the WRT constraint is violated for the corresponding processing and cleaning times. This is because that Step 2 has two parallel chambers to process wafer at the same time ($m_2 = 2$ and $z_2 = 0$). Hence, if the wafer processing time is too small, it could violate the WRT constraint easily. Also, if both the cleaning and wafer processing times are very large, the system could also violate the WRT constraint easily due to that the system cycle time could be too large.

For the area with yellow color, we know that, about half of them, we have a zero gap, and for the remaining, the gap is very small, about 8%.

In Case 2, a cluster tool has three processing steps and its configuration is $(m_1, m_2, m_3) = (1, 3, 3)$ and, $(z_1, z_2, z_3) = (0, 1, 1)$. The robot task times are $\lambda = 5$ and, $\mu = 3$, respectively. For Step 1, $\alpha_1 = 100$, $\alpha_1 = 20$; For Step 3, $\alpha_1 = 210$, $\alpha_3 = 100$; For Step 2, we have $\alpha_2 \in [100, 240]$, and $\alpha_2 \in [60, 180]$. After being processed, a wafer can stay at each step for no more than 30 time units, or $\delta_1 = \delta_2 = \delta_3 = 30$. For this case, its experimental optimality gap is shown in Fig. 4.

From Fig. 4, we can find that, for nearly all of them, we have a zero gap, and for the remaining, the gap is less than 5%.

Actually, the proposed method is globally optimal in most practical cases and the relative gap is also not very big even if it is not globally optimal. Furthermore, such a scheduling strategy is simple and very close to the traditional backward strategy. Thus, it can be applicable in the practice.

Fig. 4. Experimental optimality gap for Case 2.

IV. ILLUSTRATIVE EXAMPLES

This section uses two industrial examples to show the application of the derived approach.

Example 1: A cluster tool has three processing steps. There is one processing chamber (PC) for Steps 1 and 2, and two PCs for Step 3, i.e., $(m_1, m_2, m_3) = (1, 1, 2)$. The extended backward strategy is adopted by keeping one chamber in Step 3 empty, or we have $(z_1, z_2, z_3) = (0, 0, 1)$.

For a PC at Steps 1, 2, and 3, it takes 30, 30, and 18 time units to process a wafer, 40, 40, and 140 time units to be cleaned; eight time units for the robot to unload/load a wafer at a step; two time units for the robot to move between the PCs with or without holding a wafer. In other words, we have $\alpha_1 = 30$, $\alpha_2 = 30$, $\alpha_3 = 32$; $\alpha_4 = 40$, $\alpha_5 = 40$, $\alpha_6 = 140$, $\lambda = 8$, $\mu = 2$.

Minimize $\Pi$

$$\begin{aligned}
&\omega_{b_1} + \omega_{b_2} + \omega_{b_3} + \omega_{c_1} + \omega_{c_2} + \omega_{c_3} + \omega_{d_1} = \Pi - 80 \\
&\omega_{c_1} + \omega_{c_2} + \omega_{c_3} \leq \Pi - 68 \\
&\omega_{c_1} + \omega_{c_2} + \omega_{c_3} \leq \Pi - 68 \\
&\omega_{c_1} + \omega_{c_3} + \omega_{c_3} \leq \Pi - 70 \\
&40 \leq \omega_{c_1} + \omega_{c_2} + \omega_{c_3} \\
&40 \leq \omega_{c_1} + \omega_{c_2} + \omega_{c_3} \\
&140 \leq \Pi + (22 + \omega_{b_1} + \omega_{b_2} + \omega_{b_3}) \\
&\omega_{b_1}, \omega_{b_2}, \omega_{b_3}, \omega_{c_1}, \omega_{c_2}, \omega_{c_3}, \omega_{d_1}, \omega_{d_2}, \omega_{d_3} \geq 0
\end{aligned}$$

(4.1)

Case 1: After being processed, a wafer can stay at each step for no more than 30 time units, or $\delta_1 = \delta_2 = \delta_3 = 30$. For this case, we have: $\psi = 2(n + 1)(\mu + \lambda) = 8 \times 10 = 80$, $\xi_1 = (\alpha_1 + 4\lambda + 3\mu)/(m_1 - z_1) = 68$, $\xi_2 = (\alpha_1 + 4\lambda + 3\mu)/(m_2 - z_2) = 68$, and $\xi_3 = (\alpha_1 + 4\lambda + 3\mu)/(m_3 - z_3) = 70$.

Then, by (3.6), to calculate the optimal system cycle time,
we have Linear Program (4.1). By solving (4.1), we have \( I = 108 \). Then, based on the decided \( I \) and (3.7), we have Linear Program (4.2) to test if a feasible schedule exists. By solving (4.2), a feasible schedule is obtained with \( \Theta = I = 108 \), \( \omega_2 = 10 \), \( \omega_3 = 8 \), \( \omega_4 = 10 \), and \( \omega_5 = \omega_6 = \omega_7 = \omega_8 = 0 \). Simulation is carried out to verify the correctness of the schedule and the result is shown in Table I.

\[
\text{Minimize } \Theta
\]

\[
\begin{align*}
\Theta & \geq 108 \\
\omega_0 + \omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & = \Theta - 80 \\
\omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} & \leq \Theta - 68 \\
\omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} & \leq \Theta - 68 \\
\omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & \leq \Theta - 70 \\
\end{align*}
\]

(4.2)

Table I. The simulation result for Example 1.

<table>
<thead>
<tr>
<th>No.</th>
<th>Time (s)</th>
<th>Activities</th>
<th>Time Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-8</td>
<td>Unload from S3 1</td>
<td>108-116 Unload from S3 2</td>
</tr>
<tr>
<td>2</td>
<td>8-10</td>
<td>Move to LL</td>
<td>116-118 Move to LL</td>
</tr>
<tr>
<td>3</td>
<td>10-18</td>
<td>Load into LL</td>
<td>118-126 Load into LL</td>
</tr>
<tr>
<td>4</td>
<td>18-20</td>
<td>Move to S2</td>
<td>126-128 Move to S2</td>
</tr>
<tr>
<td>5</td>
<td>20-28</td>
<td>Unload from S2</td>
<td>128-136 Unload from S2</td>
</tr>
<tr>
<td>6</td>
<td>28-30</td>
<td>Move to S3 2</td>
<td>136-138 Move to S3 1</td>
</tr>
<tr>
<td>7</td>
<td>30-40</td>
<td>Waiting (10 s)</td>
<td>138-148 Waiting (10 s)</td>
</tr>
<tr>
<td>8</td>
<td>40-48</td>
<td>Load into S3 2</td>
<td>148-156 Load into S3 1</td>
</tr>
<tr>
<td>9</td>
<td>48-50</td>
<td>Move to S1</td>
<td>156-158 Move to S1</td>
</tr>
<tr>
<td>10</td>
<td>50-58</td>
<td>Unload from S1</td>
<td>158-166 Unload from S1</td>
</tr>
<tr>
<td>11</td>
<td>58-68</td>
<td>Move to S2</td>
<td>166-168 Move to S2</td>
</tr>
<tr>
<td>12</td>
<td>68-68</td>
<td>Waiting (8 s)</td>
<td>178-178 Waiting (8 s)</td>
</tr>
<tr>
<td>13</td>
<td>68-76</td>
<td>Load into S2</td>
<td>178-184 Load into S2</td>
</tr>
<tr>
<td>14</td>
<td>76-88</td>
<td>Move to LL</td>
<td>184-186 Move to LL</td>
</tr>
<tr>
<td>15</td>
<td>78-88</td>
<td>Waiting (10 s)</td>
<td>186-196 Waiting (10 s)</td>
</tr>
<tr>
<td>16</td>
<td>88-96</td>
<td>Unload from LL</td>
<td>196-204 Unload from LL</td>
</tr>
<tr>
<td>17</td>
<td>96-98</td>
<td>Move to S1</td>
<td>204-206 Move to S1</td>
</tr>
<tr>
<td>18</td>
<td>98-106</td>
<td>Load into S1</td>
<td>206-214 Load into S1</td>
</tr>
<tr>
<td>19</td>
<td>106-108</td>
<td>Move to S3 2</td>
<td>214-216 Move to S3 1</td>
</tr>
</tbody>
</table>

From Table I, we can see that a wafer is loaded into Chamber 2 at Step 3 (denoted as S3_2 in the table) at time instant 48 and this wafer is unloaded from the chamber at time instant 108. Thus, we have \( \alpha_3 = 32 < \tau_1 = 108 - 48 = 60 < \alpha_3 + \delta_2 = 62 \).

By observing Chamber 1 at Step 3 (denoted as S3_1), it can be seen that after a completed wafer is unloaded from it at time instant 8, a raw wafer is loaded into it at time instant 148, leading to \( \kappa_3 = 148 - 8 = 140 = \alpha_3 = 140 \), or the chamber cleaning operation in Step 3 is totally completed.

Similarly, we have \( \alpha_2 = 30 < \tau_2 = 128 - 76 = 52 < \alpha_3 + \delta_2 = 60 \), and \( \kappa_2 = 68 - 28 = 40 = \alpha_2 = 40 \); and \( \alpha_1 = 30 < \tau_1 = 158 - 106 = 52 < \alpha_1 + \delta_1 = 60 \), and \( \kappa_1 = 98 - 58 = 40 = \alpha_1 = 40 \). Furthermore, it is a cyclic schedule. Therefore, we can obtain a feasible cyclic schedule by the approach.

Table II. The simulation result for Example 2. 

Case 2: In this case, \( \delta_i \)'s, \( i \in \mathbb{N}_3 \), are changed to \( \delta_i = \delta_2 = \delta_3 = 25 \), and the others are unchanged.

\[
\text{Minimize } \Theta
\]

\[
\begin{align*}
\Theta & \geq 108 \\
\omega_0 + \omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & = \Theta - 80 \\
\omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} & \leq \Theta - 68 \\
\omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} & \leq \Theta - 68 \\
\omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & \leq \Theta - 70 \\
\end{align*}
\]

(4.3)

To test the existence of a feasible schedule, we have Linear Program (4.3). By (4.3), no solution is found, implying that no feasible schedule can be obtained for this case.

Example 2: A cluster tool has three processing steps. There are 3 PCs at Step 1, and 2 PCs at Steps 2 and 3, or we have \( (m_1, m_2, m_3) = (3, 2, 2) \). The extended backward strategy is adopted by keeping one chamber empty at each step, or we have \( (z_1, z_2, z_3) = (1, 1, 1) \).

For a PC at Steps 1, 2, and 3, it takes 140, 60, and 90 time units to process a wafer, 120, 100, and 120 time units to be cleaned; five time units for the robot to unload/load a wafer at a step; two time units for the robot to move between PCs without holding a wafer. After being processed, a wafer can stay in a PC for no more than 20 time units. In other words, we have \( \alpha_1 = 140, \alpha_2 = 60, \alpha_3 = 90; \alpha_1 = 120, \alpha_2 = 100, \alpha_3 = 120; \lambda = 5, \mu = 2 \), and \( \delta_i = \delta_i = 20. \)

\[
\text{Minimize } \Pi
\]

\[
\begin{align*}
\omega_0 + \omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & = \Pi - 56 \\
\omega_2 + \omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & \leq \Pi - 83 \\
\omega_4 + \omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & \leq \Pi - 86 \\
\omega_6 + \omega_8 + \omega_{10} + \omega_{12} + \omega_{14} & \leq \Pi - 116 \\
\end{align*}
\]

(4.4)

For this case, we have \( \psi = 2(n + 1)(\mu + \lambda) = 8 \times 7 = 56, \xi = (\alpha_0 + 4\lambda + 3\mu)(m_1 - z_1) = (140 + 26)/2 = 83, \eta = (\alpha_0 + 4\lambda + 3\mu)(m_2 - z_2) = 86, \) and \( \eta_i = (\alpha_0 + 4\lambda + 3\mu)(m_i - z_i) = 116. \) Then, by (3.6), to calculate the optimal cycle time, we have
Linear Program (4.4).

By solving (4.4), we have $\mathcal{I} = 116$. Then, based on the decided $\mathcal{I}$ and (3.7), we have Linear Program (4.5) to check if there is a feasible schedule. By solving (4.5), a feasible schedule is obtained with $\Theta = 116$, $\alpha_0 = 50$, $\Delta_2 = 10$, and $\alpha_1 = \alpha_2 = \alpha_4 = \omega_8 = \alpha_2 = 0$. To verify the correctness of the obtained schedule, simulation is done as shown in Table II.

$$
\begin{align*}
\Theta &\geq 116 \\
\omega_{01} + \omega_{02} + \omega_{21} + \omega_{22} + \omega_{31} + \omega_{32} &= \Theta - 56 \\
\omega_{31} + \omega_{32} &\leq \Theta - 86 \\
\omega_{01} + \omega_{02} + \omega_{31} &\leq -116 \\
120 &\leq \Theta + (16 + \omega_{01} + \omega_{21} + \alpha_1) \\
100 &\leq \Theta + (16 + \omega_{01} + \omega_{21} + \omega_{31}) \\
120 &\leq \Theta + (16 + \omega_{02} + \omega_{22} + \omega_4) \\
2 \theta - 26 - (\omega_{21} + \omega_{02} + \omega_{31}) &\leq 160 \\
\Theta - 26 - (\omega_{01} + \omega_{22} + \omega_{31}) &\leq 80 \\
\Theta - 26 - (\omega_{01} + \omega_{22} + \omega_{31}) &\leq 110 \\
\omega_{01}, \omega_{02}, \omega_{31}, \omega_{22}, \omega_{01}, \omega_{22}, \omega_{31}, \omega_{22} &\geq 0
\end{align*}
$$

From Table II, we observe that a wafer is unloaded into the third chamber at Step 1 (denoted as S1_3) at time instant 114 and this wafer is unloaded at time instant 260. Thus, we have $\alpha_1 = 140 < \tau_1 = 260 - 114 = 146 < \alpha_2 + \delta_2 = 160$.

Table II. The simulation result for Example 2.

<table>
<thead>
<tr>
<th>No.</th>
<th>Time(s)</th>
<th>Activities</th>
<th>Time</th>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-5</td>
<td>Unload from S3 1</td>
<td>149-151</td>
<td>Move to S2 1</td>
</tr>
<tr>
<td>2</td>
<td>5-7</td>
<td>Move to LL</td>
<td>151-161</td>
<td>Waiting (10)</td>
</tr>
<tr>
<td>3</td>
<td>7-12</td>
<td>Load into LL</td>
<td>161-166</td>
<td>Load into S2 1</td>
</tr>
<tr>
<td>4</td>
<td>12-14</td>
<td>Move to S2 1</td>
<td>166-168</td>
<td>Move to LL</td>
</tr>
<tr>
<td>5</td>
<td>14-19</td>
<td>Unload from S2 1</td>
<td>168-218</td>
<td>Waiting (50)</td>
</tr>
<tr>
<td>6</td>
<td>19-20</td>
<td>Move to S3 2</td>
<td>218-223</td>
<td>Unload from LL</td>
</tr>
<tr>
<td>7</td>
<td>21-26</td>
<td>Load into S3 2</td>
<td>223-227</td>
<td>Move to S1 1</td>
</tr>
<tr>
<td>8</td>
<td>27-28</td>
<td>Move to S1 1</td>
<td>225-230</td>
<td>Load into S1 1</td>
</tr>
<tr>
<td>9</td>
<td>29-33</td>
<td>Unload from S1 1</td>
<td>230-232</td>
<td>Move to S3 1</td>
</tr>
<tr>
<td>10</td>
<td>33-35</td>
<td>Move to S2 2</td>
<td>232-237</td>
<td>Unload from S3 1</td>
</tr>
<tr>
<td>11</td>
<td>35-45</td>
<td>Waiting (10)</td>
<td>237-239</td>
<td>Move to LL</td>
</tr>
<tr>
<td>12</td>
<td>42-50</td>
<td>Load into S2 2</td>
<td>239-244</td>
<td>Load into LL</td>
</tr>
<tr>
<td>13</td>
<td>50-52</td>
<td>Move to LL</td>
<td>244-246</td>
<td>Move to S2 1</td>
</tr>
<tr>
<td>14</td>
<td>52-102</td>
<td>Waiting (50)</td>
<td>246-251</td>
<td>Unload from S1 2</td>
</tr>
<tr>
<td>15</td>
<td>102-107</td>
<td>Unload from LL</td>
<td>251-253</td>
<td>Move to S3 2</td>
</tr>
<tr>
<td>16</td>
<td>107-109</td>
<td>Move to S1 3</td>
<td>253-258</td>
<td>Load into S3 2</td>
</tr>
<tr>
<td>17</td>
<td>109-114</td>
<td>Load into S1 3</td>
<td>258-260</td>
<td>Move to S1 3</td>
</tr>
<tr>
<td>18</td>
<td>114-116</td>
<td>Move to S3 2</td>
<td>260-265</td>
<td>Unload from S1 3</td>
</tr>
<tr>
<td>19</td>
<td>116-121</td>
<td>Unload from S3 2</td>
<td>265-267</td>
<td>Move to S2 2</td>
</tr>
<tr>
<td>20</td>
<td>121-123</td>
<td>Move to LL</td>
<td>267-277</td>
<td>Waiting (10)</td>
</tr>
<tr>
<td>21</td>
<td>123-128</td>
<td>Load into LL</td>
<td>277-282</td>
<td>Load into S2 2</td>
</tr>
<tr>
<td>22</td>
<td>128-130</td>
<td>Move to S2 2</td>
<td>282-284</td>
<td>Move to LL</td>
</tr>
<tr>
<td>23</td>
<td>130-135</td>
<td>Unload from S2 2</td>
<td>284-334</td>
<td>Waiting (50)</td>
</tr>
<tr>
<td>24</td>
<td>135-137</td>
<td>Move to S3 1</td>
<td>334-339</td>
<td>Unload from LL</td>
</tr>
<tr>
<td>25</td>
<td>137-142</td>
<td>Load into S1 3</td>
<td>339-341</td>
<td>Move to S1 2</td>
</tr>
<tr>
<td>26</td>
<td>142-144</td>
<td>Move to S1 2</td>
<td>341-346</td>
<td>Load into S1 2</td>
</tr>
<tr>
<td>27</td>
<td>144-149</td>
<td>Unload from S1 2</td>
<td>346-348</td>
<td>Move to S1 2</td>
</tr>
</tbody>
</table>

Also, we can see that, after a completed wafer is unloaded from the first chamber at Step 1 (S1_1) at time instant 33, a raw wafer is loaded into it at time instant 225 such that we have $\alpha_1 = 225 - 33 = 192 > \alpha_1 = 120$, or the chamber cleaning operation at Step 1 is totally completed.

Similarly, we have $\alpha_2 = 60 < \tau_1 = 130 - 50 = 80 = \alpha_2 + \delta_2 = 80$, and $\alpha_3 = 161 - 19 = 142 > \alpha_2 = 100$; and $\alpha_4 = 90 < \tau_1 = 116 - 26 = 90 < \alpha_3 + \delta_2 = 110$, and $\alpha_5 = 137 - 5 = 132 > \alpha_1 = 120$. Furthermore, it is a cyclic schedule. Therefore, we can obtain a feasible cyclic schedule by our approach.

V. CONCLUSIONS

With the shrink down of wafer circuit widths, stringent quality control is desired for wafer processing such that, in some leading labs, a chamber cleaning operation is performed after each wafer is removed from a chamber. By considering such operations, to obtain an efficient schedule, Yu et al. [36] propose an extended backward strategy for single-arm cluster tools by keeping some numbers of chambers empty at a processing step. However, no wafer residency time constraint is considered in their work. This work represents the first one which takes both chamber cleaning operations and wafer residency time constraint into consideration. First, it develops a timed Petri net model that is independent of the wafer flow pattern with no deadlock to describe the dynamic properties of the system. Based on this model, two linear programs are formulated to calculate the minimum cycle time and test if there is a feasible schedule. As a linear program can be efficiently solved, the derived approach is computationally efficient.

In this work, a chamber cleaning operation is required after a chamber finishes processing just $m = 1$ wafer. Hence, it is very meaningful to extend the result of this work to the cases with $m > 1$. Besides, the activity time is treated as known constant in this work. Actually, they could be subject to random variation, resulting in residency time fluctuation in a chamber, which makes the scheduling problem more complex and challenging. These issues are our future work.

REFERENCES


Fajun Yang received the B. S. degree in Industrial Engineering from Hunan University of Science and Technology, Hunan, China, in 2011, the Ph. D. degree in Mechanical Engineering from Guangdong University of Technology, China, in 2016. From 2015-2016, he was a Visiting Student with New Jersey Institute of Technology, Newark, NJ, USA. He is currently a Research Fellow with Nanyang Technological University, Singapore. He has 10+ international journal papers (majority in the IEEE Transactions). His interests are Petri nets, production planning, discrete event systems, scheduling and control. He has served as a reviewer for a number of journals.
NaiQi Wu (M’04-SM’05) received his B. S. Degree in Electrical Engineering from Anhui University of Technology, Huainan, China, in 1982, the M. S. and Ph. D. Degrees in Systems Engineering both from Xi’an Jiaotong University, Xi’an, China in 1985 and 1988, respectively. From 1988 to 1995, he was with Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, and from 1995 to 1998, with Shantou University, Shantou, China. He moved to Guangdong University of Technology, Guangzhou, China in 1998. He joined Macau University of Science and Technology, Macau in 2013. He is currently a Professor at the Institute of Systems Engineering, Macau University of Science and Technology, Macau. His research interests include production planning and scheduling, manufacturing system modeling and control, discrete event systems, Petri net theory and applications, intelligent transportation systems, and energy systems. He is the author or coauthor of one book, five book chapters, and 130+ peer-reviewed journal papers. Dr. Wu was an associate editor of the IEEE Transactions on Systems, Man, & Cybernetics, Part C, IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Systems, Man, & Cybernetics: Systems, and editor in chief of Industrial Engineering Journal. He is an associate editor of Information Sciences and IEEE/CAA Journal of Automatica Sinica.

Kaizhou Gao received the B.Sc. and master degree from China in 2005 and 2008 respectively, and the Ph.D. degree from Nanyang Technological University, Singapore, 2016. From 2008 to 2012, he was with the School of Computer, LiaoCheng University, China. He was a research associate in the School of Electronic and Electrical engineering (EEE), NTU, Singapore, from Feb. 2012 to Sep. 2013. From Oct. 2013 to Mar 2015, he was a software engineer in Singapore Institute of Manufacturing Technology (SIMTech), A*star, Singapore. Since April 2015, he was a research fellow in the School of Electronic and Electrical engineering, NTU, Singapore. His research interests include intelligent computation, optimization, scheduling, and intelligent transportation. He has published over 50 refereed papers.

Chunjian Zhang received the B. S. degree and Ph. D. degree in Industrial Engineering from Huazhong University of Science and Technology, Hubei, China, in 2011 and 2016, respectively. From 2015 to 2016, he was a Visiting Student with National University of Singapore, Singapore. He is currently a Research Fellow with Nanyang Technological University, Singapore. His interests include evolutionary algorithms, constrained optimization, multi-objective optimization, and their applications in industry.

Yuting Zhu received the B.S. degree from Southeast University, Jiangsu, China, in 2016. Currently, she is a Ph. D candidate in Nanyang Technology University, Singapore. Her research interests include discrete event systems, and supervisory control.

Rong Su (M’11-SM’14) received the B.E. degree in automatic control from the University of Science and Technology of China, Hefei, China, in 1997, and the M.A.Sc. and Ph.D. degrees both in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 2000 and 2004, respectively. Since then he was affiliated with University of Waterloo and Technical University of Eindhoven before he joined Nanyang Technological University, Singapore, in 2010. His research interests include discrete event systems, supervisory control, model-based fault diagnosis, multiagent systems, optimization and scheduling with applications in green buildings, flexible manufacturing, power management, and intelligent transportation systems. In the aforementioned areas, he has more than 130 publications in journals, book chapters, and conference proceedings, and two patents. Dr. Su is an Associate Editor for Journal of Discrete Event Dynamic Systems: Theory and Applications, Journal of Control and Decision, and Transactions of the Institute of Measurement and Control, and the Chair of IEEE Control Systems Society Technical Committee on Smart Cities.

Yan Qiao (M’16) received the B. S. and Ph. D. degrees in Industrial Engineering and Mechanical Engineering from the Guangdong University of Technology, Guangzhou, China, in 2009 and 2015, respectively.

From 2014 to 2015, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He is currently a Post-Doctoral Research Associate with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau. He has one book chapter and 20+ international journal papers (majority in the IEEE Transactions). His research interests include discrete event systems, production planning, Petri nets, scheduling and control.

Dr. Qiao was a recipient of the QSI Best Application Paper Award Finalist of 2011 IEEE International Conference on Automation Science and Engineering, the Best Student Paper Award of 2012 IEEE International Conference on Networking, Sensing and Control, and the Best Conference Paper Award Finalist of 2016 IEEE International Conference on Automation Science and Engineering. He has served as a reviewer for a number of journals.