Polynomial approach to optimal one-wafer cyclic scheduling of treelike hybrid multi-cluster tools via Petri nets

Yang, Fajun; Wu, Naiqi; Qiao, Yan; Su, Rong

2017

Yang, F., Wu, N., Qiao, Y., & Su, R. (2018). Polynomial approach to optimal one-wafer cyclic scheduling of treelike hybrid multi-cluster tools via Petri nets. *IEEE/CAA Journal of Automatica Sinica*, 5(1), 270-280. doi:10.1109/jas.2017.7510772

<https://hdl.handle.net/10356/141782>

<https://doi.org/10.1109/jas.2017.7510772>

© 2018 The Chinese Association of Automation (published by IEEE). This is an open-access article distributed under the terms of the Creative Commons Attribution License.

Downloaded on 26 Apr 2023 20:03:53 SGT

Polynomial Approach to Optimal One-wafer Cyclic Scheduling of Treelike Hybrid Multi-Cluster Tools via Petri Nets

Fajun Yang, Naiqi Wu, *Senior Member, IEEE*, Yan Qiao, *Member, IEEE*, and Rong Su, *Senior Member, IEEE*

Abstract—A treelike hybrid multi-cluster tool is composed of both single-arm and dual-arm cluster tools with a treelike topology. Scheduling such a tool is challenging. For a hybrid treelike multi-cluster tool whose bottleneck individual tool is process-bound, this work aims at finding its optimal one-wafer cyclic schedule. It is modeled with Petri nets such that a one-wafer cyclic schedule is parameterized as its robots' waiting time. Based on the model, this work proves the existence of its one-wafer cyclic schedule that features with the ease of industrial implementation. Then, computationally efficient algorithms are proposed to find the minimal cycle time and optimal one-wafer cyclic schedule. Multi-cluster tool examples are given to illustrate the proposed approach. The use of the found schedules enables industrial multi-cluster tools to operate with their highest productivity.

Index Terms—Multi-cluster tool, Petri nets, scheduling, semiconductor manufacturing.

NOMENCLATURE

$b[i]$	The index for the step that is formed by a buffering module linking C_i and C_{i+1} , and C_i is not a fork tool.
$B[i]$	$= n[i] - 1, i \in \mathbb{N}_K$
BM	Buffering module.
$b[i]_k$	The index for the step representing an outgoing module in C_i , with $1 \leq k \leq f[i]$.
C_i	The i th cluster tool in a K -cluster tool.
$D[i]$	$= \{p_{ij} \mid j \neq n[i] \text{ and } j \neq (b[i]_d - 1), d \in \mathbb{N}_{f[i]}\}$.
EST_i	An extended sub-tree of ST_j that is formed by $C_i, C_{i+1}, \dots, C_{j-2}, C_{j-1}$, and ST_j .
F	$= \{l \mid C_l \text{ is a fork tool}\}$.

Manuscript received April 28, 2017; accepted July 6, 2017. This work was supported in part by Science and Technology Development Fund (FDCT) of Macau (106/2016/A3), the National Natural Science Foundation of China (U1401240), and the Delta Electronics Inc and the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme. Recommended by Associate Editor Xiaou Li. (*Corresponding author: Fajun Yang.*)

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

F. J. Yang is with the Department of Industrial Engineering, School of Electro-Mechanical Engineering, Guangdong University of Technology, Guangzhou 510006, China, and he is also with School of Electrical and Electronic Engineering at Nanyang Technological University, 639798, Singapore (e-mail: fjiang@ntu.edu.sg).

N. Q. Wu and Y. Qiao are with the Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau 999078, China, and N. Q. Wu is also with the school of Electro-Mechanical Engineering, Guangdong University of Technology, Guangzhou 510006, China (e-mail: ngwu@must.edu.mo; yqiao@must.edu.mo).

R. Su is with the School of Electrical and Electronic Engineering at Nanyang Technological University, 639798, Singapore (e-mail: RSu@ntu.edu.sg).

Digital Object Identifier 10.1109/JAS.2017.7510772

$f[i]$	The number of outgoing module in $C_i, i \notin L$.
LB	Lower bound.
L	$= \{i \mid C_i \text{ is a leaf tool}\}$.
M	Marking for a Petri net.
$n[i]$	The index for the last step in C_i .
O^2CS	Optimal one-wafer cyclic schedule.
OSLB	One-wafer cyclic schedule with the lower bound of cycle time.
P	Set of places in a petri net.
p_{ij}	Place modeling step S_{ij} in a Petri net.
PN	Petri net.
S_{ij}	Processing step j in C_i .
q_{ij}	Place modeling R_i 's waiting before unloading a token from p_{ij} or during swap at p_{ij} for single and dual-arm tool in a PN, respectively.
R_i	The robot in C_i .
r_i	Place modeling R_i in a PN.
ST_i	A sub-tree that starts from a fork tool C_i .
S_k	$S_k = \{p \mid C_p \text{ is connected to } C_k\}$.
T	Set of transitions in a PN.
t_{ij}	Transition for R_i 's loading a token into p_{ij} in a PN.
u_{ij}	Transition for R_i 's unloading a token from p_{ij} in a PN.
x_{ij}	Transition for R_i 's moving from steps j to $j+1$ (or 0 if $j = n[i]$) with a wafer being held in a PN.
y_{ij}	Transition for R_i 's moving from steps $j+2$ to j (or from step 0 if $j = n[1]-1$, or step 1 if $j = n[1]$) without a wafer being held in a PN for a single-arm tool.
α_{ij}	Wafer processing time at S_{ij} .
$\mathcal{K}$	Capacity function in a PN.
$\mathbb{N}_n$	$= \{1, 2, \dots, n\}$.
π_i	The cycle time of C_i .
π_{ij}	The cycle time at S_{ij} .
θ_{ij}	Time taken for completing a wafer at S_{ij} .
λ_i	Time taken for R_i 's unloading/loading a wafer from/into a step for a single-arm tool or swap at a step for a dual-arm tool.
μ_i	Time taken for R_i 's moving from one step to another with or without a wafer being held.
ξ_{ij}	Time taken for completing a wafer with no robot waiting at S_{ij} .
Π	The lower bound of the cycle time for the system.
Θ	$\geq \Pi$, the cycle time for the entire system.
Π_i	The fundamental period of C_i .
τ_{ij}	Wafer sojourn time at S_{ij} .
ψ_i	R_i 's cycle time.
ψ_{i1}	R_i 's cycle time without robot waiting.
ψ_{i2}	R_i 's waiting time in a robot cycle.
Ω_n	$\Omega_n = \{0\} \cup \mathbb{N}_n$.

ω_{ij}	R_i 's waiting time before unloading a wafer from step $j, j \in \Omega_{n[i]}$.
ω_{ij1}	R_i 's waiting time during swap at p_{ij} in a dual-arm tool C_i .

I. INTRODUCTION

A cluster tool in wafer fabrication integrates a robot, two loadlocks, and a few process modules. It is called single or dual-arm cluster tool if its robot has one or two arms. Through the loadlocks, raw wafers are loaded into the system in a cassette-by-cassette way. They are processed by process modules in a pre-specified order at each step and finally return to the loadlock where it came from [1].

Multi-cluster tools are composed of several individual cluster tools connected by buffering modules (BMs) with a linear or treelike topology [2]. It is called a treelike hybrid K -cluster tool if it contains $K (\geq 2)$ individual cluster tools including both single and dual-arm tools with a treelike topology as shown in Fig.1, where the red line represents the wafer processing route.

Fig. 1. The illustration of a treelike hybrid 5-cluster tool.

Compared with the scheduling problem of complex industrial processes [3]–[8], it seems that scheduling a single cluster tool is simple. However, many factors make the scheduling problem of cluster tools very complicated. Thus, extensive research efforts have been made for effective operation of cluster tools [9]–[18]. For a detailed review in this field, the readers can refer as to [19]. They show that a cluster tool mostly operates under a steady state in a process or transport-bound region. In practice, the robot task time is much shorter than wafer processing time [20] such that it is process-bound. In such a case, for dual-arm tools, a swap strategy is effective [12], while for single-arm ones, a backward strategy is optimal [21], [22].

To schedule multi-cluster tools, Jevtic and Venkatesh [23] proposes a heuristic method. By ignoring the robot moving time, Ding *et al.* [24] find an optimal periodical schedule by using an event graph (a special class of Petri nets) combined with a network model. Yi *et al.* [25] give a decomposition method. By considering the robot moving time, Chan *et al.*

[26], [27] propose a method with polynomial complexity to find an optimal cyclic schedule for both single-arm 2-cluster tools and a treelike M -cluster tool. Generally, their method can successfully find a multi-wafer cyclic schedule, i.e., more than one wafer is produced in each period [11]. For a cluster photolithography tool with multi-robot that can process a few different wafers at the same time, [28] investigates how incremental scheduling, a technique based on prioritized planning, can be used to schedule the system.

Due to its easy implementation, a one-wafer cyclic one is most desired by practitioners if it achieves the same maximum throughput as a multi-wafer cyclic schedule does. A multi-cluster tool with its bottleneck individual tool being process-bound is said to be process-dominant [29], [30]. It is shown that, for a process-dominant single-arm multi-cluster tool with a linear topology, an optimal one-wafer cyclic schedule (O²CS) can be always found by an efficient method [29]. The methods for finding an O²CS for process-dominant linear hybrid multi-cluster tools are given in [31], [32]. The capacity of the buffers linking the individual cluster tools has effect on the productivity of a multi-cluster tool. Bai *et al.* [33] present a method to optimize the configuration of buffer capacity such that a one-wafer cyclic schedule with the lower bound of cycle time can be found for multi-cluster tools.

Scheduling a treelike hybrid multi-cluster tool is much more complex than a linear one and thus more challenging. Yang *et al.* [34] present the conditions under which a one-wafer cyclic schedule with the lower bound (LB) of cycle time (OSLB) exists for such a tool and an efficient algorithm to find it if existing. However, if the conditions are not met, it is open whether there is an O²CS (its cycle time must be longer than its LB) and how it can be found if so. To find an O²CS for treelike hybrid multi-cluster tool, it needs to coordinate the behavior of the multiple robots, which is very challenging. Based on the results from [34], this work aims to answer this problem for the first time.

Since optimal cycle time implies the maximal productivity, an O²CS should always be the priority for scheduling a multi-cluster tool. To schedule a multi-cluster tool is to allocate the finite resources to its process modules dynamically. Since a finite capacity Petri net (PN) is an effective tool for modeling resource allocation, we develop a PN model for such a tool based on the finite capacity PN. With the model, this work makes the following contribution:

1) It proves that there is always a one-wafer cyclic schedule for a treelike hybrid multi-cluster tool.

2) It develops computationally efficient algorithms to find an O²CS by simply setting the robot waiting time properly.

Next section presents the PN model for treelike hybrid multi-cluster tools. Section III uses the model to analyze the dynamic behavior of individual tools. Section IV presents the existence of a one-wafer cyclic schedule and a method to find an O²CS. Section V shows the applications of the proposed method by examples. Finally, Section VI concludes this paper.

II. PETRI NET MODELING

We make the same assumption as those in [26], [27] as given in the Supplementary File. Let $\mathbb{N}_n = \{1, 2, \dots, n\}$ and

$\Omega_n = \{0\} \cup \mathbb{N}_n$. C_i and $R_i, i \in \mathbb{N}_K$, denote the i th tool and its robot, respectively. C_1 with loadlocks in it is called the head one, $C_i, i \neq 1$, is called a leaf one if it connects one tool only, and it is a fork tool if it connects at least three adjacent tools. For example, given the tool in Fig. 1, C_3 and C_5 are leaf ones; while C_2 is fork one. Let $L = \{i \mid C_i \text{ is a leaf tool}\}$ be the set of indices for the leaf tools. For easy presentation, the tools are numbered such that, for any two adjacent tools C_k and $C_i, k \notin L$ and $i \in \mathbb{N}_K \setminus \{1\}$, with C_k/C_i being the upstream/downstream one, we have $i > k$ [27]. However, k and i are not necessarily in a consecutive order as shown in Fig. 1.

A BM connecting C_k and C_i is seen as the outgoing module for C_k and incoming one numbered as Step 0 for C_i . Let $n[i]$ be the index for the last step in $C_i, i \in \mathbb{N}_K$. Then, there are $n[i] + 1$ steps in C_i including outgoing and incoming modules. Let $f[i] \in \mathbb{N}_{n[i]}$ denote the number of outgoing modules in $C_i, i \notin L$. If $f[i] > 1$, C_i is a fork. Let S_{ij} denote Step j (except BM(s)) in C_i with S_{10} for the loadlocks in C_1 , and $S_{i(b[i]_1)}, S_{i(b[i]_2)}, \dots$, and $S_{i(b[i]_f[i])}$ with $b[i]_1 < b[i]_2 < \dots < b[i]_f[i]$ denote outgoing modules. Note that $b[i]_1, b[i]_2, \dots$, and $b[i]_f[i]$ are not necessarily in a consecutive order. The incoming module for C_i is denoted as S_{i0} . In this way, the $n[i] + 1$ steps in C_i are denoted as $S_{i0}, S_{i1}, \dots, S_{i(b[i]_1)}, \dots, S_{i(b[i]_2)}, \dots$, and $S_{i(n[i])}$, respectively. Note that $b[i]_1 = 1$ if it is Step 1 and $b[i]_f[i] = n[i]$ if it is the last step.

Petri nets are a kind of effective tools for modeling, analysis, scheduling, and control for discrete event systems [4], [5], [35]–[38]. This work extends the resource-oriented PN [37], [38] to model the system. It is defined as $\text{PN} = (P, T, I, O, M, \mathcal{K})$, where P and T are finite sets of places and transitions; I and O are the input and output functions; $M(p)$ is a marking representing the number of tokens in place p with $M_0(p)$ being the initial marking; and $\mathcal{K}$ is a capacity function. For the transition enabling and firing rules, readers can refer to [37], [38].

A. PN for Hybrid K-Cluster Tools

This work adopts the PN models developed in [32] and a brief introduction to them is presented for self-completeness. For convenience, a token and a wafer are often used without any difference.

For a single-arm or dual-arm $C_i, i \in \mathbb{N}_K$, transition $x_{ij}, j \in \Omega_{n[i]}$, models R_i 's moving from Steps j to $j + 1$ (or Step 0 if $j = n[i]$) with a wafer, places r_i and p_{ij} model R_i and $S_{ij}, j \in \Omega_{n[i]}$, respectively, with $\mathcal{K}(p_{ij}) = 1$. If C_i is a single-arm tool, $\mathcal{K}(r_i) = 1$; otherwise $\mathcal{K}(r_i) = 2$. For a single-arm C_i shown in Fig. 2, q_{ij} with $\mathcal{K}(q_{ij}) = 1$ models R_i 's waiting for unloading a wafer from $p_{ij}, j \in \Omega_{n[i]}$. d_{ij} and z_{ij} with $\mathcal{K}(d_{ij}) = \mathcal{K}(z_{ij}) = 1$ model that R_i holds a wafer for moving from Steps j to $j + 1$ (or Step 0 if $j = n[i]$) and loading a wafer into $p_{ij}, j \in \Omega_{n[i]}$, respectively. u_{ij} and t_{ij} model that R_i removes a wafer from p_{ij} and drops a wafer into $p_{ij}, j \in \Omega_{n[i]}$, respectively. $y_{ij}, j \in \Omega_{n[i]} \setminus \{0, 1\}$, models R_i 's moving from Steps $j + 2$ to j (or to $n[i] - 1$ if $j = 0$, or to $n[i]$ if $j = 1$) without carrying a wafer.

Fig. 2. A PN model for a single-arm tool C_i .

Fig. 3. A PN model for a dual-arm tool C_i .

For a dual-arm C_i shown in Fig. 3, d_{ij} and z_{ij} with $\mathcal{K}(d_{ij}) = \mathcal{K}(z_{ij}) = 1$ model the state that a swap ends and R_i 's waiting for unloading a wafer from $p_{ij}, j \in \Omega_{n[i]}$, respectively. q_{ij} with $\mathcal{K}(q_{ij}) = 2$ models that both arms of R_i hold a wafer and wait during the swap at $p_{ij}, j \in \Omega_{n[i]}$. t_{ij} and u_{ij} model R_i 's loading and unloading a wafer into and from $p_{ij}, j \in \Omega_{n[i]}$, respectively.

For a BM connecting C_k and $C_i, k \notin L, i \neq 1$, with C_k being the upstream one, there are four different cases: 1) both are dual-arm tools (D-D in short); 2) C_k is a single-arm tool and C_i a dual-arm one (S-D); 3) C_k is a dual-arm tool and C_i a single-arm one (D-S); and 4) both are single-arm tools (S-S).

A BM is treated as a step or the q th outgoing module denoted by $S_{k(b[k]_q)}, 1 \leq q \leq f[k]$, for C_k and Step 0 denoted as S_{i0} for C_i . It is modeled by $p_{k(b[k]_q)}$ and p_{i0} for C_k and C_i with $p_{k(b[k]_q)} = p_{i0}$ and $\mathcal{K}(p_{k(b[k]_q)}) = \mathcal{K}(p_{i0}) = 1$. The PN models for the four cases are shown in Fig. 4. Taking the S-S case as an example, Step $b[k]_q$ for C_k is modeled by $z_{k(b[k]_q)}, t_{k(b[k]_q)}, p_{k(b[k]_q)}, q_{k(b[k]_q)}, u_{k(b[k]_q)}$, and $d_{k(b[k]_q)}$ together with arcs $(z_{k(b[k]_q)}, t_{k(b[k]_q)}), (t_{k(b[k]_q)}, p_{k(b[k]_q)}), (p_{k(b[k]_q)}, u_{k(b[k]_q)}), (q_{k(b[k]_q)}, u_{k(b[k]_q)}),$ and $(u_{k(b[k]_q)}, d_{k(b[k]_q)})$. Step 0 for C_i is modeled by $z_{i0}, t_{i0}, p_{i0}, q_{i0}, u_{i0}$, and d_{i0} together with arcs $(z_{i0}, t_{i0}), (t_{i0}, p_{i0}), (p_{i0}, u_{i0}), (q_{i0}, u_{i0}),$ and (u_{i0}, d_{i0}) as shown in Fig. 4(d).

Initial marking M_0 can be set by using V -tokens representing virtual wafers to describe the startup transient process as

and

$$\pi_{i0} = \tau_{i0} + 4\lambda_i + 3\mu_i + \omega_{i(n[i])}. \quad (5)$$

Its robot cycle time is

$$\psi_i = 2(n[i] + 1)(\lambda_i + \mu_i) + \sum_{j=0}^{n[i]} \omega_{ij} = \psi_{i1} + \psi_{i2} \quad (6)$$

where $\psi_{i1} = 2(n[i] + 1)(\lambda_i + \mu_i)$ is the robot's activity time in a cycle and is a constant, while $\psi_{i2} = \sum_{j=0}^{n[i]} \omega_{ij}$ is the robot waiting time in a cycle.

For a dual-arm C_i , according to Wu and Zhou [1], we have the time needed for processing a wafer at Step $j, j \in \Omega_{n[i]}$

$$\xi_{ij} = \alpha_{ij} + \lambda_i. \quad (7)$$

Similarly, by replacing α_{ij} with τ_{ij} in (7), we have

$$\pi_{ij} = \tau_{ij} + \lambda_i. \quad (8)$$

Its robot cycle time is

$$\psi_i = (n[i] + 1)(\lambda_i + \mu_i) + \sum_{j=0}^{n[i]} \omega_{ij} = \psi_{i1} + \psi_{i2} \quad (9)$$

where ψ_{i1} and ψ_{i2} have the same meaning as a single-arm tool.

For each tool $C_i, i \in \mathbb{N}_K$, the productivity for each step is identical and C_i should be scheduled such that

$$\pi_i = \pi_{i0} = \pi_{i1} = \dots = \pi_{i(n[i])} = \psi_i. \quad (10)$$

Since both π_i and ψ_i are functions of ω_{ij} 's, the schedule for each tool $C_i, i \in \mathbb{N}_K$, is parameterized by robots' waiting time. Hence, the key for scheduling a treelike hybrid K -cluster tool is to determine ω_{ij} 's such that the activities of the multiple robots are coordinated to act in a paced way.

IV. SCHEDULING THE OVERALL SYSTEM

A. Schedule Properties

Let $\Pi_i = \max\{\xi_{i0}, \xi_{i1}, \dots, \xi_{i(n[i])}, \psi_{i1}\}$ denote the fundamental period (FP) of $C_i, i \in \mathbb{N}_K$. Further, let $\Pi = \max\{\Pi_1, \Pi_2, \dots, \Pi_K\} = \Pi_h, 1 \leq h \leq K$, or C_h is the bottleneck tool. For a process-dominant treelike hybrid K -cluster tool, C_h must be process-bound. Let Θ be cycle time for the system. Then, with π_i being the cycle time of C_i , to obtain a one-wafer cyclic schedule, each individual tool must have an identical cycle time, i.e.,

$$\Theta = \pi_i \geq \Pi, \quad \forall i \in \mathbb{N}_K. \quad (11)$$

From [32], all the individual tools are scheduled to be paced, if and only if, for any adjacent pair C_k and $C_i, k \notin L$ and $i \neq 1$, linked by $S_{k(b[k]_q)}, 1 \leq q \leq f[k]$, at any M , we have: 1) whenever $R_i(R_k)$ is scheduled to load a token into $p_{i0}(p_{k(b[k]_q)}), M(p_{i0}) = 0$ ($M(p_{k(b[k]_q)}) = 0$); and 2) whenever $R_i(R_k)$ is scheduled to unload a token from $p_{i0}(p_{k(b[k]_q)}), M(p_{i0}) = 1$ ($M(p_{k(b[k]_q)}) = 1$). In [32], the existence of an OSLB is discussed and an algorithm is developed to find it if existing. For the case that there is no OSLB, this work answers how to find an O²CS with the minimum cycle time $\Theta > \Pi$.

B. Optimal One-wafer Cyclic Scheduling

It has been proved in [31] that for the D-D and S-D cases, there is always a cyclic schedule for any adjacent pair C_k and C_i such that their cycle time reaches the LB. Hence, for a process-dominant treelike hybrid K -cluster tool, the conditions under which an OSLB exists are given as follows [34].

Lemma 1: For a process-dominant treelike hybrid K -cluster tool, an OSLB exists, if and only if, for any adjacent pair C_k and $C_i, k \notin L$ and $i \neq 1$, connected by $S_{k(b[k]_q)}$, the following conditions are satisfied by determining ω_{kj} 's and ω_{il} 's, $j \in \Omega_{n[k]}$ and $l \in \Omega_{n[i]}$

$$\pi_{kj} = \pi_{il} = \Pi, \quad j \in \Omega_{n[k]} \text{ and } l \in \Omega_{n[i]}. \quad (12)$$

If C_k and C_i are D-S case

$$\Pi - \lambda_k \geq 4\lambda_i + 3\mu_i + \omega_{i(n[i])}. \quad (13)$$

If C_k and C_i are S-S case

$$\Pi - (4\lambda_k + 3\mu_k + \omega_{k((b[k]_q)-1)}) \geq 4\lambda_i + 3\mu_i + \omega_{i(n[i])}. \quad (14)$$

Based on Lemma 1, an efficient method is developed in [32] to test the existence of an OSLB and find it if existing. Next, for such a system, we show that there is always a one-wafer cyclic schedule and present an efficient algorithm to find the minimal cycle time Θ and O²CS. Now we focus on the cases in which there is no OSLB.

Let B_i denote a linear multi-cluster tool in a treelike K -cluster tool called a branch that starts from C_i and ends at $C_m, m \in L$. For easy presentation, the tools in B_i are labeled consecutively as $C_i, C_{i+1}, \dots$, and C_m . Assume that, with $\Theta = \Pi$ as cycle time, the conditions given in Lemma 1 are violated for a pair of C_i and C_{i+1} in B_i . Let $A_{kj} = \omega_{kj}, k \in \{i+1, i+2, \dots, m\}$ and $j \in \Omega_{n[k]}$, where ω_{kj} is R_k 's waiting time obtained by the algorithm in [32] with cycle time $\Theta = \Pi$. Let $\Phi_{i+1}(S, S) = 4\lambda_{i+1} + 3\mu_{i+1} + A_{(i+1)(n[i+1])} - \Theta + 4\lambda_i + 3\mu_i$, when C_i and C_{i+1} are S-S; and $\Phi_{i+1}(D, S) = 4\lambda_{i+1} + 3\mu_{i+1} + A_{(i+1)(n[i+1])} - \Theta + \lambda_i$, when C_i and C_{i+1} are D-S. If $A_{(i+1)(n[i+1])} = 0$, it follows from [32] that an O²CS with cycle time $\Theta = \Pi + \Phi_{i+1}(S, S)$ (or $\Pi + \Phi_{i+1}(D, S)$ if C_i is a dual-arm tool) can be found for B_i . Otherwise, if $A_{(i+1)(n[i+1])} \neq 0$, assume that, for $g \in [i+1, m], A_{(g+1)(n[g+1])} > 0, i \leq k \leq g-1$, and $A_{(g+1)(n[g+1])} = 0$, or C_{g+1} is a dual-arm one, or $g \in L$. Define $D[k] = \{p_{kj} \mid j \neq (b[k]_1)-1 \text{ and } j \neq n[k]\}$ as the set of steps in C_k , and $B[k] = n[k]-1, k \in \{i+1, i+2, \dots, m\}$. Then, let $\Delta_g = A_{g(n[g])}/B[g], \Delta_{g-1} = A_{(g-1)(n[g-1])}/(B[g-1] + B[g]), \Delta_{g-2} = A_{(g-2)(n[g-2])}/(B[g-2] + B[g-1] + B[g]), \dots, \Delta_{i+2} = A_{(i+2)(n[i+2])}/(B[i+2] + B[i+3] + \dots + B[g])$, and define Δ_{i+1} as follows.

$$\Delta_{i+1} = \begin{cases} \Phi_{i+1}(S, S) - A_{(i+1)(n[i+1])} \\ \quad \text{if } A_{(i+1)(n[i+1])} < \Gamma \times \frac{\Phi_{i+1}(S, S)}{(\Gamma+1)} \\ \Phi_{i+1}(S, S)/(\Gamma+1) \\ \quad \text{if } A_{(i+1)(n[i+1])} \geq \Gamma \times \frac{\Phi_{i+1}(S, S)}{(\Gamma+1)} \end{cases} \quad (15)$$

if C_i is a single-arm tool, where $\Gamma = B[i+1] + B[i+2] + \dots + B[g]$.

If C_i is a dual-arm one, $\Delta_{i+1} = \Phi_{i+1}(D, S)/(\Gamma + 1)$. Then, from [32], we have the following results.

Lemma 2 [32]: For B_i that is composed of two tools, there is an O²CS and its minimal one-wafer cycle time is $\Theta = \Pi + \Delta$, where

$$\Delta = \begin{cases} \Phi_2(S, S) - A_{2(n[2])} \\ \quad \text{if } A_{2(n[2])} \in [0, (n[2] - 1) \times \frac{\Phi_2(S, S)}{n[2]}) \text{ for S-S} \\ \frac{\Phi_2(S, S)}{n[2]} \\ \quad \text{if } A_{2(n[2])} \in [(n[2] - 1) \times \frac{\Phi_2(S, S)}{n[2]}, \infty) \text{ for S-S} \\ \frac{\Phi_2(D, S)}{n[2]} \text{ for D-S.} \end{cases} \quad (16)$$

With Lemma 2, a method is proposed to find the minimal cycle time and O²CS for B_i in [32].

Lemma 3 [32]: For B_i in a treelike hybrid K -cluster tool, if a one-wafer cyclic schedule with cycle time $\Theta (\geq \Pi)$ is found, a one-wafer cyclic schedule with cycle time $(\Theta + \Delta)$ obtained by setting $\omega_{g((b[g]_1)-1)} = A_{g((b[g]_1)-1)} + \Delta, g \in \{i, i+1, \dots, m-1\}$, and $\omega_{m0} = A_{m0} + \Delta, m \in L$, is feasible.

Let ST_i denote a sub-tree in a tool that starts from a fork tool C_i . Assume that ST_i and ST_j are two STs, and ST_i is contained by ST_j . Such a relation is denoted as $ST_i < ST_j$ or $ST_j > ST_i$. The individual tools can be numbered such that $ST_j > ST_i$ only if $j < i$. However, $j < i$ does not necessarily imply $ST_j > ST_i$, for they may be independent of each other. Such a relation is denoted as $ST_j \vee ST_i$. Next, we discuss the one-wafer cyclic scheduling problem and we have the following result.

Theorem 1: For an ST_i composed of three tools, there is an O²CS and its minimal cycle time is $\Theta = \Pi + \max\{\Delta_1, \Delta_2\}$, where Δ_1 and Δ_2 are obtained in Lemma 2.

Proof: See the supplementary file. ■

The ST discussed in Theorem 1 represents the simplest case. A fork tool C_i with $f[i]$ adjacent downstream tools $C_{i-1}, C_{i-2}, \dots$, and $C_{i-f[i]}$, may contain no ST but only branches denoted by $B_{i-1}, B_{i-2}, \dots, B_{i-f[i]}$. By the method in [32], one can find O²CSs for $B_{i-q}, q \in \mathbb{N}_{f[i]}$. Assume that the cycle time for a branch composed of C_i and B_{i-q} is $\Pi + \Delta_q$. Then, based on Theorem 1, we have the following corollary.

Corollary 1: For an ST_i composed of C_i and its downstream branches $B_{i-1}, B_{i-2}, \dots$, and $B_{i-f[i]}$, there is an O²CS and its minimal one-wafer cycle time is $\Theta = \Pi + \max\{\Delta_1, \Delta_2, \dots, \Delta_{f[i]}\}$, where $\Pi + \Delta_q$ is the cycle time for the branch composed of C_i and B_{i-q} .

Then, an O²CS for a treelike hybrid K -cluster tool can be found as follows. Let C_j be a fork tool. There must be its ST_j . Assume that there are tools $C_i, C_{i+1}, \dots, C_{j-2}$, and C_{j-1} , with none being a fork tool. We call the multi-cluster tool formed by $C_i, C_{i+1}, \dots, C_{j-2}, C_{j-1}$, and ST_j an extended sub-tree of ST_j , denoted as EST_i . The difference between ST_j and EST_i is that C_j in ST_j must be a fork tool while C_i in EST_i is not a fork tool. Note that there may be more than one EST of ST_j . Let $F = \{l \mid C_l \text{ is a fork tool}\}$ be the set of indices for all fork tools. Let $j = \max_{l \in F} \{l\}$. Then, ST_j must contain no ST and its O²CS can be found based on Corollary

1. With an O²CS for ST_j found, one needs to find O²CSs for all its ESTs. Then, let $F \leftarrow F \setminus \{j\}$. If $F \neq \emptyset$, find $h = \max_{l \in F} \{l\}$. If $ST_h \vee ST_j$, it can be dealt with just as ST_j , otherwise, we have $ST_h > ST_j$. In this case, we need to find an O²CS for ST_h and all its ESTs. Then, let $F \leftarrow F \setminus \{h\}$ and repeat the above process. In this way, we find O²CSs for all STs and their corresponding ESTs. When C_1 is reached, an O²CS for the whole system is obtained. Hence, the key is how to find an O²CS for an ST and its EST. By the definition of EST_k , if B_{i-q} in Corollary 1 is replaced by EST_{i-q} , the result must also hold.

Next, we first discuss how to find an O²CS for an EST that contains one fork tool only and then present a method to find an O²CS for the whole system. Assume that $j = \max_{l \in F} \{l\}$ and EST_i is an extended sub-tree of ST_j . Furthermore, assume that, for tool pair C_i and $C_{i+1}, i+1 \leq j$, there is no one-wafer cyclic schedule with cycle time Θ . Note that, with the O²CS for ST_j found based on Corollary 1, Condition (13) or (14) is violated for pair C_i and $C_{i+1}, i+1 \leq j$, only.

Let $A_{pj} = \omega_{pj}, p \in \mathbb{N}_K \setminus \mathbb{N}_i$ and $j \in \Omega_{n[p]}$, where ω_{pj} 's are obtained by the algorithm in [32] with cycle time Θ . Define $D[l] = \{p_{lj} \mid j \neq n[l] \text{ and } j \neq (b[l]_d) - 1, d \in \mathbb{N}_{f[l]}\}$ and $B[l] = n[l] - 1, l \in \mathbb{N}_K$. Let $\Phi_{i+1}(S, S) = 4\lambda_{i+1} + 3\mu_{i+1} + A_{(i+1)(n[i+1])} - \Theta + 4\lambda_i + 3\mu_i$ for the S-S case and $\Phi_{i+1}(D, S) = 4\lambda_{i+1} + 3\mu_{i+1} + A_{(i+1)(n[i+1])} - \Theta + \lambda_i$ for the D-S case. It follows from [32] that, if $A_{(i+1)(n[i+1])} = 0$, to make (13) or (14) satisfied for C_i and C_{i+1} , Θ needs to be increased by $\Phi_{i+1}(D, S)$ or $\Phi_{i+1}(S, S)$ time units. If $A_{(i+1)(n[i+1])} \neq 0$, we need to check if $A_{j(n[j])} = 0$. If so, we find the optimal cycle time via the methods in [32]. If $A_{j(n[j])} \neq 0$ and there is a dual-arm tool in tools $C_{i+1}, C_{i+2}, \dots$, and C_j , or there is at least one $A_{p(n[p])} = 0, i+1 < p < j$. In this case, to find the optimal cycle time, we need to examine tools $C_i, C_{i+1}, \dots$, and C_p only. Since the system formed by these tools is a multi-cluster tool with linear topology, we can still find the optimal cycle time via the methods in [32]. However, when $A_{p(n[p])} > 0, i+1 \leq p \leq j$, we need to examine all the downstream tools of C_i and the problem is much more complicated. Hence, we cannot find the optimal cycle time by using the method in [32]. We discuss how to find it next.

Assume that C_j has $f[j]$ downstream tools such that ST_j has $f[j]$ branches. For C_{j-q} in $B_{j-q}, q \in \mathbb{N}_{f[j]}$, we first check whether $A_{(j-q)(n[j-q])} > 0$. If so, find g such that $A_{((j-q)+z)(n[(j-q)+z])} > 0, 0 \leq z \leq g$, and $A_{((j-q)+z+1)(n[(j-q)+z+1])} = 0$ or $C_{(j-q)+g+1}$ is a dual-arm one, or $(j-q) + g \in L$. Let

$$\begin{aligned} \Delta_{((j-q)+g)} &= \frac{A_{((j-q)+g)(n[(j-q)+g])}}{B[(j-q) + g]} \\ \Delta_{((j-q)+g-1)} &= \frac{A_{((j-q)+g-1)(n[(j-q)+g-1])}}{B[(j-q) + g - 1] + B[(j-q) + g]} \\ \Delta_{((j-q)+g-2)} &= \frac{A_{((j-q)+g-2)(n[(j-q)+g-2])}}{B[(j-q)+g-2] + B[(j-q)+g-1] + B[(j-q)+g]} \\ &\dots \\ \Delta_{(j-q)} &= \frac{A_{(j-q)(n[j-q])}}{B[j-q] + B[(j-q) + 1] + \dots + B[(j-q) + g]}. \end{aligned}$$

Further, let $\Upsilon_{j-q} = B[j-q] + B[(j-q) + 1] + \dots + B[(j-q) +$

$g], q \in \mathbb{N}_{f[j]}$, and $\Upsilon_j = \Upsilon_{j-1} + \Upsilon_{j-2} + \dots + \Upsilon_{j-f[j]}$. Note that, if $A_{(j-q)(n[j-q])} = 0$, we have $\Upsilon_{j-q} = 0$. Then, define $\Delta_j = A_{j(n[j])}/(\Upsilon_j + B[j])$, $\Delta_{j-1} = A_{(j-1)(n[j-1])}/(\Upsilon_j + B[j] + B[j-1])$, $\dots$, and $\Delta_{i+2} = A_{(i+2)(n[i+2])}/(\Upsilon_j + B[j] + B[j-1] + \dots + B[i+2])$. Let $\Lambda = \Upsilon_j + B[j] + B[j-1] + \dots + B[i+1]$, $\Delta = \min\{\Delta_{i+1}, \Delta_{i+2}, \dots, \Delta_j, \Delta_{(j-q)+z}\}$ and $\kappa = \arg \min\{\Delta_{i+1}, \Delta_{i+2}, \dots, \Delta_j, \Delta_{(j-q)+z}\}$ with $q \in \mathbb{N}_{f[j]}$ and $0 \leq z \leq g$, where

$$\Delta_{i+1} = \begin{cases} \Phi_{i+1}(S, S) - A_{(i+1)(n[i+1])} \\ \quad \text{if } A_{(i+1)(n[i+1])} \in (0, \Lambda \times \frac{\Phi_{i+1}(S, S)}{(\Lambda + 1)}) \text{ for S-S} \\ \frac{\Phi_{i+1}(S, S)}{(\Lambda + 1)} \\ \quad \text{if } A_{(i+1)(n[i+1])} \in [\Lambda \times \frac{\Phi_{i+1}(S, S)}{(\Lambda + 1)}, \infty) \text{ for S-S} \\ \frac{\Phi_{i+1}(D, S)}{(\Lambda + 1)} \text{ for D-S} \end{cases} \quad (17)$$

Then, we have the following result.

Theorem 2: Assume that: 1) $j = \max_{l \in F}\{l\}$; 2) for an EST_i of ST_j , by the algorithm in [32] with cycle time θ , Condition (14) is violated for pair C_i and C_{i+1} , $i+1 \leq j$, only; and 3) $\Delta = \Delta_\kappa = \Delta_{i+1}$. Then, an O^2CS with cycle time $\theta + \Delta$ can be found for EST_i , where Δ is given by (17).

Proof: See the supplementary file. ■

By Theorem 2, with the cycle time being increased by Δ , it guarantees that when R_i is scheduled to unload a wafer from $p_{i(b[i])}(p_{(i+1)0})$, R_{i+1} has just loaded a wafer there. Detailed explanation can be seen in the Supplementary File. If C_i and C_{i+1} are D-S, we have the following result.

Theorem 3: Assume that: 1) $j = \max_{l \in F}\{l\}$; 2) for an EST_i of ST_j , by the algorithm in [32] with cycle time θ , Condition (13) is violated for pair C_i and C_{i+1} , $i+1 \leq j$, only; and 3) $\Delta = \Delta_\kappa = \Delta_{i+1}$. Then, an O^2CS with cycle time $\theta + \Delta$ can be found for EST_i , where Δ is given by (17).

Proof: See the supplementary file. ■

The idea of Theorem 3 can be similarly shown as that of Theorem 2 and its detail explanation can be seen in the Supplementary File. In Theorems 2 and 3, we assume that $\kappa = i+1$. When $\kappa = \arg \min\{\Delta_{i+1}, \Delta_{i+2}, \dots, \Delta_j, \Delta_{(j-q)+z}\} \neq i+1$, $q \in \mathbb{N}_{f[j]}$ and $0 \leq z \leq g$, the following theorem presents how to find an O^2CS for EST_i .

Theorem 4: Assume that: 1) $j = \max_{l \in F}\{l\}$; 2) in EST_i of ST_j , by the algorithm in [32] with cycle time θ , Condition (13) or (14) is violated for tool pair C_i and C_{i+1} , $i+1 \leq j$, only; and 3) $\Delta = \Delta_\kappa \neq \Delta_{i+1}$. Then, there is an O^2CS with cycle time $\theta + \Delta$ for EST_i , where Δ is defined in (17).

Proof: See the supplementary file. ■

By Theorems 2 – 4, to find an O^2CS for an EST_i of ST_j , we require j to be the largest one in F , and such an EST_i contains one fork tool only. This is not enough for one to find an O^2CS for the entire system. However, based on Theorems 2 – 4, one can obtain an O^2CS for arbitrary EST_k as follows. Assume that, for an EST_k that contains more than one fork tool, Condition (13) or (14) is violated for C_k and C_i only, with C_i being the downstream adjacent tool of C_k . A path from C_k to a downstream C_l is called a shortest path if the number of nodes (individual tools) on the path is minimal. Then, for an EST_k , we say that C_l is “connected” to C_k if: 1) there are only single-arm tools on the shortest path from C_k to C_l ; and 2) for each tool C_p on the path, $\omega_{p(n[p])}'s > 0$. Note that C_k is connected to itself. Let $S_k = \{p | C_p \text{ is connected to } C_k\}$. Furthermore, as done above, let $\Phi_i(S, S) = 4\lambda_i + 3\mu_i + A_{i(n[i])} - \theta + 4\lambda_k + 3\mu_k$ for the S-S case and $\Phi_i(D, S) = 4\lambda_i + 3\mu_i + A_{i(n[i])} - \theta + \lambda_k$ for the D-S case. Then, for any EST_k with C_i being the downstream adjacent tool of C_k , to find the minimal one-wafer cycle time, we compute Δ_m as follows.

$$\Delta_m = \frac{A_{m(n[m])}}{\sum_{p \in S_m} B[p]}, \quad m > i \quad (18)$$

and if $m = i$, it is calculated by (19), shown at the bottom of this page.

Clearly, (19) is the extension of (17). Thus, based on (18) and (19), we can find an O^2CS for any EST_k if (13) or (14) is violated for C_k and C_i only. Then, the O^2CS for an ST_j that contains a number of EST_k 's can be obtained by using Corollary 1. We present the following algorithm.

Algorithm 1: Given an O^2CS with cycle time θ for EST_i (or ST_i , or B_i), find an O^2CS for EST_k (or ST_k) with C_i being the downstream adjacent tool of C_k .

Step 1: Given cycle time θ , check if θ is the optimal cycle time for EST_k (or ST_k) by the algorithm in [32]. If yes, determine the schedule as in [32] and go to Step 5;

Step 2: If $k \notin F$ and $A_{i(n[i])} = 0$,

Step 2.1: $\theta \leftarrow \theta + \Delta_i = \theta + \Phi_i(S, S)$ (or $\Phi_i(D, S)$);

Step 2.2: Reset the robots' waiting time by the algorithm in [32] with cycle time θ , then go to Step 5.

Step 3: If $k \notin F$ and $A_{i(n[i])} \neq 0$,

Step 3.1: For $m \in S_i$, calculate Δ_m 's via (18) or (19);

Step 3.2: If $\Delta = \min\{\Delta_p | p \in S_i\} = \Delta_i$, do:

Step 3.2.1: If it belongs to the first case in (19), then

$$\Delta_m = \Delta_i = \begin{cases} \Phi_i(S, S) - A_{i(n[i])}, \text{ if } A_{i(n[i])} \in (0, \sum_{p \in S_i} B[p] \times \frac{\Phi_i(S, S)}{\sum_{p \in S_i} B[p] + 1}) \text{ for S-S case} \\ \frac{\Phi_i(S, S)}{\sum_{p \in S_i} B[p] + 1}, \text{ if } A_{(i+1)(n[i+1])} \in [\sum_{p \in S_i} B[p] \times \frac{\Phi_i(S, S)}{\sum_{p \in S_i} B[p] + 1}, \infty) \text{ for S-S case} \\ \frac{\Phi_i(D, S)}{\sum_{p \in S_i} B[p] + 1}, \text{ for D-S case} \end{cases} \quad (19)$$

- 1) In EST_{*i*} or B_{*i*}, for $p \notin S_i$, if $p \notin L$,
 $\omega_{p((b[p]_1)-1)} = A_{p((b[p]_1)-1)} + \Delta$, otherwise
 $\omega_{p0} = A_{p0} + \Delta$;
- 2) In EST_{*i*} or B_{*i*}, if $p \in S_i$ and $p \notin F$, set

$$\omega_{pj} = A_{pj} + \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]}, j \in D[p]$$

(or $j \in \mathbb{N}_{n[p]} \setminus \{n[p]\}$ if $p \in L$)

$$\omega_{p((b[p]_1)-1)} = A_{p((b[p]_1)-1)} + \sum_{q \in S_p \setminus \{p\}} B[q] \times \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]} + \Delta$$

(or $\omega_{p0} = A_{p0} + \Delta$ if $p \in L$), and

$$\omega_{p(n[p])} = A_{p(n[p])} - \sum_{q \in S_p} B[q] \times \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]};$$

- 3) In EST_{*i*}, if $p \in S_i$ and $p \in F$, set

$$\omega_{pj} = A_{pj} + \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]}, j \in D[p]$$

$$\omega_{p((b[p]_1)-1)} = A_{p((b[p]_1)-1)} + \sum_{q \in S_{p-1}} B[q] \times \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]} + \Delta$$

$$\omega_{p((b[p]_d)-1)} = A_{p((b[p]_d)-1)} + \left(\sum_{q \in S_{p-d}} B[q] + 1 \right) \times \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]}, d \in \{2, 3, \dots, f[p]\}$$

$$\omega_{p(n[p])} = A_{p(n[p])} - \sum_{q \in S_p} B[q] \times \frac{A_{i(n[i])}}{\sum_{p \in S_i} B[p]};$$

- 4) Go to Step 5.

Step 3.2.2: If it belongs to the second/third case in (19), set the robot waiting time by repeating Procedures 1) – 4) in Step 3.2.1 with both $A_{i(n[i])}/(\sum_{p \in S_i} B[p])$ and $\Delta = \Phi_i(S, S) - A_{i(n[i])}$ being replaced by $\Delta = \Phi_i(S, S)/(\sum_{p \in S_i} B[p] + 1)$ (or $\Phi_i(D, S)/(\sum_{p \in S_i} B[p] + 1)$ if it belongs to the third case);

Step 3.3: If $\Delta = \min\{\Delta_p | p \in S_i\} = \Delta_f \neq \Delta_i$, set the robot waiting time by repeating Procedures 1) – 4) in Step 3.2.1 with both $A_{i(n[i])}/(\sum_{p \in S_i} B[p])$ and $\Delta = \Phi_i(S, S) - A_{i(n[i])}$ being replaced by $\Delta = \Delta_f$, then go back to Step 3.1.

Step 4: If $k \in F$,

Step 4.1: Find the optimal cycle time $\Theta + \Delta_q$ for EST_{*k*} composed of C_k and EST_{*k-q*} (or ST_{*k-q*}), $q \in \mathbb{N}_{f[k]}$, by repeating Steps 2 and 3;

Step 4.2: $\Theta \leftarrow \Theta + \max\{\Delta_1, \Delta_2, \dots, \Delta_{f[k]}\}$;

Step 4.3: Find an O²CS for ST_{*k*} by resetting the robot waiting time with cycle time Θ by using the algorithm in [32].

Step 5: Output the schedule.

By Step 2 in Algorithm 1, if $k \notin F$ and $A_{i(n[i])} = 0$, it follows from [32] that, to make (13) or (14) satisfied, Θ

needs to be increased by $\Phi_{i+1}(D, S)$ or $\Phi_{i+1}(S, S)$ time units. By Step 3, if $k \notin F$ and $A_{i(n[i])} \neq 0$, we calculate Δ_m 's, $m \in S_i$, via (18) or (19). If $\Delta = \min\{\Delta_p | p \in S_i\} = \Delta_i$, and C_k and C_i are S-S (or D-S) case, we obtain an O²CS for EST_{*k*} by setting the robot waiting time as Theorem 2 (or 3) does. Otherwise, if $\Delta = \min\{\Delta_p | p \in S_i\} = \Delta_f \neq \Delta_i$, we obtain an O²CS for EST_{*k*} by setting the robot waiting time as Theorem 4 does. By Step 4, if $k \in F$, we find the optimal cycle time for EST_{*k*} composed of C_k and EST_{*k-q*} (or ST_{*k-q*}), $q \in \mathbb{N}_{f[k]}$, by repeating Steps 2 and 3. Then, via Corollary 1, we have $\Theta \leftarrow \Theta + \max\{\Delta_1, \Delta_2, \dots, \Delta_{f[k]}\}$. At last, by setting the robot waiting time with cycle time Θ via the algorithm in [32], we obtain an O²CS for EST_{*k*}.

With the above analysis, an O²CS for a process-dominant treelike hybrid *K*-cluster tool can be found as follows. We first find an O²CS for ST_{*j*} with $j = \max_{l \in F}\{l\}$ and its ESTs. Then, let $F \leftarrow F \setminus \{j\}$, if $F \neq \emptyset$, find $h = \max_{l \in F}\{l\}$ and an O²CS for ST_{*h*} and its ESTs. Continuing this process, an O²CS can be obtained for the entire system. However, by Algorithm 1, it requires that, for an EST_{*k*}, Condition (13) or (14) is violated for tool pair C_k and C_i only. This may not hold for general cases. To solve this problem, an O²CS for the ESTs of ST_{*j*} can be found as follows. Based on the O²CS of ST_{*j*}, for EST_{*j-1*} with C_{j-1} being not a fork, the above requirement must be satisfied. Thus, one can find an O²CS for EST_{*j-1*} by Algorithm 1. Then, one can do that for EST_{*j-2*}, till EST_{*i*} such that the upstream adjacent tool of C_i is a fork. In this way, the above requirement is always satisfied. Thus, we present the following algorithm.

Algorithm 2: Find an O²CS for a process-dominant treelike hybrid *K*-cluster tool.

Step 1: $\Theta = \max\{\Pi_1, \Pi_2, \dots, \Pi_K\}$ and $F = \{l | C_l \text{ is a fork tool}\}$.

Step 2: Check the existence of an OSLB by the algorithm in [32].

Step 2.1: If yes, determine the schedule as in [32] and go to Step 4;

Step 2.2: Otherwise find fork tool C_j with $j = \max_{l \in F}\{l\}$ and go to Step 3.

Step 3: Find O²CSs for ST_{*j*} and all its EST_{*k*}'s.

Step 3.1: Find an O²CS for ST_{*j*};

Step 3.2: Find O²CSs for all EST_{*k*}'s of ST_{*j*}

Step 3.2.1: Find k such that C_k is the upstream adjacent tool of C_j ;

Step 3.2.2: If C_k is a fork tool, go to Step 3.3; otherwise go to Step 3.2.3;

Step 3.2.3: Find an O²CS for EST_{*k*} by using Algorithm 1;

Step 3.2.4: $i \leftarrow k$, if $i = 1$, go to Step 4; otherwise find k such that C_k is the upstream adjacent tool of C_i and go to Step 3.2.2;

Step 3.3: Let $F \leftarrow F \setminus \{j\}$, if $F \neq \emptyset$, go back to Step 2.2.

Step 4: Output the schedule.

Notice that, for a treelike hybrid *K*-cluster tool, the number of B's, ST's, and EST's together must be less than *K*. Also,

any B or ST, or EST contains no more than K tools. Given B, or ST, or EST, one needs to calculate the optimal cycle time at most $(K - 1)$ times by increasing Δ . Thus, the computational complexity of the proposed method is $O(K^2)$. With K being limited in practice, the presented method is efficient and applicable to industrial cases.

V. ILLUSTRATIVE EXAMPLES

We use two examples to show the application of the proposed method.

Example 1: Consider a treelike hybrid 3-cluster tool, where C_1 is a fork tool and its adjacent downstream tools are C_2 and C_3 . Among them, C_2 and C_3 are single-arm tools, and C_1 is a dual-arm one. Their activity time is as follows. For C_1 , $(\alpha_{10}, \alpha_{11}, \alpha_{12}, \alpha_{13}, \lambda_1, \mu_1) = (0, 77, 0, 0, 23, 1)$; for C_2 , $(\alpha_{20}, \alpha_{21}, \alpha_{22}, \lambda_2, \mu_2) = (0, 75, 79, 4, 1)$; and for C_3 , $(\alpha_{30}, \alpha_{31}, \alpha_{32}, \lambda_3, \mu_3) = (0, 71, 69, 6, 1)$.

For C_1 , we have $\xi_{10} = 23$ s, $\xi_{11} = 100$ s, $\xi_{12} = 23$ s, $\xi_{13} = 23$ s, $\psi_{11} = (n[1] + 1) \times (\lambda_1 + \mu_1) = 4 \times 24 = 96$ s, and $\Pi_1 = 100$ s. For C_2 , we have $\xi_{20} = 19$ s, $\xi_{21} = 94$ s, $\xi_{22} = 98$ s, $\psi_{21} = 2(n[2] + 1) \times (\lambda_2 + \mu_2) = 6 \times 5 = 30$ s and $\Pi_2 = 98$ s. For C_3 , we have $\xi_{30} = 27$ s, $\xi_{31} = 98$ s, $\xi_{32} = 96$ s, $\psi_{31} = 2(n[3] + 1) \times (\lambda_3 + \mu_3) = 6 \times 7 = 42$ s and $\Pi_3 = 98$ s. Since $\Pi_1 = 100 > \Pi_2 = \Pi_3 = 98$ and C_1 is process-bound, it is process-dominant and we let $\Theta = \pi_1 = \pi_2 = \pi_3 = \Pi = \Pi_1 = 100$ s. By the algorithm in [32], with $\Theta = 100$ s, the robots' waiting time is set as $\omega_{20} = 6$ s, $\omega_{21} = 2$ s, and $\omega_{22} = \Pi - \psi_{21} - \omega_{20} - \omega_{21} = 100 - 30 - 6 - 2 = 62$ s; $\omega_{30} = 2$ s, $\omega_{31} = 4$ s, and $\omega_{32} = \Pi - \psi_{31} - \omega_{30} - \omega_{31} = 100 - 42 - 2 - 4 = 52$ s. Then, for C_1 , $\Pi - (4\lambda_2 + 3\mu_2 + \omega_{22}) - \lambda_1 = 100 - (19 + 62) - 23 = -4 < 0$ and $\Pi - (4\lambda_3 + 3\mu_3 + \omega_{32}) - \lambda_1 = 100 - (27 + 52) - 23 = -2 < 0$, or (14) is violated and there is no OSLB. We need to find an O^2CS .

By Algorithm 2, we have $\Delta_2 = \Phi_2(D, S)/n[2] = 4/2 = 2$ s, $\Delta_3 = \Phi_3(D, S)/n[3] = 2/2 = 1$ s, $\Delta = \max\{\Delta_2, \Delta_3\} = 2$ s, and $\Theta \leftarrow \Theta + \Delta = 102$ s. Then, let $A_{ij} = \omega_{ij}$, $i \in \mathbb{N}_3 \setminus \{1\}$ and $j \in \Omega_{i(n[i])}$. By Algorithm 2, the robot waiting time is set as $\omega_{30} = A_{30} + \Delta = 4$ s, $\omega_{31} = A_{31} + \Delta = 6$ s, $\omega_{32} = A_{32} - \Delta = 50$ s, $\omega_{20} = A_{20} + \Delta = 8$ s, $\omega_{21} = A_{21} + \Delta = 4$ s, $\omega_{22} = A_{22} - \Delta = 60$ s, $\omega_{10} = A_{10} + \Delta = 6$ s, and $\omega_{11} = \omega_{12} = \omega_{13} = 0$. In this way, an O^2CS is obtained and its Gantt chart is shown in the Supplementary File.

Example 2: Consider a treelike hybrid 5-cluster tool with C_2 being a fork tool, and its adjacent downstream tools are C_3 and C_5 . C_4 and C_2 are downstream tools of C_3 and C_1 , respectively. Among them, C_1 is a dual-arm tool and the others are single-arm ones. Their activity time is as follows: for C_1 , $(\alpha_{10}, \alpha_{11}, \lambda_1, \mu_1) = (0, 61.5, 0, 28.5, 0.5)$; for C_2 , $(\alpha_{20}, \alpha_{21}, \alpha_{22}, \lambda_2, \mu_2) = (0, 0, 0, 10, 1)$; for C_3 , $(\alpha_{30}, \alpha_{31}, \alpha_{32}, \alpha_{33}, \lambda_3, \mu_3) = (0, 56, 0, 58, 7, 1)$; for C_4 , $(\alpha_{40}, \alpha_{41}, \alpha_{42}, \alpha_{43}, \lambda_4, \mu_4) = (0, 56, 66, 65, 5, 1)$; and for C_5 , $(\alpha_{50}, \alpha_{51}, \alpha_{52}, \lambda_5, \mu_5) = (0, 48, 50, 6, 1)$.

It is found that the lower bound of cycle time is $\Theta = \Pi = 90$ s. With $\Theta = \Pi = 90$ s, the robot waiting time is set as follows. For C_4 , we have $\omega_{40} = 11$ s, $\omega_{41} = 1$ s, $\omega_{42} = 2$ s, and $\omega_{43} = 28$ s; for C_5 , $\omega_{50} = 15$ s, $\omega_{51} = 13$ s, and $\omega_{52} =$

20 s; for C_3 , $\omega_{31} = 8$ s, $\omega_{30} = 3$ s, $\omega_{32} = 1$ s, and $\omega_{33} = 14$ s; for C_2 , $\omega_{20} = 2$ s, $\omega_{21} = 0$, and $\omega_{22} = 22$ s. Then, for C_1 , $\Pi - (4\lambda_2 + 3\mu_2 + \omega_{22}) - \lambda_1 = 90 - (43 + 22) - 28.5 = -3.5 < 0$, or there is no OSLB. Therefore, we need to find an O^2CS .

With $\omega_{22} = 22 > 0$, $\omega_{52} = 20 > 0$, $\omega_{33} = 14 > 0$, and $\omega_{43} = 28 > 0$, we have $\sum_{p \in S_2} B[p] = (B[2] + B[3] + B[4] + B[5]) = (1 + 2 + 2 + 1) = 6$ and $\Phi_2(D, S) = (4\lambda_2 + 3\mu_2 + \omega_{22}) + \lambda_1 - \Pi = 3.5$ s. Then, $\Delta_2 = \Phi_2(D, S)/(\sum_{p \in S_2} B[p] + 1) = 3.5/7 = 0.5$ s, $\Delta_3 = \omega_{33}/(B[3] + B[4]) = 14/3$ s, $\Delta_4 = \omega_{43}/B[4] = 28/2 = 14$ s, $\Delta_5 = \omega_{52}/B[5] = 20/1 = 20$ s, $\Delta = \min\{\Delta_2, \Delta_3, \Delta_4, \Delta_5\} = \Delta_2 = 0.5$ s, and $\Theta \leftarrow \Pi + \Delta = 90.5$ s. Let $A_{ij} = \omega_{ij}$, $i \in \mathbb{N}_5 \setminus \{1\}$ and $j \in \Omega_{i(n[i])}$, where ω_{ij} 's are given above. Then, by Algorithm 2, the robot waiting time is set as follows. For C_4 , $\omega_{40} = A_{40} + \Delta = 11.5$ s, $\omega_{41} = A_{41} + \Delta = 1.5$ s, $\omega_{42} = A_{42} + \Delta = 2.5$ s, and $\omega_{43} = A_{43} - 2\Delta = 27$ s; for C_3 , $\omega_{31} = A_{31} + 2\Delta + \Delta = 9.5$ s, $\omega_{30} = A_{30} + \Delta = 3.5$ s, $\omega_{32} = A_{32} + \Delta = 1.5$ s, and $\omega_{33} = A_{33} - 2\Delta - 2\Delta = 12$ s; for C_5 , $\omega_{50} = A_{50} + \Delta = 15.5$ s, $\omega_{51} = A_{51} + \Delta = 13.5$ s, and $\omega_{52} = A_{52} - \Delta = 19.5$ s; for C_2 , $\omega_{20} = A_{20} + 4\Delta + \Delta = 4.5$ s, $\omega_{21} = A_{21} + \Delta + \Delta = 1$ s, and $\omega_{22} = A_{22} - 4\Delta - \Delta - \Delta = 19$ s; and for C_1 , $\omega_{10} = \Theta - \psi_{11} = 3.5$ s and $\omega_{11} = \omega_{12} = 0$. In this way, an O^2CS is obtained and its Gantt chart is shown in the Supplementary File.

VI. CONCLUSIONS

Compared with linear multi-cluster tools, a treelike hybrid multi-cluster tool is topologically more complex. Optimal scheduling of such a multi-cluster tool requires that the activities of multiple robots for accessing the shared buffering modules should be properly coordinated. For a process-dominant treelike hybrid K -cluster tool, our prior work [34] derives the conditions under which there is a one-wafer cyclic schedule that reaches the lower bound of cycle time and develops an algorithm to find it if existing. However, if these conditions are not satisfied, it is unknown whether there is an optimal one-wafer cyclic schedule or not. Can we find it if existing? Aiming at answering these two questions, this work conducts a study based on the results obtained in [34]. We have successfully shown that there is always a one-wafer cyclic schedule. We have also developed an efficient algorithm to find the minimal cycle time and the optimal one-wafer cyclic schedule.

For some wafer fabrication processes, there are strict wafer residency time constraints, and in some steps, there may have parallel modules. Meanwhile, in practice, the activity time may be subject to bounded variation. Dealing with them is our future work.

REFERENCES

- [1] N. Q. Wu and M. C. Zhou, "A closed-form solution for schedulability and optimal scheduling of dual-arm cluster tools with wafer residency time constraint based on steady schedule analysis," *IEEE Trans. Automat. Sci. Eng.*, vol. 7, no. 2, pp. 303–315, Apr. 2010.
- [2] W. K. Chan, J. G. Yi, and S. W. Ding, "On the optimality of one-unit cycle scheduling of multi-cluster tools with single-blade robots," in *Proc. IEEE Int. Conf. Automation Science and Engineering*, Scottsdale, AZ, USA, 2007, pp. 392–397.

- [3] N. Q. Wu, L. P. Bai, and M. C. Zhou, "An efficient scheduling method for crude oil operations in refinery with crude oil type mixing requirements," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 46, no. 3, pp. 413–426, Mar. 2016.
- [4] N. Q. Wu, M. C. Zhou, L. P. Bai, and Z. W. Li, "Short-term scheduling of crude oil operations in refinery with high-fusion-point oil and two transportation pipelines," *Enterp. Inform. Syst.*, vol. 10, no. 6, pp. 581–610, Sep. 2016.
- [5] N. Q. Wu, M. C. Zhou, and Z. W. Li, "Short-term scheduling of crude-oil operations: Enhancement of crude-oil operations scheduling using a petri net-based control-theoretic approach," *IEEE Robot. Automat. Mag.*, vol. 22, no. 2, pp. 64–76, Jun. 2015.
- [6] Y. Hou, N. Q. Wu, M. C. Zhou, and Z. W. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Transactions on Systems, Man, Cybernetics: Systems*, vol. 47, no. 3, pp. 517–530, Mar. 2017.
- [7] S. W. Zhang, N. Q. Wu, Z. W. Li, T. Qu, and C. D. Li, "Petri net-based approach to short-term scheduling of crude oil operations with less tank requirement," *Information Sciences*, vol. 417, pp. 247–261, Nov. 2017.
- [8] N. Q. Wu, Z. W. Li, and T. Qu, "Energy efficiency optimization in scheduling crude oil operations of refinery based on linear programming," *Journal of Cleaner Production*, vol. 166, pp. 49–57, Nov. 2017.
- [9] T. L. Perkinson, R. S. Gyurcsik, and P. K. McLarty, "Single-wafer cluster tool performance: An analysis of the effects of redundant chambers and revisitation sequences on throughput," *IEEE Trans. Semicond. Manuf.*, vol. 9, no. 3, pp. 384–400, Aug. 1996.
- [10] T. L. Perkinson, P. K. McLarty, R. S. Gyurcsik, and R. K. Cavin III, "Single-wafer cluster tool performance: An analysis of throughput," *IEEE Trans. Semicond. Manuf.*, vol. 7, no. 3, pp. 369–373, Aug. 1994.
- [11] S. P. Sethi, C. Sriskandarajah, G. Sorger, J. Blazewicz, and W. Kubiak, "Sequencing of parts and robot moves in a robotic cell," *Int. J. Flex. Manuf. Syst.*, vol. 4, no. 3–4, pp. 331–358, Jun. 1992.
- [12] S. Venkatesh, R. Davenport, P. Foxhoven, and J. Nulman, "A steady-state throughput analysis of cluster tools: Dual-blade versus single-blade robots," *IEEE Trans. Semicond. Manuf.*, vol. 10, no. 4, pp. 418–424, Nov. 1997.
- [13] N. Q. Wu, F. Chu, C. B. Chu, and M. C. Zhou, "Petri net modeling and cycle-time analysis of dual-arm cluster tools with wafer revisiting," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 1, pp. 196–207, Jan. 2013.
- [14] N. Q. Wu and M. C. Zhou, "Modeling, analysis and control of dual-arm cluster tools with residency time constraint and activity time variation based on Petri nets," *IEEE Trans. Automat. Sci. Eng.*, vol. 9, no. 2, pp. 446–454, Apr. 2012.
- [15] N. Q. Wu and M. C. Zhou, "Schedulability analysis and optimal scheduling of dual-arm cluster tools with residency time constraint and activity time variation," *IEEE Trans. Automat. Sci. Eng.*, vol. 9, no. 1, pp. 203–209, Jan. 2012.
- [16] N. Q. Wu, M. C. Zhou, F. Chu, and C. B. Chu, "A Petri-net-based scheduling strategy for dual-arm cluster tools with wafer revisiting," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 5, pp. 1182–1194, Sep. 2013.
- [17] F. J. Yang, N. Q. Wu, Y. Qiao, M. C. Zhou, and Z. W. Li, "Scheduling of single-arm cluster tools for an atomic layer deposition process with residency time constraints," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 47, no. 3, pp. 502–516, Mar. 2017.
- [18] W. M. Zuberek, "Timed Petri nets in modeling and analysis of cluster tools," *IEEE Trans. Robot. Automat.*, vol. 17, no. 5, pp. 562–575, Oct. 2001.
- [19] C. R. Pan, M. C. Zhou, Y. Qiao, and N. Q. Wu, "Scheduling cluster tools in semiconductor manufacturing: Recent advances and challenges," *IEEE Trans. Automat. Sci. Eng.*, to be published. doi: 10.1109/TASE.2016.2642997.
- [20] J. H. Kim, T. E. Lee, H. Y. Lee, and D. B. Park, "Scheduling analysis of time-constrained dual-armed cluster tools," *IEEE Trans. Semicond. Manuf.*, vol. 16, no. 3, pp. 521–534, Aug. 2003.
- [21] T. E. Lee, H. Y. Lee, and Y. H. Shin, "Workload balancing and scheduling of a single-armed cluster tool," in *Proc. 5th APIEMS Conf.*, Gold Coast, Australia, 2004, pp. 1–15.
- [22] M. J. Lopez and S. C. Wood, "Systems of multiple cluster tools: Configuration, reliability, and performance," *IEEE Trans. Semicond. Manuf.*, vol. 16, no. 2, pp. 170–178, May 2003.
- [23] D. Jevtic and S. Venkatesh, "Method and apparatus for scheduling wafer processing within a multiple chamber semiconductor wafer processing tool having a multiple blade robot," U.S. Patent 6 224 638, May 1, 2001.
- [24] S. W. Ding, J. G. Yi, and M. T. Zhang, "Multicluster tools scheduling: An integrated event graph and network model approach," *IEEE Trans. Semicond. Manuf.*, vol. 19, no. 3, pp. 339–351, Aug. 2006.
- [25] J. G. Yi, S. W. Ding, D. Z. Song, and M. T. Zhang, "Steady-state throughput and scheduling analysis of multicluster tools: A decomposition approach," *IEEE Trans. Automat. Sci. Eng.*, vol. 5, no. 2, pp. 321–336, Apr. 2008.
- [26] W. K. V. Chan, J. G. Yi, and S. W. Ding, "Optimal scheduling of multicluster tools with constant robot moving times, Part I: Two-cluster analysis," *IEEE Trans. Automat. Sci. Eng.*, vol. 8, no. 1, pp. 5–16, Jan. 2011.
- [27] W. K. V. Chan, S. W. Ding, J. G. Yi, and D. Z. Song, "Optimal scheduling of multicluster tools with constant robot moving times, Part II: Tree-like topology configurations," *IEEE Trans. Automat. Sci. Eng.*, vol. 8, no. 1, pp. 17–28, Jan. 2011.
- [28] S. Ware and R. Su, "An application of incremental scheduling to a cluster photolithography tool," in *IFAC World Congress 2017*, Toulouse, France, to be published.
- [29] Q. H. Zhu, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Petri net-based optimal one-wafer scheduling of single-arm multi-cluster tools in semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 26, no. 4, pp. 578–591, Nov. 2013.
- [30] Q. H. Zhu, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Scheduling of single-arm multi-cluster tools with wafer residency time constraints in semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 1, pp. 117–125, Feb. 2015.
- [31] F. J. Yang, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Petri net-based optimal one-wafer cyclic scheduling of hybrid multi-cluster tools in wafer fabrication," *IEEE Trans. Semicond. Manuf.*, vol. 27, no. 2, pp. 192–203, May 2014.
- [32] F. J. Yang, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Petri net-based polynomially complex approach to optimal one-wafer cyclic scheduling of hybrid multi-cluster tools in semiconductor manufacturing," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 44, no. 12, pp. 1598–1610, Dec. 2014.
- [33] L. P. Bai, N. Q. Wu, Z. W. Li, and M. C. Zhou, "Optimal one-wafer cyclic scheduling and buffer space configuration for single-arm multicluster tools with linear topology," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 46, no. 10, pp. 1456–1467, Oct. 2016.
- [34] F. J. Yang, N. Q. Wu, Y. Qiao, and M. C. Zhou, "Optimal one-wafer cyclic scheduling of hybrid multirobot cluster tools with tree topology," *IEEE Trans. Syst., Man, Cybern. Syst.*, to be published, doi: 10.1109/TSMC.2016.2587697.
- [35] Y. F. Chen, Z. W. Li, K. Barkaoui, N. Q. Wu, and M. C. Zhou, "Compact supervisory control of discrete event systems by Petri nets with data inhibitor arcs," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 47, no. 2, pp. 364–379, Feb. 2017.
- [36] H. F. Chen, N. Q. Wu, and M. C. Zhou, "A novel method for deadlock prevention of AMS by using resource-oriented Petri nets" *Inform. Sci.*, vol. 363, pp. 178–189, Oct. 2016.
- [37] N. Q. Wu, "Necessary and sufficient conditions for deadlock-free operation in flexible manufacturing systems using a colored petri net

model,” *IEEE Trans. Syst., Man, Cybern., C Appl. Rev.*, vol. 29, no. 2, pp. 192–204, May 1999.

[38] N. Q. Wu and M. C. Zhou, *System Modeling and Control with Resource-Oriented Petri nets*. New York, USA: CRC Press, 2009.

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

Fajun Yang received the B.S. degree in industrial engineering from Hunan University of Science and Technology, Xiangtan, China, in 2011, the Ph.D. degree in mechanical engineering from Guangdong University of Technology, Guangzhou, 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+ journal and conference papers. His research

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 in 2013. He is currently a Professor at the Institute of Systems Engineering, Macau University of Science and Technology. 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 now a Guest Editor of *IEEE/CAA Journal of Automatica Sinica* and an Associate Editor of *Information Sciences*.

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. 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.

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, multi-agent 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.