Optimal Scheduling of Multicluster Tools With Constant Robot Moving Times, Part II: Tree-Like Topology Configurations

Wai Kin Victor Chan, *Member, IEEE*, Shengwei Ding, Jingang Yi, *Senior Member, IEEE*, and Dezhen Song, *Senior Member, IEEE*

***Abstract*—In this paper, we analyze optimal scheduling of a tree-like multicluster tool with single-blade robots and constant robot moving times. We present a recursive minimal cycle time algorithm to reveal a multi-unit resource cycle for multicluster tools under a given robot schedule. For a serial-cluster tool, we provide a closed-form formulation for the minimal cycle time. The formulation explicitly provides the interaction relationship among clusters. We further present decomposition conditions under which the optimal scheduling of multicluster becomes much easier and straightforward. Optimality conditions for the widely used robot pull schedule are also provided. An example from industry production is used to illustrate the analytical results. The decomposition and optimality conditions for the robot pull schedule are also illustrated by Monte Carlo simulation for the industrial example.**

***Note to Practitioners*—The complexity of optimal scheduling of multicluster tools comes from cluster interactions. In this paper, we present a minimal cycle time calculation algorithm for a tree-like multicluster tool. For widely used serially-connected cluster tools, we provide an analytical formulation for computing the minimal cycle time under a given schedule. For practical applications, we present the conditions under which scheduling the multicluster tool can be decomposed into scheduling multiple single clusters. A robot pull schedule can then be applied to each single cluster tool to obtain the minimal cycle time. We demonstrate the application of the algorithms and analyses using a chemical-mechanical planarization polisher example.**

***Index Terms*—Cluster tools, multiple machines, performance/productivity, scheduling, semiconductor manufacturing.**

Manuscript received January 12, 2010; accepted March 08, 2010. Date of publication April 26, 2010; date of current version January 07, 2011. This paper was recommended for publication by Associate Editor N. Wu and Editor M. Zhou upon evaluation of the reviewers' comments. This paper was presented in part at the 2007 IEEE International Conference on Automation Science and Engineering, Scottsdale, AZ, September 22–25, 2007, and the 2008 IEEE International Conference on Automation Science and Engineering, Washington DC, August 24–26, 2008.

W.-K. V. Chan is with the Department of Industrial and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: chanw@rpi.edu).

S. Ding is with the Direct Store Delivery (DSD) Division of Nestle USA, Oakland, CA 94618 USA (e-mail: dingsw@cal.berkeley.edu).

J. Yi is with the Department of Mechanical and Aerospace Engineering, Rutgers University, Piscataway, NJ 08854 USA (e-mail: jgyi@rutgers.edu).

D. Song is with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: dzsong@cse.tamu.edu).

Corresponding author J. Yi.

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

Digital Object Identifier 10.1109/TASE.2010.2046893

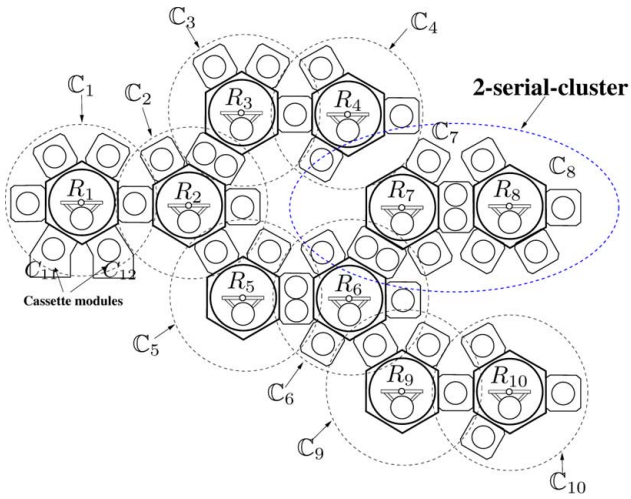

Fig. 1. A schematic of a ten-cluster tool (a tree-like topology).

I. Introduction

In THE companion paper [1], we discussed a resource cycle-based approach to analyze the minimal cycle time and optimal robot schedules for two-cluster tools. The goal of this second-part paper is to extend the methodology in [1] to tree-like M-cluster tools. A tree-like M-cluster tool consists of M single connected clusters without loops. Each cluster is equipped with a single-blade robot with a constant moving time among PMs. Fig. 1 shows an example of a ten-cluster tool. We also present sufficient conditions under which the decomposition method proposed in [2] is valid and the robot pull scheduling strategy [1]–[4], a widely used scheduling strategy in production, is optimal for multicluster tools.

The tree-like cluster topology is a generalization of most production cluster tools. Studying such a general configuration will produce knowledge not only to further understand the complexity of scheduling multicluster tools but also to design new tools such as linear cluster tools [5], [6]. The complexity in scheduling multicluster tools mainly lies at understanding interactions among clusters [1]. We studied the interactions between two adjacent clusters in [1]. We will show in this paper that the cluster interactions not only exist for any two adjacent clusters but also among multiple clusters within one branch or even cross-branch clusters of multicluster tools. We provide analyses and algorithms to capture these interactions.

Multicluster scheduling has been an active research area in recent years. Geismar *et al.* [7] present the lower-bound throughput of a three-cluster robotic cell with single-blade robots and compare the tool's throughput with the pull strategy by simulation. The pull schedule specifies the robot movement by moving wafers from the last process module (PM) to the first PM in sequence. It shows that the pull strategy achieves the lower-bound performance in 87% of all simulated cases. A simulation-based approach is also reported in [8] for multicluster tools. Yi *et al.* [2] has studied the decomposition and optimality conditions of the pull schedule. Zero robot moving times are assumed in [2] and under such an assumption, the optimality of the pull schedule is obtained.

All existing work focus on serially-connected multicluster tools. We consider a more complex tree-like cluster connection topology. In this paper, we use the resource cycle-based approach [1] and we take a buffer-centered viewpoint to understand the interaction among clusters. A recursive approach is proposed to capture the aggregated cluster interactions. The contribution of this paper is not simply an extension of the results in [1] and [2]. First, we consider constant robot moving times and the pull scheduling discussed in [2] is no longer always optimal. Second, the tree-like cluster connection topology introduces complex interactions among multiple clusters that can be located across several branches of the cluster tree, and the results in [1] cannot be directly applied to capture these interactions. Finally, the analyses in this paper are used to derive the decomposition and optimality conditions for robot pull schedules.

The remainder of this paper is organized as follows. In Section II, we present the resource cycle-based recursive algorithm and analysis. In Section III, we present a closed-form formulation of minimal cycle time calculation and properties of scheduling a serially connected multicluster tool. We discuss decoupling conditions and optimality conditions for the pull schedule in Section IV. In Section V, we present an illustrative example before concluding the paper in Section VI.

II. Resource-Based M-Cluster Tool Analysis

A. M-Cluster Tool Configuration

We consider the following assumptions for an M-cluster tool: 1) all wafers follow an identical flow that visits each PM exactly once; 2) the load-locks at the first single cluster always has wafers/spaces for a robot to pick or place at any time; 3) all the activity times are deterministic; and 4) buffer modules have either one- or two-wafer capacity. We use the same notations and definitions in the companion paper [1] without reintroducing them. As we discussed in [1], cluster scheduling can be determined by robot moving sequences and therefore, we use activity sequences to describe cluster tools' schedules.

We consider an m-serial-cluster tool, as shown in Fig. 2, as a special type of multicluster tools. An *m-serial-cluster tool* consists of m single cluster tools connected in series, and each single cluster connects to exactly two neighboring clusters, except the *leaf (ending)* and the *head (starting)* clusters, which connect to only one neighboring cluster. We define a *root cluster* that is connected to several serial-cluster tools. For example, clusters $\mathbb{C}_2$ and $\mathbb{C}_6$ in Fig. 1 are root clusters. For any two

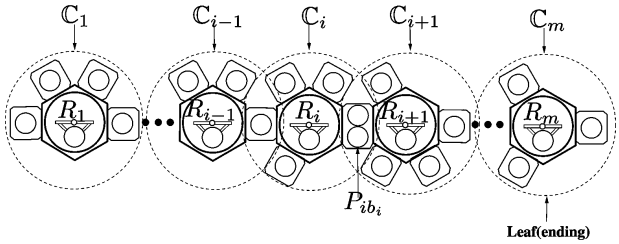

Fig. 2. A schematic of an m-serial-cluster tool with leaf cluster $\mathbb{C}_m$.

connected clusters $\mathbb{C}_i$ and $\mathbb{C}_j$[1], we always denote the downstream cluster $\mathbb{C}_j$ with a larger cluster indexing number, that is, $j > i$. We also denote the buffer module between $\mathbb{C}_i$ and $\mathbb{C}_j$ as $P_{ib_{ij}}$.

We assume a single-cluster $\mathbb{C}_i$ has c_i PMs, denoted by P_{il}, $l = 1, \ldots, c_i$, and a single-blade robot R_i. Buffer module $P_{ib_{ij}}$, $1 \leq b_{ij} \leq c_i$, is considered as one PM. Associated with $\mathbb{C}_i$, there is a vector of activity times $\mathbf{t}_i = (t_{i0}, t_{i1}, \ldots, t_{ic_i}; \epsilon_i, \delta_i)$, where ϵ_i and δ_i are robot R_i pick/place time and moving time among PMs, respectively. For presentation clarity, let $RCT(P_{ib_{ij}})$ and $RCT^0(P_{ib_{ij}})$ denote the resource cycle time of buffer module $P_{ib_{ij}}$ by assuming its virtual processing times are $t_{v_{ij}}$ and zero, respectively. Since the buffer module $P_{ib_{ij}}$ plays a dual role as the virtual PM (of $\mathbb{C}_i$) and the virtual load-locks module (P_{j0} of $\mathbb{C}_j$), similar to the literature and [1], we assume a robot moving time δ_i for R_i to access buffer $P_{ib_{ij}}$ P_{j0}.

As shown in [1] and [4], the so-called basic cycles have smaller cycle times than other nonbasic cycles in single-cluster tools. In this paper, we focus on basic cycles in scheduling individual single clusters of an M-cluster tool.

B. Dual-Role Properties of Buffer Modules

In [1], we analyzed the cycle time of a two-cluster tool that consists of $\mathbb{C}_i$ and $\mathbb{C}_j$ with buffer module $P_{ib_{ij}}$. We show that the cycle time of this two-cluster tool can be broken down into three parts: Part 1 is the cycle time of $\mathbb{C}_i$ by treating the buffer module $P_{ib_{ij}}$ as a virtual PM with processing time $t_{v_{ij}}$; Part 2 is the cycle time of $\mathbb{C}_j$ decoupled from $\mathbb{C}_i$ by assuming infinite wafers and spaces at the virtual load-locks $P_{j0}(P_{ib_{ij}})$; and Part 3 is the time delay due to the interaction between $\mathbb{C}_i$ and $\mathbb{C}_j$ and characterized by the n-concatenated robot-move-sequences (n-CRM, $n > 1$) cycle with the average one-unit cycle time K_{ij}[1].

By treating $P_{ib_{ij}}$ as a virtual PM with virtual processing time $t_{v_{ij}}$, we define

$$T_i^F = \beta_i^{\delta_i} + \sum_{l=1}^{c_i} \alpha_{il} \tag{1}$$

with $\beta_i^{\delta_i} = 2\epsilon_i + \delta_i$ and $\alpha_{il} = t_{il} + \delta_i + 2\epsilon_i$. We extend the results in [1] by defining two operating modes of the buffer module $P_{ib_{ij}}$:

(1) "Closed"-mode (*C-mode*): In this mode, the buffer module is treated as a virtual PM with processing time $t_{v_{ij}} = RCT(P_{j0})$. Any cycle time calculation then follows by treating $P_{ib_{ij}}$ as a virtual PM.

[1]Due to the tree-like topology, indexes of adjacent two clusters are not necessary in a consecutive order, for example, $\mathbb{C}_2$ and $\mathbb{C}_5$, or $\mathbb{C}_6$ and $\mathbb{C}_9$ in Fig. 1.

(2) "Expanded"-mode (*E-mode*): In this mode, all the robot movement and processing times in $\mathbb{C}_j$ (and any other connected clusters in the downstream branch) are included in the n-CRM cycle through $P_{ib_{ij}}$. The n-CRM cycle lasts T_i^F and produces n_j wafers.

The introduction of these two buffer-operating modes will facilitate the discussion on characterizing the interactions among clusters. These interactions are clearly seen from the dual role of the buffer modules: on one hand, the interaction between two clusters can be simply decoupled and we can compare the one-unit cycle times of each single cluster to obtain the resulting cycle time. The schedule of the two-cluster tool is obtained by combining and aligning two single-cluster schedules. This fact is captured by treating the buffer in C-mode. On the other hand, the interaction between two clusters can produce n-CRM cycles, which could dominate the one-unit cycle of any individual single cluster. This phenomenon has been discussed in details in [1] and is captured through treating buffers in E-mode. If the C-mode and E-mode roles of the buffer modules are considered recursively, we capture all possible interactions among connected clusters, not restricting to two adjacent clusters or connected clusters in one branch.

Let CYC_k denote the kth n-CRM cycle ($n \geq 1$) with the M-cluster tool. Associated with CYC_k, we define a triplet $(T(CYC_k), n(CYC_k), B(CYC_k))$, where $T(CYC_k)$ is the cycle time of CYC_k, $n(CYC_k)$ is the number of wafers produced by CYC_k, and $B(CYC_k)$ is the set of buffer modules that are involved in CYC_k. For example, for any 1-CRM cycle CYC_k (i.e., one-unit resource cycle) of $\mathbb{C}_i$

$$T(CYC_k) = \begin{cases} RCT(P_{il}), & CYC_k \text{ is for resource } P_{il} \\ RCT(R_i), & CYC_k \text{ is for resource } R_i \end{cases} \tag{2a}$$

$$n(CYC_k) = 1 \tag{2b}$$

$$B(CYC_k) = \begin{cases} \text{Buff}(P_{il}), & CYC_k \text{ is for resource } P_{il} \\ \text{Buff}(R_i), & CYC_k \text{ is for resource } R_i \end{cases}, \tag{2c}$$

where $l = 1, \ldots, c_i$, and

$$\text{Buff}(P_{il}) = \{P_{ij}, j \in UA(P_{il}), P_{ij} \text{ is a buffer module}\}$$
$$\text{Buff}(R_i) = \{P_{ij}, j \in \Lambda_i^R, P_{ij} \text{ is a buffer module}\}.$$

Remark 1: The introduction of triplet $(T(CYC_k), n(CYC_k), B(CYC_k))$ to characterize the property of resource cycle CYC_k one of the key developments for analyzing and capturing the interactions among clusters. In Section II-C, we search and track each feasible resource cycle of cluster tools by updating the triplet representation.

C. Recursive Minimal Cycle Time Analysis of M-Cluster Tools

Following the above buffer-mode viewpoint, we calculate the minimal cycle time of a two-connected-cluster tool ($\mathbb{C}_i$ and $\mathbb{C}_j$) as follows: Step 1: find the 1-CRM cycle time $T_i(\pi_i)$ of $\mathbb{C}_i$ with $P_{ib_{ij}}$ treated as the *C-mode* with processing time $t_{v_{ij}}$; Step 2: find the 1-CRM cycle time $T_j(\pi_j)$ of $\mathbb{C}_j$ assuming always wafers/spaces in $P_{j0}(P_{ib_{ij}})$; Step 3: using the existing n-CRM cycles CYC_k, find the new n-CRM cycle $CYC_{k'}$ by treating $P_{ib_{ij}}$ in the E-mode. The corresponding cycle time $T(CYC_{k'})$ is offset by $T_j^F - RCT(P_{j0})$, namely, $T(CYC_{k'}) \leftarrow T(CYC_k) + T_j^F - RCT(P_{j0})$, and corresponding wafer number $n(CYC_{k'})$ is increased by $n_j - 1$, namely, $n(CYC_{k'}) \leftarrow n(CYC_k) + n_j - 1$, where n_j is determined by [1, Lemma 1]. We repeat the above Step 3 for all possible n-CRM cycles. Let $\mathbb{K} = \{k, k', \ldots\}$ denote the indexing set of all n-CRM cycles. The minimum cycle time of the two-cluster tool is then $\max_{l \in \mathbb{K}} \{T_i(\pi_i), T_j(\pi_j), CT(CYC_l)\}$, where the average one-unit cycle time $CT(CYC_l)$ of CYC_l is calculated as $CT(CYC_l) = T(CYC_l)/n(CYC_l)$.

We generalize the above approach recursively to any tree-like M-cluster tools. Algorithm 1 illustrates the procedure to compute the minimal cycle time of an M-cluster tool under a given schedule. The algorithm defines two separate sets, $\mathbb{L}$ and $\mathbb{K}$, to save the 1-CRM and n-CRM ($n > 1$) cycles, respectively. The interactions among single clusters are captured by the n-CRM cycles CYC_k. The n-CRM properties are recursively updated by the triplet $(T(CYC_{k'}), n(CYC_{k'}), B(CYC_{k'}))$ of $CYC_{k'}$. The minimal cycle time of the M-cluster is then the maximum of all one-unit cycle times and the average one-unit cycle times of all n-CRM cycles.

Algorithm 1: Minimal cycle time calculation of an M-cluster tool under a schedule $\boldsymbol{\pi} = \{\pi_1, \pi_2, \ldots, \pi_M\}$.

```
Initialize 𝕄 ← ∅, 𝕃 ← ∅, 𝕂 ← ∅;
While |𝕄| < M do
    Initialize C ← ∅;
    for i ∈ 𝕄 do
        find the direct connected cluster C_j. If j ∉ 𝕄,
        C ← C ∪ {j};
    end
    If |𝕄| < 1, C ← C ∪ {1};
    for j ∈ C do
        Assume C-mode for buffer module P_{jb_ij};
        Find all 1-CRM cycles CYC_l, l ∈ L_j (index set
        L_j of all 1-CRM cycles) of C_j using (2);
        𝕃 ← 𝕃 ∪ L_j, T_j(π_j) = max_{l∈L_j} T(CYC_l);
        for k ∈ 𝕂 ∪ 𝕃 do
            If P_{j0} ∈ B(CYC_k), form a new n-CRM
            CYC_{k'} with triplet properties
            T(CYC_{k'}) ← T(CYC_k) + T_j^F − RCT(P_{j0}),
            n(CYC_{k'}) ← n(CYC_k) + n_j − 1,
            B(CYC_{k'}) ← B(CYC_k)+
            {C_j buffer modules}
            CT(CYC_{k'}) ← (T(CYC_{k'})/n(CYC_{k'}));
            𝕂 ← 𝕂 ∪ {k'}, 𝕄 ← 𝕄 ∪ {j};
        end
    end
end
T_{1M} = max_{1≤j≤M, k∈𝕂} {T_j(π_j), CT(CYC_k)}.
```

The following two examples illustrate the algorithm and the dependences among clusters.

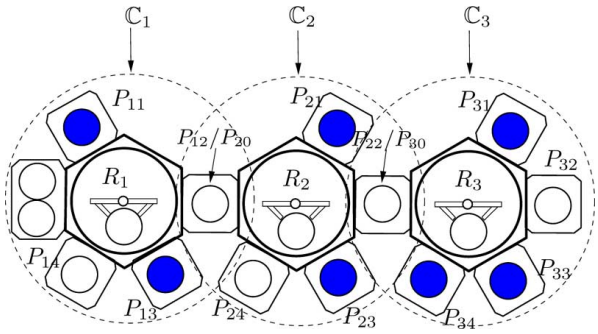

Fig. 3. A three-serial-cluster tool example under schedule (π_1, π_2, π_3).

Example 1: Fig. 3 shows a three-cluster tool with two single-space buffers. Consider the schedule (π_1, π_2, π_3) as

$$\pi_1 = (A_{10}A_{13}A_{14}A_{12}A_{11}), \quad \pi_2 = (A_{20}A_{23}A_{24}A_{22}A_{21}),$$
$$\pi_3 = (A_{30}A_{31}A_{34}A_{33}A_{32}).$$

By [1, Lemma 1], schedule (π_1, π_2, π_3) gives a wafer distribution $(n_1 - 1, n_2 - 1, n_3) = (2, 2, 3)$.

Algorithm 1 first considers $\mathbb{C}_1$ under the *C-mode* for buffer $P_{12}(P_{20})$. Four 1-CRM cycles are added to $\mathbb{L}$:

$$CYC_1^1 = (RCT(R_1), 1, \emptyset), \quad CYC_1^2 = (RCT(P_{14}), 1, \emptyset),$$
$$CYC_1^3 = (RCT(P_{12}), 1, \{P_{12}\}),$$
$$CYC_1^4 = (RCT(P_{11}), 1, \emptyset).$$

$T_1(\pi_1) = \max_{1 \le l \le 4} T\left(CYC_1^l\right)$. Then, $\mathbb{C}_2$ is considered and there are four new 1-CRM cycles in $\mathbb{L}$:

$$CYC_2^1 = (RCT(R_2), 1, \emptyset), \quad CYC_2^2 = (RCT(P_{24}), 1, \emptyset),$$
$$CYC_2^3 = (RCT(P_{22}), 1, \{P_{22}\}),$$
$$CYC_2^4 = (RCT(P_{21}), 1, \emptyset).$$

$T_2(\pi_2) = \max_{1 \le l \le 4} T\left(CYC_2^l\right)$. The algorithm checks if buffer module P_{12} has been included in any resource cycle and CYC_1^3 is found. Then, a new 3-CRM cycle $CYC_{1'} = \left(RCT(P_{12}) + T_2^F - RCT(P_{20}), 3, \{P_{12}, P_{22}\}\right)$ is added to $\mathbb{K}$. Next, $\mathbb{C}_3$ is considered and four new 1-CRM cycles are added to $\mathbb{L}$:

$$CYC_3^1 = (RCT(R_3), 1, \emptyset), \quad CYC_3^2 = (RCT(P_{34}), 1, \emptyset),$$
$$CYC_3^3 = (RCT(P_{33}), 1, \emptyset), \quad CYC_3^4 = (RCT(P_{31}), 1, \emptyset).$$

$T_3(\pi_3) = \max_{1 \le l \le 4} T\left(CYC_3^l\right)$. The algorithm checks if buffer module P_{22} has been included in any of the previously identified resource cycles and results in CYC_2^3 and $CYC_{1'}$. Then, two more new cycles, one 3-CRM and the other 5-CRM, $CYC_{2'} = \left(RCT(P_{22}) + T_3^F - RCT(P_{30}), 3, \{P_{22}\}\right)$ and $CYC_{3'} = \left(RCT(P_{12}) + T_2^F - RCT(P_{20}) + T_3^F - RCT(P_{30}), 5, \{P_{12}, P_{22}\}\right)$, respectively, are added to $\mathbb{K}$. The minimal cycle time of the three-cluster tool is given as

$$T_{13} = \max\left\{T_1(\pi_1), T_2(\pi_2), T_3(\pi_3), \max_{1 \le k \le 3} CT(CYC_{k'})\right\}.$$

TABLE I
PROCESSING TIMES (IN SECONDS) OF THE THREE-SERIAL-CLUSTER TOOL IN EXAMPLE 1

Cases	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
1	(10, 0, 10, 10; 5, 3)	(70, 0, 58, 5; 3, 1)	(45, 2, 2, 2; 3, 1)
2	(34, 0, 31, 4; 6, 3)	(4, 0, 4, 4; 8, 1)	(35, 35, 89, 88; 3, 1)
3	(34, 0, 31, 4; 10, 1)	(82, 0, 54, 12; 7, 1)	(54, 38, 91, 90; 3, 1)

TABLE II
CYCLE TIMES CALCULATION OF THE THREE-SERIAL-CLUSTER TOOL IN EXAMPLE 1 UNDER THREE DIFFERENT PROCESSING TIMES (IN SECONDS)

Cases	$T_1(\pi_1)$	$T_2(\pi_2)$	$T_3(\pi_3)$	CT_1	CT_2	CT_3	T_{13}
Fig. 4(a)	87	85	84	88	33.67	56.6	88
Fig. 4(b)	91	93	104	62.33	105.67	82.4	105.67
Fig. 4(c)	113	113	114	114	113	114.8	114.8

Table I lists three different sets of the processing times $\mathbf{t}_i = (t_{i1}, t_{i2}, t_{i3}, t_{i4}; \epsilon_i, \delta_i)$ (in s) of $\mathbb{C}_i$, $i = 1, 2, 3$. For Case 1, the 3-CRM cycle $CYC_{1'}$ (interaction between $\mathbb{C}_1$ and $\mathbb{C}_2$) dominates and is highlighted in the Gantt chart in Fig. 1. For Case 2, the 3-CRM cycle $CYC_{2'}$ (interaction between $\mathbb{C}_2$ and $\mathbb{C}_3$) dominates and is highlighted in the Gantt chart in Fig. 1. Finally, for Case 3, the 5-CRM cycle $CYC_{3'}$ (interaction among $\mathbb{C}_1$, $\mathbb{C}_2$ and $\mathbb{C}_3$) dominates and is highlighted in the Gantt chart in Fig. 1. Fig. 5 shows wafer flows of the corresponding n-CRM cycles. The detailed calculation of the cycle times is listed in Table II.

Example 2: Fig. 6 shows a tree-like three-cluster tool. Consider the schedule (π_1, π_2, π_3) as

$$\pi_1 = (A_{10}A_{14}A_{12}A_{13}A_{11}), \quad \pi_2 = (A_{20}A_{23}A_{24}A_{22}A_{21}),$$
$$\pi_3 = (A_{30}A_{31}A_{34}A_{33}A_{32})$$

and the consequent wafer distribution is $(n_1 - 1, n_2, n_3 - 1) = (2, 3, 2)$.

Similar to the previous example, Algorithm 1 first considers cluster $\mathbb{C}_1$ with the C-mode for buffers $P_{12}(P_{20})$ and $P_{13}(P_{30})$. Four 1-CRM cycles are added to $\mathbb{L}$: $CYC_1^1 = (RCT(R_1), 1, \{P_{13}\})$, $CYC_1^2 = (RCT(P_{14}), 1, \{P_{13}\})$, $CYC_1^3 = (RCT(P_{12}), 1, \{P_{12}, P_{13}\})$, and $CYC_1^4 = (RCT(P_{11}), 1, \emptyset)$. Then, $\mathbb{C}_2$ is considered next and there are other four 1-CRM cycles into $\mathbb{L}$:

$$CYC_2^1 = (RCT(R_2), 1, \emptyset), \quad CYC_2^2 = (RCT(P_{24}), 1, \emptyset),$$
$$CYC_2^3 = (RCT(P_{22}), 1, \{P_{22}\}),$$
$$CYC_2^4 = (RCT(P_{21}), 1, \emptyset).$$

The algorithm checks if P_{12} has been included in any identified resource cycle and CYC_1^3 is found. A 3-CRM cycle $CYC_{1'} = \left(RCT(P_{12}) + T_2^F - RCT(P_{20}), 3, \{P_{12}, P_{13}\}\right)$ is then added to $\mathbb{K}$. Next, $\mathbb{C}_3$ is considered and four new 1-CRM cycles similar to the previous example are added to $\mathbb{L}$. Finally, the algorithm checks if buffer module P_{13} has been included in any of the previously identified resource cycles and it results in CYC_1^1, CYC_1^2, CYC_1^3 and $CYC_{1'}$. Then, four more new cycles, three 3-CRMs and one 5-CRM, $CYC_{2'} = \left(RCT(R_1) + T_3^F - RCT(P_{30}), 3, \{P_{13}\}\right)$, $CYC_{3'} = \left(RCT(P_{14}) + T_3^F - RCT(P_{30}), 3, \{P_{13}\}\right)$, $CYC_{4'} = \left(RCT(P_{12}) + T_2^F - RCT(P_{20}), 3, \{P_{12}, P_{13}\}\right)$, and $CYC_{5'} =$

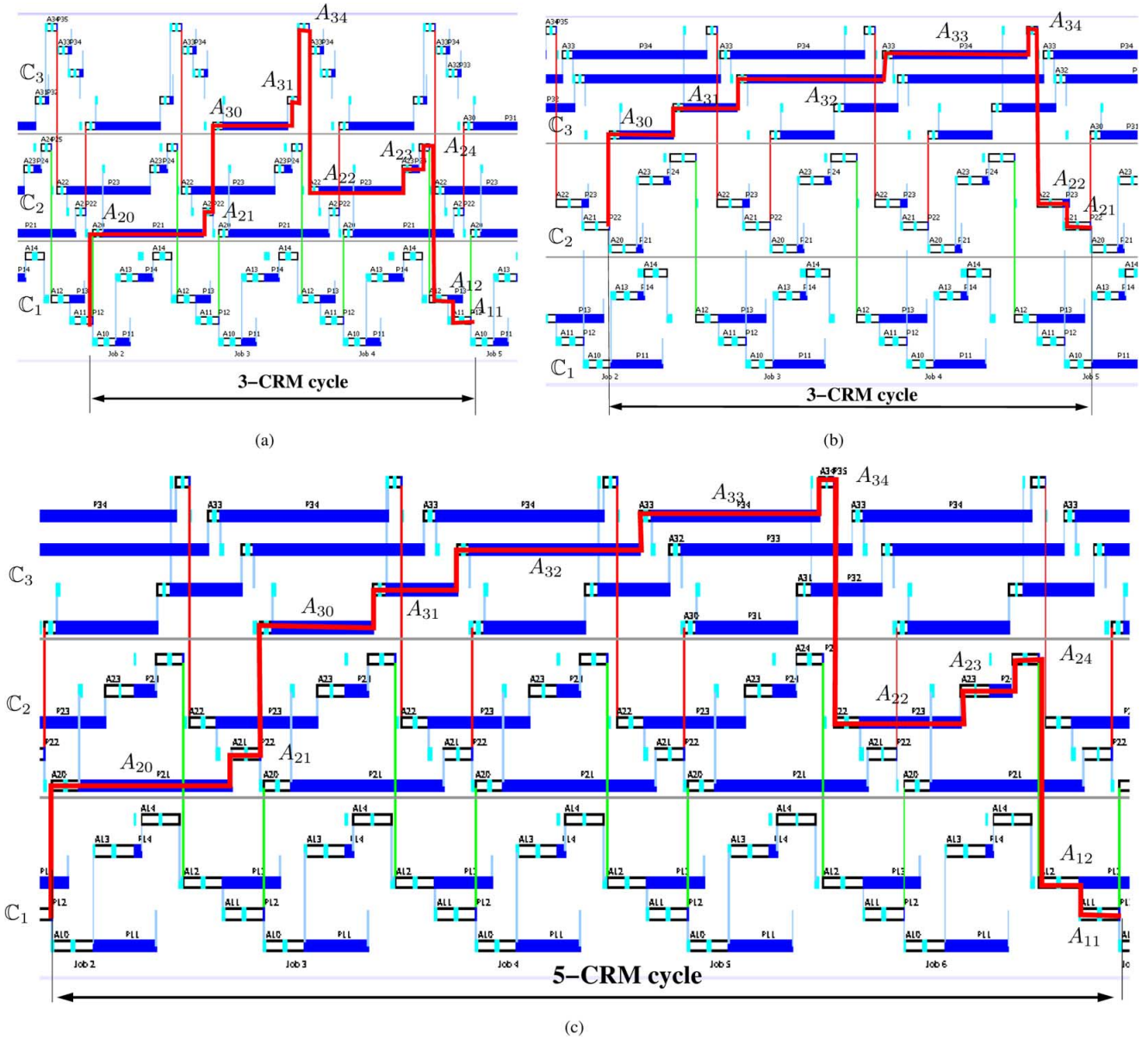

Fig. 4. The Gantt charts of the three n-CRM cycles for the three-cluster tool in Example 1. (a) A 3-CRM cycle between $\mathbb{C}_1$ and $\mathbb{C}_2$. (b) A 3-CRM cycle between $\mathbb{C}_2$ and $\mathbb{C}_3$. (c) A 5-CRM cycle among $\mathbb{C}_1$, $\mathbb{C}_2$ and $\mathbb{C}_3$. In all figures, the red highlighted lines show the critical paths that generate the minimal cycle time of the three-cluster tool under the processing time configuration.

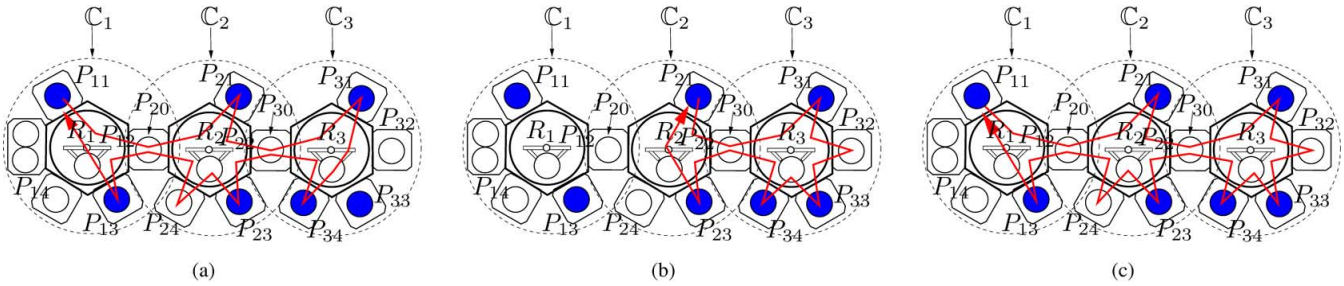

Fig. 5. The schematic of wafer flows of the three n-CRM cycles for the three-cluster tool in Example 1.

$\big(RCT(P_{12}) + T_2^F - RCT(P_{20}) + T_3^F - RCT(P_{30}), 5, \{P_{12}, P_{13}\}\big)$ are added to $\mathbb{K}$. The minimal cycle time of the three-cluster tool is given as

$$T_{13} = \max\left\{T_1(\pi_1), T_2(\pi_2), T_3(\pi_3), \max_{1\le k\le 5} CT\left(CYC_{k'}\right)\right\}.$$

We list five different combinations of the cluster processing times as shown in Table III. Applying Algorithm 1, we obtain the minimum cycle time T_{13} of the three-cluster tool as listed in Table IV. For each case, one of the n-CRM cycles dominates all other cycles. Therefore, the minimal cycle time of the mul-

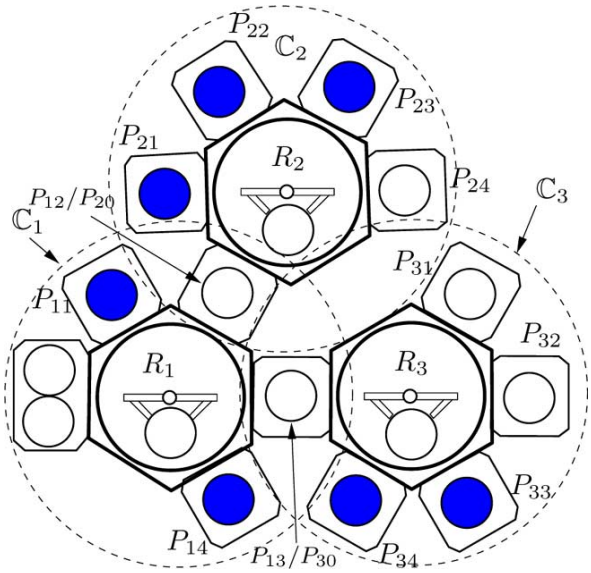

Fig. 6. A tree-like three-cluster tool example under schedule (π_1, π_2, π_3).

TABLE III
PROCESSING TIMES (IN SECONDS) OF THE TREE-LIKE THREE-CLUSTER TOOL IN EXAMPLE 2

Cases	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
1	$(10, 0, 0, 10; 5, 2)$	$(70, 70, 58, 5; 3, 1)$	$(5, 58, 70, 70; 3, 1)$
2	$(10, 0, 0, 10; 5, 2)$	$(70, 70, 58, 5; 3, 1)$	$(10, 2, 2, 2; 3, 1)$
3	$(10, 0, 0, 10; 5, 2)$	$(50, 50, 48, 5; 3, 1)$	$(5, 58, 70, 70; 3, 1)$
4	$(10, 0, 0, 49; 5, 2)$	$(50, 50, 48, 5; 3, 1)$	$(5, 58, 70, 70; 3, 1)$
5	$(10, 0, 0, 10; 5, 1)$	$(50, 50, 48, 5; 3, 1)$	$(5, 58, 70, 70; 3, 1)$

TABLE IV
CYCLE TIMES CALCULATION OF THE TREE-LIKE THREE-CLUSTER TOOL IN EXAMPLE 2 UNDER FIVE DIFFERENT PROCESSING TIMES (IN SECONDS)

Cases	$T_1(\pi_1)$	$T_2(\pi_2)$	$T_3(\pi_3)$	CT_1	CT_2	CT_3	CT_4	CT_5	T_{13}
1	95	85	85	101	102	91	101	**102.8**	102.8
2	100	85	49	**102.7**	39.7	27	38.7	65.4	102.7
3	95	75	85	84.3	**102**	91	101	92.8	102
4	101	75	85	84.3	102	**104**	101	92.8	104
5	88	75	85	83	99	89	**99.7**	92	99.7

ticluster tool depends on the timing configurations and the possible interactions among connected clusters. These connected clusters are not necessarily adjacent in topological locations in the cluster. For example, under Case 5, the interaction between $\mathbb{C}_2$ and $\mathbb{C}_3$, which are not directly connected, can dominate the cycle time of the three-cluster tool.

D. Complexity of Algorithm 1

Since Algorithm 1 goes through all possible cycles of the M-cluster tool, its complexity depends on the complexity of the tool itself. For example, if the total amount of possible cycles in the system is $n = |\mathbb{K} \cup \mathbb{L}|$ and the number of clusters in the system is M, then the maximum amount of searches for Line 11 statement in Algorithm 1 is $O(nM)$. However, given M, the value of n depends on the cluster tool's configuration.

We give an extreme example of high complexity. We consider a six-cluster tool ($M = 6$). Cluster $\mathbb{C}_1$ is the root cluster, $\mathbb{C}_2, \ldots, \mathbb{C}_6$ are all leaf clusters and they are connected through buffer modules $P_{12}, \ldots, P_{16}$ of $\mathbb{C}_1$. This is a very similar topology to the cluster tool in Example 2 shown in Fig. 6. We assume that in the given schedule, $P_{12}, \ldots, P_{16}$ all belong to Λ^R. Then, any of the five buffer modules, $P_{12}, \ldots, P_{16}$, can be in either C-mode or E-mode by Algorithm 1. This generates a total amount of $n = 2^{M-1}$ possible n-CRM cycles. If M increases, the complexity of Algorithm 1 is exponential as $O(M2^M)$.

Remark 2: Note that even if we use other modeling approach such as graphic modeling methods for cluster tool scheduling problem, we cannot avoid searching for all feasible n-CRM cycles. For example, if a Petri net model is applied to study the system [10], [11], we need to find the corresponding circuits for each n-CRM cycle. This requires a complete search of the graph, which could be exponential. On the other hand, most production cluster tools' configuration are much simpler than the above extreme example. For example, the m-serial-cluster tools discussed in Section III allow some special properties such that cost for searching all feasible n-CRM cycles is polynomial.

III. m-SERIAL-CLUSTER TOOLS

In this section, we analyze m-serial-cluster tools using the results presented in Section III. The motivation for emphasizing the study of m-serial-cluster tools is to reveal some underlying properties for most practical applications.

We consider an m-serial-cluster tool as shown in Fig. 2. Without loss of generality, we number the cluster sequentially from lead to leaf clusters as $\mathbb{C}_1$ to $\mathbb{C}_m$. We also denote the buffer module between $\mathbb{C}_i$ and $\mathbb{C}_{i+1}$ as P_{ib_i}. To simplify the presentation, we shall assume that: 1) all buffer modules are single space and 2) buffer module P_{ib_i} is not in the resource of $RCT(P_{i0})$, $i = 1, \ldots, m$. It is straightforward to relax the first assumption and obtain the similar results for double-space buffer cases from [1, Th. 3]. The reason we use the second assumption is to reduce the number of n-CRM cycles induced by nonadjacent clusters to obtain reasonable analytical results for practical applications. The use of these assumptions however does not restrict our results.

Theorem 1: For an m-serial-cluster tool under a schedule $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$, the minimal cycle time is

$$T_{1m}(\pi_1, \ldots, \pi_m) = \max\left\{ \max_{1 \le i \le m} T_i(\pi_i), \max_{\substack{1 \le i \le m-1 \\ j > i}} K_{ij} \right\}, \tag{3}$$

where for $j = i+1, \ldots, m$, $i = 1, \ldots, m-1$

$$K_{ij} = \frac{RCT(P_{ib_i}) + \sum_{k=i+1}^{j} \left(T_k^F - RCT(P_{k0})\right)}{\sum_{k=i+1}^{j-1} (n_k - 1) + n_j}. \tag{4}$$

$T_i(\pi_i)$ is obtained by using a virtual processing time $t_{v_i} = \beta_i^{\delta_i} + \sum_{l=1}^{p_i} \alpha_{il} + \beta_i + \sum_{l=q_i+1}^{c_i} \alpha_{il}$, where

$$p_i = \max\{l : \text{activity sequence } (\mathrm{A_{i0}A_{i1}} \ldots \mathrm{A_{il}}) \text{ is contained in } \pi_i\},$$
$$q_i = \min\{l : \text{activity sequence } (\mathrm{A_{il}A_{i,l+1}} \ldots \mathrm{A_{i,c_{i-1}}A_{i,c_i}}) \text{ is contained in } \pi_i\}.$$

If there exists $b_i \notin \Lambda_i^P$, then the term $RCT(P_{ib_i})$ in K_{ij} becomes $RCT(P_{ib_i}) = \max\{RCT(Q) : Q \in \left\{P_{ik_L(P_{ib_i})}, P_{ik_R(P_{ib_i})}, R_i\right\}\}$, where $P_{ik_L(P_{ib_i})}$ and $P_{ik_R(P_{ib_i})}$ are the similar definitions in [1, Th. 2].

Proof: All 1-CRM cycle time $T_i(\pi_i)$ terms [i.e., the first term in right side of (3)] are given in Algorithm 1 by recursively using virtual processing time t_{v_i}. Alternatively, calculation of these terms can also be obtained from extension of [1, Th. 2]. We focus on the proof of the terms K_{ij} in (3). Since all clusters are connected in series, each n-CRM cycle can be identified by a subset of clusters, and without loss of generality, we denote such a subset of clusters as $(\mathbb{C}_i, \mathbb{C}_{i+1}, \ldots, \mathbb{C}_j)$, where $\mathbb{C}_i$ and $\mathbb{C}_j$ are the involved lowest and highest indexed clusters, respectively. Then, buffer modules $P_{ib_i}, P_{i+1,b_{i+1}}, \ldots, P_{j-1,b_{j-1}}$ are all in E-mode. If $j = m$, there is no P_{jb_j}; otherwise, buffer P_{jb_j} is in the C-mode. If all $b_l \in \Lambda_l^P$, then there is only one 1-CRM in $\mathbb{C}_l$ in which P_{lb_l} is involved, $l = i, i+1, \ldots, j-1$. So we can use K_{ij} to denote the cycle times of all these n-CRM cycles correspondingly. According to Algorithm 1, the triplet of n-CRM CYC of $(\mathbb{C}_i, \mathbb{C}_{i+1}, \ldots, \mathbb{C}_j)$ has the following attributes:

$$\begin{aligned} T(CYC) &= RCT(P_{ib_i}) + T_{i+1}^F - RCT(P_{i+1,0}) \\ &\quad + \cdots + T_j^F - RCT(P_{j0}) \\ &= RCT(P_{ib_i}) + \sum_{k=i+1}^{j} \left(T_k^F - RCT(P_{k0})\right), \quad (5) \\ n(CYC) &= 1 + (n_{i+1} - 1) + \cdots + (n_{j-1} - 1) + (n_j - 1) \\ &= \sum_{k=i+1}^{j-1} (n_k - 1) + n_j, \\ B(CYC) &= \{P_{ib_i}, P_{i+1,b_{i+1}}, \ldots, P_{j-1,b_{j-1}}, P_{jb_j}\}. \quad (6) \end{aligned}$$

Then, the average one-unit cycle time $K_{ij} = CT(CYC) = T(CYC)/n(CYC)$ can be expressed as in (4).

On the other hand, if there exists $b_l \in \Lambda_l^P, l = i, i+1, \ldots, j-1$, then there are three cases of 1-CRM in $\mathbb{C}_l$ in which P_{lb_l} is involved. These three cases are indeed captured by set Q defined in the theorem. In these cases, we follow the above proof with $RCT(Q)$ replacing $RCT(P_{ib_i})$ in (5) and (6). This completes the proof. ∎

Theorem 1 can be viewed as an extension of the formulation for two-cluster tools given in [1, Th. 2]. However, unlike the two-cluster case, we here need to handle the dependencies between every pair of adjacent and nonadjacent single clusters in an m-serial-cluster tool. These cluster interactions are also affected by all the intermediate single clusters. These dependences are quantitatively captured by the K_{ij}s and t_{v_i}s terms in the theorem.

For an m-serial cluster tool, according to K_{ij} formulation given in (4), the total number of n-CRM cycles is the same amount of index set (i, j) combinations, which is $m(m-1)/2$. That implies that the complexity of Algorithm 1 for m-serial cluster tools is polynomial as $O(m^2)$.

Remark 3: Algorithm 1 and Theorem 1 compute the minimal cycle time of an M-cluster and m-serial-cluster tools, respectively, under *a given schedule*. Finding the optimal robot sequences of the M-cluster tool under general configurations is NP-hard [8] and is out of scope of this work. For most practical applications, we are interested in finding the optimal schedules efficiently and effectively under certain conditions. In the following, we present these conditions under which optimal schedules of the M-cluster tool can be quickly found.

IV. Decoupling Conditions and Optimality for Robot Pull Strategies

In this section, we discuss conditions under which a multicluster tool can be decoupled into multiple single clusters so that existing efficient algorithms for single-cluster tools can be used to find the optimal schedules. We also derive optimality conditions for the robot pull strategy. Note that the results discussed in this section are the generalization of those in [2] under conditions of constant robot moving times and tree-like cluster topology.

Definition 1: Decoupling Equivalence (DE) is a property of an M-cluster tool with which the throughput of the tool is equal to the maximum throughput of the M decoupled single-cluster tools. Precisely, if an M-cluster tool possesses the DE property under schedule $\pi = (\pi_1, \ldots, \pi_M)$, its cycle time can be calculated as

$$T_{1M}(\pi) = \max_{1 \le i \le M} T_i(\pi_i), \quad (7)$$

where for each $\mathbb{C}_i$ connected to $\mathbb{C}_j$, $T_i(\pi_i)$ is calculated in a way that for a single-space buffer $P_{ib_{ij}}$, the virtual processing time is $t_{v_{ij}}$ (determined by treating $P_{ib_{ij}}$ in C-mode), while for a double-space buffer $t_{v_{ij}} = 0$.

When the DE property holds, all double-space buffers can be replaced by virtual PMs with zero processing times and all single-space buffers can be considered as virtual process modules with processing time $t_{v_{ij}}$. The results in [1, Th. 3] implies that a two-cluster tool with a double-space buffer module can be decoupled. However, if the buffer module is single space, the existence of the DE property depends on the timing data and the robot schedules. For brevity, we only consider the DE property under the single-space buffer case and buffer modules satisfy $b_{ij} \in \Lambda_i^P$. We can obtain the similar results if $b_{ij} \in \Lambda_i^R$. In the following, we first present the conditions under which the pull schedule is optimal for multicluster tool and then discuss the DE conditions.

Proposition 1: For $\mathbb{C}_i$ in a tree-like multicluster tool, the pull schedule π_i^p is optimal when $t_{il} \ge \delta_l, l = 1, \ldots, c_i, l \ne b_i$ and $4\epsilon_l + 3\delta_n \ge \delta_i$, $n > i$, where $\mathbb{C}_n$ is a cluster that is directly connected to $\mathbb{C}_i$.

Proof: See the Appendix ∎

In [2], only one-unit cycle has been considered. To rule out the n-CRM, the authors have proposed a sufficient condition. This sufficient condition is stated as [2, Proposition 2] and is to constrain the so-called minimal robot cassette waiting time t_i^R of $\mathbb{C}_i$ to be smaller than $\max_{1 \le i \le m} T_i(\pi_i)$ of an m-serial-cluster tool. We here show that the minimal robot cassette waiting time condition discussed in [2] indeed implies that the K_{ij} terms in (3) in Theorem 1 do not dominate other terms in (3). For simplicity, we show only for a single-wafer capacity buffer case and it is straightforward to extend to a double-wafer capacity buffer case.

Let us consider a two-cluster tool $\mathbb{C}_1$ and $\mathbb{C}_2$ with a single-wafer capacity buffer P_{1b_1} under the pull schedules (π_1^p, π_2^p).

Using the notation in this paper, the minimal robot cassette waiting time condition in [2] is given as

$$t_2^R \le \max\left\{T_1^0(\pi_1^p), T_2^0(\pi_2^p)\right\} - RCT^0(P_{1b_1}) - \beta_2^{\delta_2}, \quad (8)$$

where the minimal cassette waiting time $t_2^R = \max\left\{\alpha_2^{\min} - c_2\beta_2^{\delta_2}, 0\right\} + \beta_2^{\delta_2}$, $\alpha_i^{\min} = \min_{1\le l\le c_i}\alpha_{il}$, and $\alpha_i^{\max} = \max_{1\le l\le c_i}\alpha_{il}$, $i = 1, 2$. In [2], robot loading/unloading wafer times $\epsilon_i = T_i$, robot moving times $\delta_i = 0$, and therefore $\beta_i = \beta_i^{\delta_i} = 2\epsilon_i$, and $\alpha_{il} = 2\epsilon_i + t_{il}$.

Proposition 2: For a two-serial-cluster tool $\mathbb{C}_1$ and $\mathbb{C}_2$ under pull schedules (π_1^p, π_2^p) in [2], if the minimal cassette waiting time t_2^R satisfies (8), then the K_{12} term in Theorem 1 satisfies $K_{12} \le T_2^0(\pi_2^p)$, that is, K_{12} never dominates $T_2(\pi_2^p)$.

Proof: Under the pull schedule π_2^p, we have $T_2(\pi_2^p) = \alpha_2^{\max} + \beta_2^{\delta_2}$. We write the average one-unit cycle time K_{12} as

$$K_{12} = \frac{RCT^0(P_{1b_1}) + \beta_2^{\delta_2} + \sum_{l=1}^{c_2}\alpha_{2l}}{c_2}. \quad (9)$$

Since t_2^R satisfies (8), we have

$$\max\left\{\alpha_2^{\min} - c_2\beta_2^{\delta_2}, 0\right\} + \beta_2^{\delta_2} \le \alpha_2^{\max} + \beta_2^{\delta_2} - RCT^0(P_{1b_1}) - \beta_2^{\delta_2}$$

and thus

$$\alpha_2^{\min} + RCT^0(P_{1b_1}) + \beta_2^{\delta_2} \le \alpha_2^{\max} + c_2\beta_2^{\delta_2}. \quad (10)$$

Let $1 \le k \le c_2$ denote the PM index with the minimum processing time within $\mathbb{C}_2$, that is, $k = \arg_{1\le l\le c_2}\min\alpha_{2l}$. It is then straightforward to obtain the following inequality:

$$\sum_{l=1, l\ne k}^{c_2}\alpha_{2l} \le (c_2 - 1)\alpha_2^{\max}. \quad (11)$$

From (10) and (11), we have

$$\alpha_{2k} + RCT^0(P_{1b_1}) + \beta_2^{\delta_2} + \sum_{l=1, l\ne k}^{c_2}\alpha_{2l} \le \alpha_2^{\max} + c_2\beta_2^{\delta_2} + (c_2 - 1)\alpha_2^{\max}$$

and further obtain

$$RCT^0(P_{1b_1}) + \beta_2^{\delta_2} + \sum_{l=1}^{c_2}\alpha_{2l} \le c_2\left(\alpha_2^{\max} + \beta_2^{\delta_2}\right).$$

From the above inequality, we obtain

$$\frac{RCT^0(P_{1b_1}) + \beta_2^{\delta_2} + \sum_{l=1}^{c_2}\alpha_{2l}}{c_2} \le \alpha_2^{\max} + \beta_2^{\delta_2}.$$

Using (9), we obtain $K_{12} \le T_2^0(\pi_2^p)$, which completes the proof. ■

With Proposition 2, we obtain the following decomposition conditions for an M-cluster tool.

Corollary 1: For $\mathbb{C}_i$, $2 \le i \le M$, of an M-cluster tool under the pull schedule, suppose that out of c_i PMs, there are c_{b_i} buffer modules that connect $\mathbb{C}_i$ to downstream clusters. If a cluster $\mathbb{C}_k$ is connected to $\mathbb{C}_i$ from upstream and following condition is satisfied:

$$\alpha_i^{\min} + \beta_i^{\delta_i} - (c_i - c_{b_i})\beta_i \le \alpha_i^{\max} - RCT^0(P_{kb_{ki}}) \quad (12)$$

then clusters $\mathbb{C}_i$ and $\mathbb{C}_k$ can be decoupled. If condition (12) is satisfied for every $\mathbb{C}_i$, $2 \le i \le M$, then the entire M-cluster tool can be decoupled.

Remark 4: It is straightforward to apply the results in Corollary 1 to m-serial-cluster tools and obtain the decomposition conditions in [2]. We omit the derivation here for brevity. It is also clear that the results in Theorem 1 is an extension of these results in [2] with consideration of the n-CRM cycles among the multiclusters.

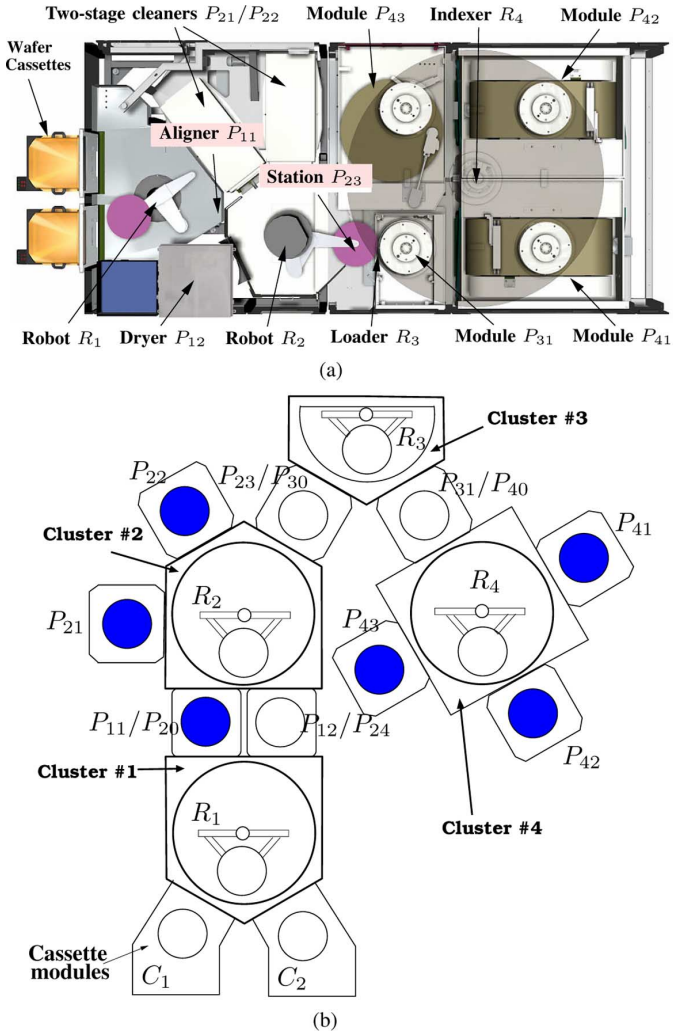

Fig. 7. Lam Teres chemical-mechanical planarization (CMP) polisher. (a) A top view of a 300-mm CMP polisher [12]. (b) A simplified CMP four-cluster tool.

V. Experimental Examples

We apply the results of this paper to a real production system. We consider a chemical-mechanical planarization (CMP) polisher used in semiconductor manufacturing. The CMP process is widely used to planarize the wafer surface and to enhance the photolithograph process performance [12]. The Teres CMP polisher (top view) shown in Fig. 7(a) was manufactured by Lam

TABLE V
FOUR-CLUSTER POLISHER PARAMETERS. ALL TIME PARAMETERS ARE IN SECONDS

Cluster	ϵ_i (s)	δ_i (s)	t_{il} (s)	c_i
$\mathbb{C}_1$	5	5	–	0
$\mathbb{C}_2$	5	5	$t_{21} = 90$, $t_{22} = 30$	2
$\mathbb{C}_3$	1	1	–	0
$\mathbb{C}_4$	1	0	$t_{41} = t_{42} = t_{43} = t_{44} = 60$	3

TABLE VI
RESOURCE CYCLE TIMES (SECONDS)

T_1	T_2	T_3	T_4	$RCT(P_{23})$	$RCT(P_{31})$	K_{23}	K_{34}	K_{24}	T_{14}
35	125	11	64	45	11	45	65	77	125

Research Corporation. Fig. 7(b) shows the cluster layout configuration for the polisher. The CMP polisher can be modeled as a four-cluster tool with four single-blade robots $R_i, i = 1, \ldots, 4$. A similar system was studied in [8] under an assumption of zero robot moving times. We here relax this assumption. The wafer flow is as follows:

$$C_1 \overset{R_1}{\to} P_{11}/P_{20} \overset{R_2}{\to} P_{23}/P_{30} \overset{R_3}{\to} P_{31}/P_{40} \overset{R_4}{\to} P_{41} \overset{R_4}{\to} P_{42} \overset{R_4}{\to} P_{43} \overset{R_4}{\to} P_{31}/P_{40} \overset{R_3}{\to} P_{23}/P_{30} \overset{R_2}{\to} P_{21} \overset{R_2}{\to} P_{22} \overset{R_2}{\to} P_{12}/P_{24} \overset{R_1}{\to} C_2.$$

The processing timing data are shown in Table V. Since $\mathbb{C}_3$ is a wafer-transfer cluster and $\mathbb{C}_4$ is a process cluster, the robot moving time and load/unload times in these two clusters are relatively shorter.

We apply Theorem 1 and the DE properties to obtain a simple closed-form formulation. From (3) and (4), under pull schedule $\pi^p = (\pi_1^p, \pi_2^p, \pi_3^p, \pi_4^p)$, we obtain the cycle time of the polisher as

$$\begin{aligned} T_{14}(\pi^p) &= \max\left\{T_1\left(\pi_1^p\right), T_{24}\left(\pi_2^p, \pi_3^p, \pi_4^p\right)\right\} \\ &= \max\Bigg\{T_1\left(\pi_1^p\right), T_2\left(\pi_2^p\right), T_3\left(\pi_3^p\right), \\ &\qquad T_4\left(\pi_4^p\right), K_{23}, K_{34}, K_{24}\Bigg\} \end{aligned}$$

where $T_1\left(\pi_1^p\right) = RCT(P_{11}), t_{v_3} = \beta_4^{\delta_4} + \beta_4, t_{v_2} = \beta_3^{\delta_3} + \beta_3 + t_{v_3}, T_2\left(\pi_2^p\right) = \max_{i=1,2,3}\{RCT(P_{2i}), RCT(R_2)\}, T_3(\pi_3) = RCT(P_{31}), T_4\left(\pi_4^p\right) = \max_{i=1,2,3}\{RCT(P_{4i}), RCT(R_4)\}$,

$$K_{23} = \frac{RCT(P_{23}) + T_3^F - RCT(P_{30})}{n_3},$$

$$K_{34} = \frac{RCT(P_{31}) + T_4^F - RCT(P_{40})}{n_4},$$

$$K_{24} = \frac{RCT(P_{23}) + T_3^F - RCT(P_{30}) + T_4^F - RCT(P_{40})}{n_3 + n_4 - 1}$$

and $n_3 = 1, n_4 = 3$.

The optimality conditions specified in Proposition 1 can be easily verified since $t_{il} \geq \delta_i, l = 1, \ldots, c_i, l \neq b_2, b_3, i = 2, 3, 4$, and $4\epsilon_{i+1} + 3\delta_{i+1} \geq \delta_i, i = 2, 3$. Therefore, the robot pull strategies are optimal schedules. Table VI lists all the calculated values for the resource cycle times. The final one-unit cycle time for the polisher is 125 s. Note that under the cluster tool data, the 3-CRM cycles do not dominate the cycle time and this cluster tool possesses the DE property.

We further perform a large number of Monte Carlo simulations to examine the effect of the uncertainties in processing, loading/unloading, and moving times on the DE property and the optimality of the pull schedule. The results are shown in Table VII. According to the guidelines in [13], we conduct seven experiments with exponentially decreasing mean processing times. Each experiment contains four simulations, which have exponentially decreasing variances in the processing time distribution. Every row in Table VII is a simulation experiment. For example, the third row is the simulation under uniform distribution with mean processing time 600 s and variance $(750 - 450)2/12 = 86.6$ s, which is one half of that of the simulation in the 2nd row or one quarter of that of the simulation in the 1st row. Each simulation experiment includes 1 million replications, all following the same distribution.

In all simulation experiments, $\epsilon_i, \delta_i \sim U(0.1, 10), i = 1, 2$ and $\epsilon_i, \delta_i \sim U(0.1, 1), i = 3, 4$. The first two columns in Table VII show the experimental parameters. The next seven columns list the percentage of dominance of each resource cycle. The last two columns present the percentage that the DE property holds and the percentage that the pull schedule is optimal, respectively. It can be seen that the DE property holds in almost 99% while the optimality of the pull schedule holds in almost 82% on average of all experiments. These results are due to the use of small robot moving and load/unload times. Fig. 8 shows that when both the robot moving and load/unload times change from $U(0.1, 1)$ to $U(0.1, 120)$ (increasing variations), the average percentage that the DE property holds and the average percentage that the pull schedule is optimal decrease quickly at the beginning but very slowly around 68% and 47%, respectively. In all of these experiments, we use our analytical results to predict the minimal cycle time of the polisher under the robot pull strategies.

VI. CONCLUSION

This paper extended the resource cycle-based method in the companion [1] to analyze the optimal scheduling problem for an M-cluster tool with single-blade robots. We presented a recursive algorithm to compute the minimal cycle time of the cluster tool for a set of given robot movement sequences. We demonstrated the impact of the cluster interactions on the cycle time. These interactions have been shown, in both closed-form formulation and numerical examples, not only between two adjacent clusters but also among multiple clusters within different cluster branches of a multicluster tool. Conditions were also established for decoupling a multicluster tool and verifying optimality of the pull schedule. Under the decoupling conditions, the multicluster tool scheduling problem can be solved in polynomial time, and if the processing times further satisfy the optimality conditions for the pull schedule, the optimal scheduling problem becomes straightforward. A CMP polisher in semiconductor manufacturing production was used as a multicluster tool example to illustrate the proposed formulation and algorithms. The Monte Carlo simulation results suggested that under most

TABLE VII
PROBABILITIES (IN PERCENTAGE) OF DE PROPERTY AND OPTIMALITY OF PULL SCHEDULES

Mean	$U(a,b)$	$T_1(\pi)$	$T_2(\pi)$	$T_3(\pi_3)$	$T_4(\pi_4)$	K_{23}	K_{34}	K_{24}	Prob(DE)	Prob(Pull)
	(0,1200)	0	43.716	0	56.229	0	0.055	0	99.9	99.9
	(300,900)	0	47.602	0	52.265	0	0.133	0	99.9	100
60×10	(450,750)	0	54.411	0	45.029	0	0.56	0	99.4	100
	(540,660)	0	70.607	0	27.586	0	1.807	0	98.2	100
	(0,600)	0	47.352	0	52.494	0	0.154	0	99.8	99.6
	(150,450)	0	54.326	0	45.142	0	0.532	0	99.5	100
60×5	(225,375)	0	66.346	0	32.18	0	1.474	0	98.5	100
	(270,330)	0	85.679	0	12.218	0	2.103	0	97.9	100
	(0,240)	0.001	57.484	0	41.822	0	0.693	0	99.3	97.7
	(60,180)	0	70.339	0	27.909	0	1.752	0	98.2	99.8
60×2	(90,150)	0	85.785	0	12.078	0	2.137	0	97.9	100
	(108,132)	0	96.185	0	2.967	0	0.848	0	99.2	100
	(0,120)	0.049	74.370	0	24.389	0	1.192	0	98.8	90.7
	(30,90)	0.022	87.045	0	11.342	0	1.591	0	98.4	91.1
60×1	(45,75)	0.010	94.818	0	4.129	0	1.043	0	99.0	88.6
	(54,66)	0.001	98.233	0	1.255	0	0.511	0	99.5	86.8
	(0,60)	1.889	90.480	0	7.162	0	0.469	0	99.5	68.7
	(15,45)	3.792	92.990	0	2.743	0	0.475	0	99.5	72.3
60×0.5	(22.5,37.5)	5.038	93.510	0	1.036	0	0.416	0	99.6	71.6
	(27,33)	5.906	93.356	0	0.374	0	0.364	0	99.6	71.4
	(0,24)	10.227	89.265	0	0.463	0.001	0.044	0	100	48.6
	(6,18)	10.770	88.970	0	0.207	0	0.053	0	100	68.4
60×0.2	(9,15)	10.912	88.918	0	0.094	0	0.076	0	99.9	70.8
	(10.8,13.2)	11.026	88.818	0	0.029	0	0.127	0	99.8	71.1
	(0,12)	10.997	88.903	0	0.069	0.021	0.010	0	100	31.9
	(3,9)	11.171	88.735	0	0.036	0.049	0.009	0	99.9	47.8
60×0.1	(4.5,7.5)	11.208	88.707	0	0.018	0.056	0.011	0	99.9	53.0
	(5.4,6.6)	11.173	88.717	0	0.011	0.075	0.024	0	99.9	56.0

processing time cases in practice, the decomposition and pull scheduling strategy achieve the minimal cycle time.

APPENDIX

We compare the pull schedule π_i^p with any other nonpull schedule π_i'. We consider a schedule $\pi^p = (\pi_1, \pi_2, \ldots, \pi_i^p, \ldots, \pi_M)$ and $\pi' = (\pi_1, \pi_2, \ldots, \pi_i', \ldots, \pi_M)$, where π_j, $j = 1, \ldots, M$ and $j \neq i$, can be any schedule for $\mathbb{C}_j$. The proposition implies that for any resource cycle CYC_A in π^p, we can find a corresponding resource cycle CYC_B in π' under which the cycle time is greater than or equal to that of π^p.

In the following, we first consider the cases where both CYC_A and CYC_B are 1-CRM cycles, and then the n-CRM cycles $(n > 1)$. Taking π' as a reference, there exist following cases of CYC_A in π^p.

Case 1: If $CYC_A = (RCT(P_{ik}), 1, \text{Buff}(P_{ik}))$ in π_i^p, where $k \in \Lambda_i^P$ of schedule π_i', then we can find the corresponding 1-CRM cycle $CYC_B = (RCT(P_{ik}), 1, \text{Buff}(P_{il}))$, $l \in UA(P_{ik})$ of π_i' such that

$$\begin{aligned} CT(CYC_B) &= RCT(P_{ik}) = (|IA(P_{ik})| - |UA(P_{ik})|)\beta_i \\ &\quad + \sum_{l \in UA(P_{ik})} \alpha_{il} \\ &= \beta_i + \alpha_{ik} + \sum_{l \in UA(P_{ik}), l \neq k} \alpha_{il} \\ &\geq \beta_i + \alpha_{ik} = CT(CYC_A) + (|IA(P_{ik})| \\ &\quad - |UA(P_{ik})| - 1)\beta_i. \end{aligned}$$

Note that in the above derivation, we use [1, eq. (3)] to calculate the resource cycle time of a PM. To obtain the inequality in the above equation, we use the fact that $|IA(P_{ik})| - |UA(P_{ik})| \geq 1$ by [1, eq. (3)].

Case 2: If $CYC_A = (RCT(P_{ik}), 1, \text{Buff}(P_{ik}))$, where $k \in \Lambda_i^R$ of schedule π_i', then we can find the corresponding 1-CRM cycle $CYC_B = (RCT(R_i), 1, \text{Buff}(P_{ik}))$ such that

$$\begin{aligned} CT(CYC_B) &= (c_i + 1 - |\Lambda_i^R|)\beta_i + \sum_{l \in \Lambda_i^R} \alpha_{il} \\ &= \beta_i + \alpha_{ik} + (c_i - 1)\beta_i \\ &\geq \beta_i + \alpha_{ik} = CT(CYC_A) + \sum_{l \in \Lambda_i^R, l \neq k} (\alpha_{il} - \beta_l). \end{aligned}$$

Note that in the above derivation, we use [1, (4)] to calculate the resource cycle time of a robot. We obtain the inequality using the facts $c_i \geq 1$ and $\alpha_{il} - \beta_i \geq 0$.

Case 3: If $CYC_A = (RCT(R_i), 1, \emptyset)$ in schedule π_i^p, then we can find the corresponding 1-CRM cycle $CYC_B = (RCT(R_i), 1, \text{Buff}(R_i \text{ in } \pi_i))$ such that

$$\begin{aligned} CT(CYC_B) &= \left(c_i + 1 - \left|\Lambda_i^R\right|\right)\beta_i + \sum_{l \in \Lambda_i^R} \alpha_{il} \\ &= (c_i + 1)\beta_i + \sum_{l \in \Lambda_i^R} (\alpha_{il} - \beta_i) \\ &\geq (c_i + 1)\beta_i = CT(CYC_A) \end{aligned}$$

by the fact that $t_{il} \geq \delta_i$ and thus $\alpha_{il} - \beta_i \geq 0$.

Note that $4\epsilon_l + 3\delta_l$ is the virtual processing time of $P_{ib_{il}}$ (the buffer module between $\mathbb{C}_i$ and $\mathbb{C}_l$ and also P_{l0} of $\mathbb{C}_l$) if $P_{l0} \in \Lambda_l^P$. The condition $4\epsilon_l + 3\delta_l \geq \delta_i$ implies that virtual processing time $t_{v_{il}}$ of $P_{ib_{il}}$ satisfies $t_{v_{il}} \geq \delta_i$, and therefore $t_{v_{il}}$ can be used to replace t_{ik}, where $k = b_{il}$ for the buffer module.

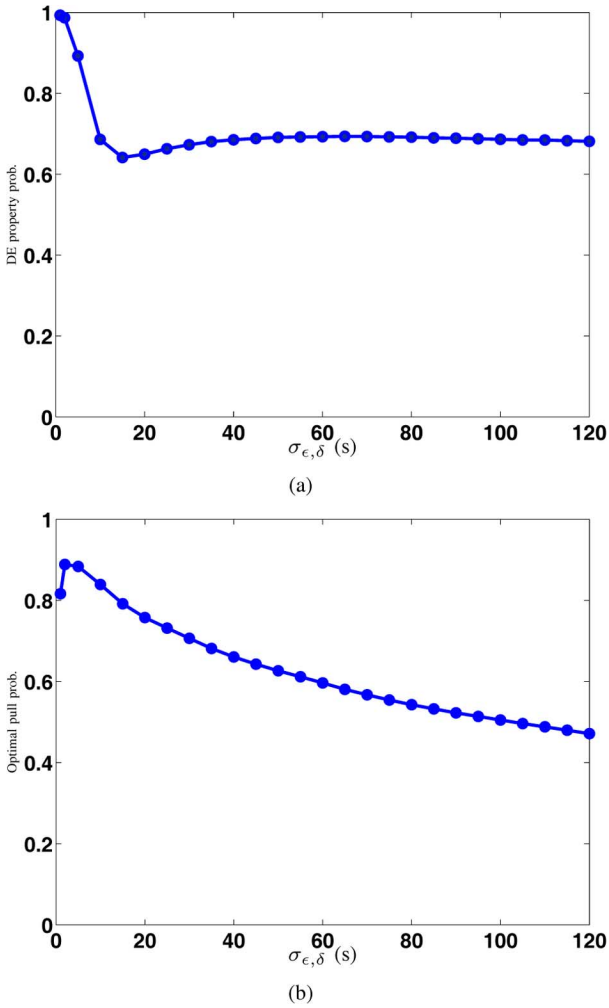

Fig. 8. Sensitivity analyses of DE property and pull optimality versus the variation of robot moving and load/unload time.

Within any n-CRM ($n > 1$) cycle $CYC_{k'}$, there is at least one cluster that is connected to an E-mode buffer. We define the set of indexes of all such clusters as the *index set of expanded clusters*, denoted as $\mathbb{E}(CYC_{k'})$. Next, we check all the n-CRM cycles in which $\mathbb{C}_i$ is involved.

Case 4: For any n-CRM resource cycle CYC_A in π^p that $\mathbb{C}_i$ is the first cluster, we find the corresponding CYC_B in π' as follows: Step 1: find the base resource cycle of CYC_A, denoted as ${}^1CYC_A$, where ${}^1CYC_A$ is a 1-CRM cycle of $\mathbb{C}_i$; Step 2: according to Cases 1–3 above, we can find the corresponding ${}^1CYC_B$ of ${}^1CYC_A$; Step 3: we expand the buffer modules of ${}^1CYC_B$ to include all $\mathbb{C}_j$ that are in CYC_A. Then, $CYC_A = \left(T({}^1CYC_A)+T', 1+n', \{P_{ib_i}, B'\}\right)$ and $CYC_B = \left(T({}^1CYC_B)+T', 1+n', \{P_{ib_i}, B'\}\right)$, where $T' = \sum_{j\in\mathbb{E}({}^1CYC_B)}(T_j^F - RCT(P_{j0}))$, $n' = \sum_{j\in\mathbb{E}({}^1CYC_B)}(n_j-1)$, $B' = \{\text{all buffer modules in } \mathbb{C}_j\}$, $j \in \mathbb{E}\left({}^1CYC_B\right)$, and $j \neq i$. Since $RCT(P_{ib_i} \text{in } \pi_i^p) \leq RCT(P_{ib_i} \text{in } \pi_i')$ by the above proofs in Cases 1–3, thus we have

$$\frac{RCT_{\pi_i^p}(P_{ib_i}) + T'}{1+n'} \leq \frac{RCT_{\pi_i'}(P_{ib_i}) + T'}{1+n'},$$

which implies the average one-unit cycle time $CT(CYC_A) \leq CT(CYC_B)$.

Case 5: If $\mathbb{C}_i$ is not the root cluster of the M-cluster tool, then there is a cluster, say $\mathbb{C}_s$, connected to $\mathbb{C}_i$. The corresponding buffer module is then $P_{sb_{si}}$.

i) If an n-CRM CYC_A in π^p has $P_{sb_{si}}$ in the E-mode, we find the corresponding CYC_B in π' as follows: Step 1: find ${}^1CYC_A$ in π^p; Step 2: set ${}^1CYC_B = {}^1CYC_A$, where ${}^1CYC_B$ is in π'; Step 3: expand the buffer modules of ${}^1CYC_B$ to include all $\mathbb{C}_j$ that are in CYC_A. Since the schedules of all other cluster (except for $\mathbb{C}_i$) are the same in π' and π^p, let $T' = CT\left({}^1CYC_A\right) + \sum_{j\in\mathbb{E}({}^1CYC_B)}\left(T_j^F - RCT(P_{j0})\right)$, $n' = 1+\sum_{j\in\mathbb{E}({}^1CYC_B)}(n_j-1)$, $B' = \{B({}^1CYC_A),$ all buffer modules in$\mathbb{C}_j\}$, $j \in_{j\in\mathbb{E}({}^1CYC_B)}$, and $j \neq i$. We have $CYC_A = \left(T' + T_{i\pi^p}^F - RCT(P_{i0}), n' + (c_i - 1), \{B', \text{Buff}(P_{il})\}\right)$ and $CYC_B = \left(T' + T_{i\pi^p}^F - RCT(P_{i0}), n' + (c_i - |\Lambda_i^R| - 1), \{B', \text{Buff}(P_{il})\}\right)$. According to (1), $T_i^F\left(\pi^p\right) = T_i^F\left(\pi'\right) =: T_i^F$. Then, we have

$$\frac{T' + T_i^F}{c_i - 1} > \frac{T' + T_i^F}{c_i - |\Lambda_i^R| - 1}$$

and thus $CT(CYC_A) \leq CT(CYC_B)$.

ii) If CYC_A is the 1-CRM cycle of $P_{sb_{si}}$ in π^p, then we can find in π' with the corresponding CYC_B as the 1-CRM cycle of $P_{sb_{si}}$. This means that $P_{sb_{si}}$ is considered as in the C-mode in both schedules. There are two possible scenarios: (a) In π', $P_{i0} \in \Lambda_i^P$, then in both schedules, $RCT(P_{i0}) = 4\epsilon_i + 3\delta_i$ and thus $CT(CYC_A) = CT(CYC_B)$. (b) In π', $P_{i0} \in \Lambda_i^R$, then $RCT(P_{i0}) > 4\epsilon_i + 3\delta_i$ in π'. Then, we obtain $CT(CYC_A) < CT(CYC_B)$.

All possible cases are analyzed and this completes the proof.

REFERENCES

[1] W.-K. Chan, J. Yi, and S. Ding, "Optimal scheduling of multicluster tools with constant robot moving times, Part I: Two-cluster analysis," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 1, pp. 5–16, Jan. 2011.

[2] J. Yi, S. Ding, D. Song, and M. T. Zhang, "Steady-state throughput and scheduling analysis of multi-cluster tools: An decomposition approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 2, pp. 321–336, Apr. 2008.

[3] T. Perkinson, P. McLarty, R. Gyurcsik, and R. Cavin, "Single-wafer cluster tool performance: An analysis of throughput," *IEEE Trans. Semiconduct. Manuf.*, vol. 7, pp. 369–373, 1994.

[4] M. Dawande, C. Sriskandarajah, and S. Sethi, "On throughtput maximization in constant travel-time robotic cells," *Manuf. Serv. Oper. Manage.*, vol. 4, no. 4, pp. 296–312, 2002.

[5] P. van der Meulen, "Linear semiconductor manufacturing logistics and the impact on cycle time," in *Proc. 18th Ann. IEEE/SEMI Adv. Semiconduct. Manuf. Conf.*, Stresa, Italy, 2007, pp. 111–116.

[6] J. Yi, M. T. Zhang, S. Ding, and P. van der Meulen, "Throughput analysis of linear cluster tools," in *Proc. IEEE Conf. Autom. Sci. Eng.*, Scottsdale, AZ, 2007, pp. 1063–1068.

[7] N. Geismar, C. Sriskandarajah, and N. Ramanan, "Increasing throughput for robotic cells with parallel machines and multiple robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 1, no. 1, pp. 84–89, Jan. 2004.

[8] S. Ding, J. Yi, and M. T. Zhang, "Multi-cluster tools scheduling: An integrated event graph and network model approach," *IEEE Trans. Semiconduct. Manuf.*, vol. 19, pp. 339–351, 2006.

[9] C. Sriskandarajah, I. Drobouchevitch, S. Sethi, and R. Chandrasekaran, "Scheduling multiple parts in a robotic cell served by a dual-gripper robot," *Opers. Res.*, vol. 52, no. 1, pp. 65–82, 2004.
[10] N. Wu, C. Chu, F. Chu, and M. Zhou, "A Petri net method for schedulability and scheduling problems in single-arm cluster tools with wafer residency time constraints," *IEEE Trans. Semiconduct. Manuf.*, vol. 21, pp. 224–237, 2008.
[11] J.-H. Kim and T.-E. Lee, "Schedulability analysis of time-constrained cluster tools with bounded time variation by an extended Petri net," *IEEE Trans. Automat. Sci. Eng.*, vol. 5, no. 3, pp. 490–503, Jul. 2008.
[12] J. Yi, "Friction modeling in linear chemical-mechanical planarization: Process monitoring for wafer polishing in semiconductor manufacturing," *IEEE Control Syst. Mag.*, vol. 28, no. 5, pp. 59–78, 2008.
[13] Y. Hall and M. Posner, "Generating experimental data for computational testing with machine scheduling applications," *Opers. Res.*, vol. 49, no. 6, pp. 854–865, 2001.

Wai Kin Victor Chan (M'05) received the B.Eng. degree from Shanghai Jiao Tong University, Shanghai, China, the M.Eng. degree in electrical engineering from Tsinghua University, Beijing, China, and the M.S. and Ph.D. degrees in industrial engineering and operations research from the University of California, Berkeley, in 2001 and 2005, respectively.

He is an Assistant Professor with the Department of Industrial and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. His research interests include discrete-event simulation, agent-based simulation, and their applications in energy markets, social networks, service systems, and manufacturing.

Shengwei Ding received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, China, in 1996 and 1999, respectively, and the Ph.D. degree in industrial engineering and operations research from the University of California, Berkeley, in 2004.

He is currently with the Direct Store Delivery (DSD) Division of Nestle USA as a member of the supply chain integration (SCI) team. From 2007 to 2009, he worked at Leachman and Associates LLC, a firm providing consulting and software for operations management to semiconductor manufacturers and other corporations. Prior to that, he was a Postdoctoral Fellow at the University of California, Berkeley. His research interests include optimization, queueing models, simulation, planning and scheduling, production management, and semiconductor manufacturing.

Jingang Yi (S'99–M'02–SM'07) received the B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 1993, the M.Eng. degree in precision instruments from Tsinghua University, Beijing, China, in 1996, and the M.A. degree in mathematics and the Ph.D. degree in mechanical engineering from the University of California, Berkeley, in 2001 and 2002, respectively.

He is currently an Assistant Professor of Mechanical Engineering at Rutgers University. Prior to joining Rutgers University in August 2008, he was an Assistant Professor of Mechanical Engineering at San Diego State University since January 2007. From May 2002 to January 2005, he was with Lam Research Corporation, Fremont, CA, as a member of Technical Staff. From January 2005 to December 2006, he was with the Department of Mechanical Engineering, Texas A&M University, as a Visiting Assistant Professor. His research interests include autonomous robotic systems, dynamic systems and control, mechatronics, automation science and engineering, with applications to semiconductor manufacturing, intelligent transportation and biomedical systems.

Dr. Yi is a member of the American Society of Mechanical Engineers (ASME). He is a recipient of the NSF Faculty Early Career Development (CAREER) Award in 2010. He has coauthored papers that have been awarded the Best Student Paper Award Finalist of the 2008 ASME Dynamic Systems and Control Conference, the Best Conference Paper Award Finalists of the 2007 and 2008 IEEE International Conference on Automation Science and Engineering, and the Kayamori Best Paper Award of the 2005 IEEE International Conference on Robotics and Automation. He currently serves as an Associate Editor of the ASME Dynamic Systems and Control Division and the IEEE Robotics and Automation Society Conference Editorial Boards. He also serves as a Guest Editor of IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.

Dezhen Song (S'02–M'04–SM'09) received the B.S. and M.S. degrees from Zhejiang University, Zhejiang, China, in 1995 and 1998, and the Ph.D. degree from the University of California, Berkeley, in 2004.

He is an Assistant Professor with Texas A&M University, College Station. His primary research area is networked robotics, computer vision, optimization, and stochastic modeling.

Dr. Song received the Kayamori Best Paper Award at the 2005 IEEE International Conference on Robotics and Automation (with J. Yi and S. Ding) and the NSF Faculty Early Career Development (CAREER) Award in 2007. He Co-Chaired the IEEE Robotics and Automation Society (RAS) Technical Committee on Networked Robots from 2007 to 2009. He is an Associate Editor of the IEEE TRANSACTIONS ON ROBOTICS and the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.