Multiclass Flow Line Models of Semiconductor Manufacturing Equipment for Fab-Level Simulation

James R. Morrison, Member, IEEE

Abstract—For multiclass flow line models, we identify a class of service times that allow a decomposition of the system into subsets of servers called channels. In each channel, the customer delay is well structured and we develop a recursion to calculate it. The recursions provide an alternative to the elementary evolution equations. By considering batch arrivals and restricting the structure of the model, the recursions can require nearly one order of magnitude less computation than is otherwise possible.

Flow lines can be used as models for semiconductor manufacturing equipment such as multicluster or clustered photolithography tools. The models allow for internal wafer buffers and setups that are wafer location dependent. The models have shown to be very accurate in tests with data from clustered photolithography tools in production. As such, the models may serve as good candidates to improve the fidelity of existing equipment models in fab-level simulation.

Note to Practitioners—Clustered photolithography and multicluster tools in semiconductor manufacturing feature internal wafer buffers and setups that depend on wafer location. These features are important to system performance when product diversity and small wafer lots are common as in high-mix fabs and in the anticipated 450 mm fabs. Unfortunately, existing fab-level simulation models do not include these equipment features.

We develop flow line models that allow for diverse products, wafer lots and wafer location dependent setups. The models developed can require about one order of magnitude less computation than otherwise possible to model practical tools. In addition, the models have given throughput and cycle time values to within 1.0% and 4.0% of actual values from production clustered photolithography tools. As such, a flow line can serve to replace existing equipment models in fab-level simulators with reasonable computational cost while retaining high accuracy.

Index Terms—Fab-level simulation, flow line, photolithography cluster tools, semiconductor manufacturing automation.

I. INTRODUCTION

The anticipated proliferation of lots with fewer wafers and increase in product diversity in semiconductor wafer fabricators (fabs) will present challenges for operations, planning and performance evaluation [1]. Operationally, more setups will be required due to diversity and AMHS traffic will increase due to smaller lot sizes. Planning and performance evaluation of fabs will be complicated as current equipment models seldom incorporate features that will increase in prominence. To address some of these challenges, we develop expressive equipment models for use in fab simulation. The models can require about one order of magnitude less computation than is otherwise possible.

Simulation models are often used to model fab behavior and answer questions related to changes in equipment capacity, operations, material handling automation, lot scheduling, maintenance priorities, etc. State-of-the-art simulation software allows for fairly rich equipment models that include features such as first wafer delay, throughput rate after the first wafer, batch sizes, setups, sampling, failure and repair. For many tools in the current environment these models are sufficient.

However, coupled with a trend toward the use of cluster tools, the anticipated product mix changes may result in significantly less accurate simulation models. The reason is that tools will spend more time in transient operation. Two key features that are not included in equipment models for fab-level simulation, but play an essential role in transient behavior, are internal wafer buffers and setups dependent on wafer location. Such features can be common in clustered photolithography tools and linked cluster tools [2].

One challenge to incorporating additional features is computational complexity. The more expressive the model, the greater the computation required. So long as accurate predictions of throughput can be obtained, one is not generally concerned with the details of wafer transport robot movement in a fab-level simulation. A first modeling assumption is thus to ignore wafer transport robots. For clustered photolithography tools with deterministic process times, the resulting system can be modeled as a deterministic flow line so long as it is not robot-bound.

Flow lines and related systems have been studied for many years, see, for example, [3]–[4] and [5]–[8]. Due to the intractability of analysis, approximate decomposition and aggregation methods are generally used. However, in some limited cases and with enough assumptions, exact analytic results are possible. The classic papers [9] and [10] obtained recursions for the exit time of wafers from a flow line with a single class of wafers. It is known that the throughput of a deterministic flow line with no setups and a single class of customer (or wafer in semiconductor manufacturing) is dictated by its bottleneck process time. To better understand transient behavior and reduce computation for simulation, [11] and [12] developed an exact system decomposition leading to recursions characterizing the internal advancement of wafers in a flow line. They incorporated wafer location dependent setups and
reduced the computational complexity by almost two orders of magnitude. However, they allow only a single class of wafers and are not generally applicable. Another approach for simulation is to use an approximate hybrid simulation model, see, for example, [13]–[15]; they can significantly reduce complexity while retaining accuracy.

Here, we study deterministic multiclass flow line models that allow for different process times dependent upon the wafer as in [16]. The goal is to develop equipment models for fab-level simulation that are both expressive and reasonably computationally tractable. We obtain the following results, which hold under assumptions on the process times detailed in the sequel.

• Flow lines can be exactly (not approximately) decomposed into segments called channels. In a channel, the wafer delay is structured and there is a recursion to calculate it (Theorems 1–3).
• Practical features such as wafer location dependent setups, buffers and wafer lots can be incorporated (Corollaries 1 and 2).
• Models with a single channel can allow a reduction in computation of about one order of magnitude (Theorem 4).
• The models can serve as approximations for practical flow lines that do not satisfy our assumptions and give accurate throughput and cycle time values (Section VII). Further, tests on a clustered photolithography tool in production have shown throughput and cycle time values within 1.0% and 4.0% of the actual values, respectively (Section VI-A).

While the results have a similar flavor as the single class case, there are significant differences. First, we identify a class of process times that allow an exact channel decomposition with multiple classes of wafers. While recursions for channel delay can be developed, there does not appear to be a simple recursion for exit times from the system. The recursions contain terms that account for the transition from one class to another. Finally, there is not as much computational advantage.

The models are expressive, accurate in tests and more computationally tractable than otherwise possible. They are thus good candidates for use in fab-level simulation that accounts for impending changes to the number of wafers per lot and increased product diversity.

Some of the results of this paper first appeared in abbreviated form and without proof in [17] and [18].

This paper is organized as follows. In Section II, we describe the flow line models under consideration. Wafer delay is considered in Section III. In Section IV, we incorporate the practical features of setups and wafer lots. Computational complexity and model fidelity are discussed in Sections V and VI, respectively. Numerical experiments are discussed in Section VII. Concluding remarks are presented in Section VIII.

II. MULTICLASS FLOW LINES

A flow line consists of a series of \( E \) modules, \( m_1, \ldots, m_E \), from which wafers must receive service in order. There may be a buffer of capacity \( b_i \) before module \( m_i \); the buffer before the first module is of infinite capacity \( b_1 = \infty \). Let \( F \) denote the internal buffer space, that is, \( F := b_2 + \cdots + b_E \). Wafers arrive as an arbitrary process, either singly or in batches called lots.

Wafers are served in first come first served (FCFS) manner from the buffers. It is easy to incorporate different wafer priorities for entrance to \( m_1 \), but for simplicity we use FCFS there too. There are \( K \) classes of wafers, \( 1, \ldots, K \), distinguished by their service times, also called process times. We use \( c_k(w) \) to denote the class of wafer \( w \). Let \( \tau^k_j \) denote the deterministic service time requirement for a wafer of class \( k \) in module \( m_j \).

An arriving wafer \( w \) waits in the queue, if there is one, until module \( m_1 \) is vacant. At that instant, wafer \( w \) immediately enters service with module \( m_1 \). Once wafer \( w \) has received service from module \( m_1 \), it next advances to module \( m_{k+1} \) if all preceding wafers have vacated that module. In that case, it begins service immediately. If it is not, and there is an available slot in that module’s buffer, wafer \( w \) will enter the buffer. If there is no slot available, the wafer languishes in module \( m_{k+1} \) advances. If \( i = E \), the wafer exits the system. Using the terminology of [3], the system is an asynchronous flow line with manufacturing blocking. Such a flow line is depicted in Fig. 1.

As discussed in [9] and [12], buffers can be modeled as process modules with zero process time. This is because the time a wafer spends in a buffer is the same in both models. In a model with a buffer slot, a wafer will spend zero time in the slot if buffering is not required. In a model with a zero process time server, the wafer spends zero time in the module if buffering is not required. Similarly, for when the buffer is needed. The occupancy time is the same in both cases. We thus hereafter consider that a flow line consists of \( M := E + F \) process modules, with the ones corresponding to buffers having \( \tau^k_j = 0 \).

Let \( a_{w,j} \) denote the arrival instant of wafer \( w \); suppose \( a_{w,j+1} \geq a_{w,j} \). All wafers in a lot arrive at the same instant and are served in the order of their wafer index \( w \). Let \( X_{w,j} \) denote the entry time of wafer \( w \) to module \( m_j \). Let \( C_{w,j} := X_{w,j} + \tau^j_k(w) \) denote the time of service completion of wafer \( w \) from module \( m_j \). The system obeys the elementary evolution equations,

\[
X_{w,1} = \max\{a_{w,1}, X_{w-1,2}\}
\]
\[
X_{w,j+1} = \max\{C_{w,j}, X_{w-1,j+2}\}
\]
\[
X_{w,M} = \max\{C_{w,M-1}, C_{w,M}\}
\]

for all \( j = 1, \ldots, M-2 \) with the initial conditions \( X_{0,j} = -\infty \). The difficulty with the elementary evolution equations is computational intensity—one must calculate the advance of each wafer through every module.

Definitions and assumptions follow.

Assumption AI: Process Times Between Classes: The process times satisfy \( \tau^j_k(w) \geq \eta_k \tau^j_j, \) for all \( j = 1, \ldots, M \) and \( k = 1, \ldots, K-1. \) Each \( \eta_k \in (0,1). \) Let \( \eta := \eta_1 \cdots \eta_{K-1}. \) The bottleneck module \( m_B \) satisfies \( \tau^j_B \geq \tau^j_B \) for all \( j \) and \( B > \eta_j \) for all \( j \). Under Assumption AI it is the same
module for all classes. In the scheduling literature, a system satisfying Assumption A1 is called a proportional-machine flow line. It means, for example, that the process times for any class are proportional to those in class 1. \( \eta_k \) is the constant by which all class \( k \) process times are multiplied to give process times for class \( k+1 \).

**Definition: Dominating Modules**: For wafer class \( k \), the modules for which \( \tau_j^k > \tau_j^k \), for all \( \ell < j \), are called dominating modules.

Under Assumption A1, the dominating modules are the same for each class. By convention, module \( m_1 \) is the first dominating module. \( m_B \) is the last. Let \( \sigma \) denote their number and use \( \alpha \) as their index, so that \( \alpha = 1, \ldots, \sigma \). We use the notation \( \beta(\alpha) \) to denote the module index of dominating module \( \alpha \). Thus, \( \beta(1) = 1 \) and \( \beta(\sigma) = B \).

**Definition: Channel**: For \( \alpha = 1, \ldots, \sigma - 1 \), the modules \( m_{\beta(\alpha)}, \ldots, m_{\beta(\alpha+1)} \) form a channel. Call it channel-\( \alpha \).

**Assumption A2: Process Times Within a Class**: The process times for class \( k \) wafers satisfy \( \tau_j^k \leq (\eta_j)^{\beta(\alpha+1)} \), for \( \beta(\alpha) < j < \beta(\alpha+1) \), for all \( \alpha = 1, \ldots, \sigma - 1 \). After the bottleneck module \( \tau_j^k \leq (\eta_j)^{\beta(B)} \), for \( j > B \).

Note that in particular, this implies that \( \tau_j^k \leq (\eta_j)^{\beta(B)} \), for \( \beta(\alpha) < j < \beta(\alpha+1) \). After the bottleneck module \( \tau_j^k \leq (\eta_j)^{\beta(B)} \), for \( j > B \). This also means that there is no module after the bottleneck with \( \tau_j^k = \tau_B^k \). We now demonstrate these assumptions and definitions.

**Example 1**: Consider a deterministic multiclass flow line with three wafer classes and \( M = 9 \) modules. There are three dominating modules \( m_1, m_4, m_7 \). The bottleneck is \( m_7 \) so that \( B = 7 \). The bottleneck process times are \( \tau_B^k = 60 \), \( \tau_B^0 = 40 \) and \( \tau_B^3 = 30 \). Suppose that Assumption A1 holds, and let \( \eta_1 = \tau_B^1/\tau_B^1 = 2/3 \) and \( \eta_2 = \tau_B^3/\tau_B^3 = 3/4 \). The product \( \eta = \eta_1 \cdot \eta_2 = 0.5 \). Process times satisfying Assumptions A1 and A2 are given in Table I.

Throughout the paper, when we write sums, if the upper limit is less than the lower one, we consider the sum to be 0. For example, \( \sum_{j=0}^{\ell} \alpha^j = 0 \), for \( \ell < 0 \).

### III. WAVER DELAY

We are now poised to develop recursions for wafer completion time from the system and delay in the channels. Throughout this section we consider an initially empty deterministic multiclass flow line satisfying Assumptions A1 and A2. In addition, we let the bottleneck process times be subject to wafer dependent increases. That is, we replace the bottleneck process times with \( \tau_B^k(w) := \tau_B^k(\omega) + s(w) \), where \( s(w) \geq 0 \). Such increases may correspond to setup times at or after the bottleneck when changing from wafer to wafer. These setups are common for the bottleneck of the clustered photolithography tool, the photolithography scanner. They can occur due to reticle alignment when changing from one class of wafer to another.

In [9], a succinct recursion for the completion time of wafers from a single-class flow line is developed. The result does not hold in general. However, under Assumptions A1 and A2, we can show that there is no contention (wafers never are delayed) after the bottleneck. For simplicity, let \( T_B^k(w) := s(w) \cdot I_{[\tau_B^k(w) \geq \tau_B^k(\omega)]} + \sum_{j=0}^n \tau_j^k(\omega) \cdot p \cdot I_{[\tau_j^k(w)]} \cdot p \cdot I_{[\tau_j^k(w)]} \), where \( p < q \) and \( I_{\{\}} \) is the indicator function for the condition \{\}. Here, as throughout the paper, we relegate all proofs to the Appendix.

**Lemma 1: Start Times After the Bottleneck**: The start times for \( w \geq 1 \) satisfy:

- \( X_{w,B+1} = X_{w,B} + \tau_B^k(w) \)
- \( X_{w,j+1} = X_{w,j} + \tau_j^k(\omega) \), for all \( B < j < M \).

Also, \( C_{w,M} = X_{w,M} + \tau_M^k(w) + I_{[M-B]} \cdot s(w) \).

**Lemma 2: Time Separation Between Wafers in Post-Bottleneck Modules**: The following hold for \( w \geq 1 \):

- \( X_{w,j+1} \geq X_{w,j} + \tau_B^k(w+1) + \tau_B^k(w+1) - T_B^k(w+1) + \tau_B^k(w+1) \), for all \( B < j < M \).
- \( C_{w,M} \geq X_{w,M} + \tau_B^k(w + 1) + [T_B^k(w+1) + \tau_B^k(w+1)] \).

We refer to Lemmas 1 and 2 as "no contention after the bottleneck" results; there is no delay after the bottleneck. Modules \( m_{B+1}, \ldots, m_M \) can be removed and accounted for by simply adding a fixed postprocessing time equal to \( T_B^k(1) \) for each wafer. Lemma 2 essentially states that the interval between wafer entry to the post-bottleneck series of modules is not less than the bottleneck process time. Since, different classes of wafers may then advance at different rates (though they never are delayed), the entry time in a module from one wafer to the next can differ from the bottleneck process time.

In many cluster tools, such as clustered photolithography tools, the start time of a setup may depend upon the location of wafers in the tool. For example, in a multicluster tool [12], [19], a setup for one of the clusters may wait until all wafers have exited the cluster. In a clustered photolithography tool, the setup may wait for all wafers to vacate the initial modules. We thus require visibility to the advancement of wafers.

We next show that if a deterministic multiclass flow line satisfying Assumptions A1 and A2 consists of a single channel the delay is well structured. Process times for a single channel flow line are depicted in Fig. 2. This structure allows one to develop a recursion for the total wafer delay inside the channel.

Let \( d_{w,j} := X_{w,j+1} - X_{w,j} - \tau_j^k(\omega) \) denote the queuing wafer \( w \) suffers, while in module \( m_j \), \( j = 1, \ldots, B = 1 \). Use \( Y(w) \) to denote the total wafer delay inside the system, that is, \( Y(w) := \sum_{j=1}^{B} d_{w,j} \). As there is no delay in modules \( m_j, j \geq B \), by Lemma 1, we do not include \( j \geq B \) terms in the sum. Let

\[
S_B^k(w) := \sum_{j=0}^{B-1} \left[ \tau_B^k(w - B + j) - \tau_j^k(w) \right]
\]

where \( \tau_B^k(w) := \tau_B^k(1), \forall w \leq 0 \). This sum serves as an upper bound on the total delay that wafer \( w \) can experience in modules \( m_k, \ldots, B = 1 \). The idea is that when the channel is filled with wafers, wafers prior to the bottleneck can advance one step more. 


<table>
<thead>
<tr>
<th>Class</th>
<th>( \tau_1^k )</th>
<th>( \tau_2^k )</th>
<th>( \tau_3^k )</th>
<th>( \tau_4^k )</th>
<th>( \tau_5^k )</th>
<th>( \tau_6^k )</th>
<th>( \tau_7^k )</th>
<th>( \tau_8^k )</th>
<th>( \tau_9^k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>36</td>
<td>18</td>
<td>9</td>
<td>50</td>
<td>25</td>
<td>12.5</td>
<td>60</td>
<td>30</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>24</td>
<td>12</td>
<td>6</td>
<td>100</td>
<td>3</td>
<td>100</td>
<td>12</td>
<td>40</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>18</td>
<td>9</td>
<td>4.5</td>
<td>25</td>
<td>12.5</td>
<td>6.25</td>
<td>30</td>
<td>15</td>
<td>7.5</td>
</tr>
</tbody>
</table>
every time the bottleneck completes service. Thus, the maximum delay in module $m_i$ is $\tau_B(w + B + \ell) - \tau_\ell^*(w)$. Use $(\cdot)^+$ to denote $\max\{\cdot, 0\}$.

**Theorem 1: The Structure of Delay:** Consider a flow line under the assumptions of this Section. Further, there is a single channel, so that $\tau_1(w^*) > \tau_j(w^*)$, for all $j < B$, and the last module is the bottleneck. If some $d_{wij} \neq 0, 1 \leq j < B$, then
- $d_{wij} = 0$, for $j < j^*$;
- $d_{wij} \in (0, \tau_B(w - B + j^*) - \tau_j^*(w)]$;
- $d_{wij} = \tau_B(w - B + j^*) - \tau_j^*(w)$, for $j > j^*$;
for some module index $j^* < B$. If $j^* = 1$, there is delay in all modules. If $j^* = B - 1$, there is only delay in module $m_j$.

**Theorem 2: Recursion for Delay in a Single Channel Multiclass Flow Line:** Consider a flow line as in Theorem 1. For wafer $w \geq 1$, the total delay in the channel obeys
\[
Y(w) = \min \left\{ \Delta^B(w), Y(w - 1) + \tau_B(w - 1) - \max \{\tau_1(w - 1), a_w - X_{w-1,1}\} + T_{B-1}(w - 1) - T_{B-1}(w) \right\}^+
\]
with the initial conditions $Y(0) := 0$, $\tau_1^*(0) := \tau_1(1)$, and $\tau_B(0) := \tau_B(1)$. The delay in a module can be calculated directly from $Y(w)$ as
\[
d_{wij} = \min \left\{ \tau_B(w - B + j) - \tau_j^*(w), \quad Y(w) - S_{j+1}(w) \right\}^+
\]
for $j = 1, \ldots, B - 1$. The start times obey
\[
X_{w,1} = \max \left\{ a_w, X_{w-1,1} + \tau_1^*(w - 1) + d_{w-1,1} \right\}
\]
with initial condition $X_{0,1} = -\infty$ and $d_{0,1} = 0$. The queuing to enter module $m_1$ is $d_{w,0} := X_{w-1,1} - a_w$.

Thus, without resorting to the elementary evolution equations, one can determine the total delay the next wafer will experience and exactly where that delay will occur. This is possible without simulating the advancement of each wafer through every module.

By definition $C_{w,M} = a_w + d_{w,0} + Y(w) + T_{B}(w)$. Based on (1), this can be considered a recursion. It is more complicated than for a single class system.

**Example 2: Delay in a Channel:** Consider a flow line with two classes of wafers, four modules and deterministic process times. Let $\tau_1^2 = 36, \tau_2^2 = 18, \tau_3^2 = 9$, and $\tau_4^2 = 50$. Let $\tau_1^3 = 18, \tau_2^3 = 9, \tau_3^3 = 4.5$, and $\tau_4^3 = 25$. The delays each arriving wafer faces in the modules are given in Table II. There $w$ is the wafer index and $k$ is the class. Note that the structure of Theorem 1 holds. If there is delay inside the channel, modules after the first module in which delay occurs have the maximum delay. Theorem 2 gives a succinct recursion for calculating delay as an alternative to the elementary evolution equations.

For the remainder of this section, we focus on flow lines with at least two channels. With appropriate definition of arrival times and bottleneck process times, Theorem 2 can be applied to any channel in a multichannel flow line. Thus, the channels serve as a decomposition of the system.

Let $Y_{\alpha}(w)$ denote the delay experienced by wafer $w$ in channel-$\alpha$, excluding delay in the final module. That is, let
\[
Y_{\alpha}(w) := \sum_{\ell=\beta_{\alpha}}^{\beta_{\alpha}+1} d_{\alpha,\ell}.f.
\]
Similarly, let
\[
S_{p,\alpha}(w) := \sum_{\beta_{\alpha}}^{\beta_{\alpha}+1} \left[ \tau_{\beta_{\alpha}}(w - \beta_{\alpha} + 1) - \tau_{\ell}^*(w) \right],
\]
where $\tau_{\beta_{\alpha}}(w) := \tau_{\beta_{\alpha}}^*(w) + d_{\beta_{\alpha},\ell}(w)$, for $\alpha = 1, \ldots, \sigma - 1$, $\tau_{\beta_{\alpha}}(w) := \tau_{\beta_{\alpha}}^*(w) + s(w)$ and $\tau_{\beta_{\alpha}}(w) := \tau_{\beta_{\alpha}}(1), w \leq 0$. $S_{p,\alpha}(w)$ is an upper bound the maximum delay wafer if can face in modules $\beta_{\alpha}+1, \ldots, 0$ of channel-$\alpha$.

**Theorem 3: Recursion for Delay in a Multiclass Flow Line:** Consider a flow line under the assumptions of this Section. For wafer $w \geq 1$, the total delay in channel-$\alpha$ obeys
\[
Y_{\alpha}(w) = \min \left\{ \Delta_{\beta_{\alpha}}(w), Y_{\alpha}(w - 1) + \tau_{\beta_{\alpha}+1}(w - 1) - \tau_{\ell}^*(w) \right\}^+
\]
with initial conditions $Y_{\alpha}(0) := 0$, $\tau_{\beta_{\alpha}}^*(0) := \tau_{\beta_{\alpha}}(1)$, and $\tau_{\ell}^*(0) := \tau_{\ell}^*(1)$, otherwise. The terms $a_w := a_w$ and $a_w := X_{w,\beta_{\alpha}}(w)$. The entry times to each channel obey
\[
X_{w,1} = \max \left\{ a_w, X_{w-1,1} + \tau_1^*(w - 1) + d_{w-1,1} \right\},
\]
\[
X_{w,\beta_{\alpha}} = a_w + d_{w,0} + T_{B}(w) + \sum_{\alpha=1}^{\beta_{\alpha}} Y_{\alpha}(w)
\]

<table>
<thead>
<tr>
<th>$w$</th>
<th>$k$</th>
<th>$a_w$</th>
<th>$d_{w,0}$</th>
<th>$d_{w,1}$</th>
<th>$d_{w,2}$</th>
<th>$d_{w,3}$</th>
<th>$d_{w,4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>17</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>54</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>28</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>97</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>41</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>150</td>
<td>0</td>
<td>9</td>
<td>41</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>155</td>
<td>31</td>
<td>9</td>
<td>41</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>176</td>
<td>37</td>
<td>32</td>
<td>41</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>200</td>
<td>63</td>
<td>32</td>
<td>16</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>305</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>

*Example 2: Delay in a Channel:* Consider a flow line with two classes of wafers, four modules and deterministic process times. Let $\tau_1^2 = 36, \tau_2^2 = 18, \tau_3^2 = 9$, and $\tau_4^2 = 50$. Let $\tau_1^3 = 18, \tau_2^3 = 9, \tau_3^3 = 4.5$, and $\tau_4^3 = 25$. The delays each arriving wafer faces in the modules are given in Table II. There $w$ is the wafer index and $k$ is the class. Note that the structure of Theorem 1 holds. If there is delay inside the channel, modules after the first module in which delay occurs have the maximum delay. Theorem 2 gives a succinct recursion for calculating delay as an alternative to the elementary evolution equations.
Setsups can be an essential feature of equipment such as multi-cluster tools or clustered photolithography scanners. Wafer lots are the unit of work in semiconductor wafer fabrication and allow reduced computation. We consider wafer lots in only a single channel flow line for two reasons. First, it is this system that allows the greatest computational reduction. Second, such a model has been quite accurate in practical tests.

A. State-Dependent Setups

To describe setups we first consider an independent cluster tool. When changing from one class of wafer to another, the process modules typically adjust their settings for temperature and chemical characteristics. Often, the entire cluster must be vacant for this process to begin. This isolates the wafers from fluctuating tool settings. Note that there is also a class of setups, referred to as rolling setups. These allow a process module setup to be conducted just in advance of the new wafers, without requiring all modules to be empty. An example of tools capable of rolling setups are linear cluster tools described in [20] and [21]. For multicluster tools or clustered photolithography tools, setups may be conducted on a section by section basis.

We model this phenomenon as follows. There is a distinguished module index \( P(w) \) that restricts the start time of the setup. If a setup is required before wafer \( w \) can begin, it may not commence until all preceding wafers have vacated module \( m_{P(w)} \). Let \( \tau_S(w) \) denote the duration of setup for wafer \( w \). Consider two policies for initiating the setup. First, the setup must wait until both wafer \( w - 1 \) has vacated module \( m_{P(w)} \) and wafer \( w \) has arrived. In the second case, allow the setup to begin immediately upon vacation of module \( m_{P(w)} \). Let \( \tau_{V-1}(k) \) denote the instant that wafer \( w - 1 \) vacates module \( m_{P(k)} \). Let \( X_{w-1}^{*} \) denote the adjusted entry time of wafer \( w \) to module \( m_1 \). We thus have:

- Policy 1: \( X_{w-1}^{*} = \max\{a_w, V_{w-1,P(k)} + \tau_S(w)\} \).
- Policy 2: \( X_{w-1}^{*} = \max\{a_w, V_{w-1,P(k)} + \tau_S(k)\} \).

If no setup is required for wafer \( w \), simply set \( P(w) = 1 \) and \( \tau_S(w) = 0 \). Then, the above adjusted \( X_{w-1}^{*} \) can replace \( X_{w-1} \).

Corollary 1: Theorems 2 and 3 hold with \( X_{w-1}^{*} \) replaced by \( X_{w-1}^{*} \).

\[ \alpha P(w) \] denote the dominating module prior to \( m_{P(w)} \), so that \( \beta(\alpha P(w)) \leq P(w) < \beta(1 + \alpha P(w)) \). The vacation time \( V_{w-1,P(w)} \) can be calculated as

\[ V_{w-1,P(w)} = X_{w-1,P(w)} + \sum_{\forall \tau \in \alpha P(w)} Y_{\tau}(w - 1) \]
Use \( L \) as the index of the lots and, abusing notation slightly, let \( \alpha_L \) denote the arrival time of lot \( L \). Let \( W(L) \) denote the number of wafers in lot \( L \). We map the lot index to the wafer index with the function \( \Omega(L, \omega) \). That is, \( \Omega(L, \omega) \) gives the wafer index of the \( \omega \)th wafer of lot \( L \). For example, if lots 1, 2, 3, and 4 each contained 25 wafers, the ninth wafer of lot 5 has wafer index \( \Omega(5,9) = 109 \). Assume that each lot has a single class of wafers and, again abusing notation slightly, let \( c(L) \) denote that class. Thus, \( c(L) = c(\Omega(L, \omega)) \), for all \( L \) and \( 1 \leq \omega \leq W(L) \).

For all wafers except the first wafer of each lot, we assume that \( \tau_B(w) = \tau_B^{(w)} \). That is, \( s(\Omega(L, \omega)) = 0 \), for all \( L \) and \( 2 \leq \omega \leq W(L) \). The next results follow immediately.

**Corollary 2: Wafer Lots in a Single Channel Flow Line:** Consider a single channel flow line under the assumptions of Section III and this subsection. For wafers \( w = \Omega(L, \omega) \) with \( \omega = 2, \ldots, W(L) - 1 \), the total delay obeys

\[
Y(w + 1) = \min \left\{ c_B^{(w)}(w) - \tau_B^{(w)}(w + 1), B \right\}
\]

where \( c(w) = c(1) \), for all \( w \leq 0 \). The recursion also holds for \( w = \Omega(L, 1) \) if we use \( \tau_B^{(w)}(w) \) instead of \( \tau_B^{(w)}(w) \).

**Corollary 3: Queue Time to Enter the Flow Line:** Consider a flow line under the assumptions of Section III and this subsection. For wafers \( w = \Omega(L, \omega) \) with \( \omega = 2, \ldots, W(L) - 1 \),

\[
d_{w,0} = \tau_1^{(w-1)} + \tau_{-1,0} + \tau_{w-1,1}.
\]

Wafer lots allow for significantly reduced computation.

**V. COMPUTATIONAL COMPLEXITY**

We now show that for a single channel flow line with wafer lots, the computation required for the approach of Corollary 2 can be about one order of magnitude less than is possible via the elementary evolution equations. We compare the computation required using three different methods of simulation.

- **Full simulation (FS)** uses the elementary evolution equations for all wafers in modules \( m_1, \ldots, m_B \).
- **Theorem 3 (TH3)** is used to calculate \( Y^{(\alpha)}(w) \) for all \( \alpha = 1, \ldots, \sigma - 1 \), for each wafer.
- **Corollary 2 (C2)** is used to calculate \( Y(w) \) for \( \forall \) wafers. We do not worry about the modules \( m_{B+1}, \ldots, m_M \), since there is no contention after module \( m_B \) by Lemma 1. The exit time of a wafer from the system can be obtained as \( X_w = \tau_B(w) + \tau_B^{(w)}(m_B + 1) \). We consider that addition and substraction are computationally equivalent and denote them as “ADD.” Similarly, minimization, maximization and \( \{c^+, \}^+ \) are denoted as “MAX.” Multiplication is denoted as “MULT.” We do not include calculations for setups. Nor do we consider calculations to obtain offsets to index values as \( c(w + \beta(\alpha) = \beta(\alpha + 1)) \); such values can be calculated initially, stored and determined without computation during the recursions.

**Theorem 4:** **Computation Required:** For systems with \( \sigma \geq 2 \), the number of computations required to simulate \( G \) lots of \( W \) wafers each is given in Tables IV and V. Note that there are \( K \) classes of wafers.

Note that in the case of \( \sigma = 1 \), the bottleneck is the first module. By Lemmas 1 and 2, there is no need to simulate after this module; we can use a postprocessing delay.

We compare the total computation required for FS and C2. The total recursion computations for FS are \( 2GWB - (B + 1) \); for FScomp. The total recursion computations for C2 are \( 11GW + 13G \); for C2comp. We have

\[
\lim_{G \to \infty} \frac{FS_{\text{comp}}}{C2_{\text{comp}}} = \frac{2BW}{11W + 13}.
\]

Since the number of wafers per lot in most fabs averages around 24, and this is anticipated to decrease to perhaps as low as 12 wafers per lot, the ratio for a large number of lots is about \((1/6)B\). For a practical clustered photolithography tool with a 24 wafer pre-scan buffer, \( B \approx 40 \), so that \( \lim_{G \to \infty} FS_{\text{comp}}/C2_{\text{comp}} \approx 6.67 \). C2 is almost one order of magnitude more efficient than FS.

We conclude our discussion on complexity with an example.

**Example 4:** **Computational Complexity:** We consider a flow line model for a practical clustered photolithography tool. There are 12 modules before the photolithography scanner (module \( m_{10} \) is the penultimate dominating module, \( \beta(\sigma - 1) = 10 \)). There is a 24 wafer buffer between the pre-scan modules and the scanner; they are modeled as 24 modules with zero process times. There are two locations in the scanner for wafers, including the bottlenose scan operation; this is module \( m_{38} \), so \( B = 38 \). There are 12 modules after the bottleneck. Consider \( K = 30 \) classes of wafers. Suppose that there are 4 dominating modules, so that \( \sigma = 4 \). From Theorem 4, the computation required to simulate the processing of 3000 lots each consisting of 12 wafers is given in Tables VI and VII. We consider the sum of the ADD and MAX computations for comparison. As is evident from the table, TH3 requires about 87% of the computation of FS. C2 requires less than 1/6 the computation of FS.
VI. MODEL FIDELITY

Here, we discuss the quality of the resulting models and develop approximate adjustment terms accounting for our restrictive assumptions.

A. Discussion of Model Features

It is important to assess the quality of any model for its intended application. The flow line model under consideration has limitations; we discuss their significance.

A first modeling simplification to reduce complexity is to ignore the wafer transport robots. Such a simplification is necessary for fab-level simulation. The internal workings of a single tool in a model consisting of 1000 tools represents unnecessarily computational and excessively detailed. The goal is to control the wafer transport robots. Assuming the system is not robot-bound, wafer transport time that cannot be conducted in parallel with the process should be added to the process time.

Except as discussed, the flow line model employed here is deterministic and without failures. In general, including failures prevents the exact decomposition into channels. Many studies have analyzed failures (often via approximate decomposition and aggregation techniques), see, for example, [3]–[4] and [5]–[8]. The elementary evolution equations must be employed for simulation in that case. Similarly, for random process times as in [22].

As in [9], we have assumed the system is initially empty. It is common practice to start simulations in the empty state. Also, tools frequently return to an empty state during a simulation. Such an assumption is thus acceptable. Note however, that it is useful for wafer transport robot control to be able to operate from any initial condition. Robot control is not our objective.

While Assumptions A1 and A2 are sufficient to enable our results, they are restrictive. Nevertheless, they preserve numerous properties of the system. The bottleneck process times are maintained. The total process time for a wafer can be maintained by adding a constant value to the completion time from the bottleneck; see [18].

To reduce computation, Corollary 2 focuses on a single channel flow line. Such a simplified model can be used to approximate a more complicated one. A key issue is that such a model will not have the same $V_{w=1,P(u)}$ as the original system. An entry time delay for wafers can be introduced to correct this; see the next subsection.

Despite the limitations of the models, they have shown high fidelity in practical application. Single channel flow line models have been tested against data from a clustered photolithography tool (CPT) in production and gave throughout and cycle time values within 1.0% and 4.0%, respectively. We are not at liberty to further discuss the details of the actual implementation. However, in Section VII, we will conduct simulation tests to compare flow line models that do not satisfy Assumptions A1 and A2 to a single channel flow line satisfying both assumptions.

B. Adjustments to Account for Model Assumptions

Here, we develop adjustments to address the shortcomings of our model assumptions and with an eye toward using a single channel as an approximation for a more general flow line. We consider two systems with the same number of modules. Each has wafer arrival times $a_{w}$, wafer classes $c_{w}$, setup durations $\tau_{S}(w)$ and setup modules $P_{w}$ (they are the same in both systems). The first is a flow line model as in Section II that we refer to as DFL (deterministic flow line); it need not satisfy Assumptions A1 and A2. The second satisfies both Assumptions A1 and A2 and has only a single channel; we refer to it as DFL-1-A (“1” for one channel and “A” for assumptions). Since a setup for wafer $w$ can only begin after time $V_{w=1,P_{w}}$, this vacation time can have significant influence on system performance. Let $\tau_{j}$ and $n_{j}$ denote the process times at module $m_{j}$ in the DFL and DFL-1-A systems, respectively.

Note that our application system typically has a single module which serves as the bottleneck (for all classes). We consider that there are $K$ wafer classes and the bottleneck module is the same in both systems. Further, $\tau_{DFL}^{K} = n_{DFL}^{K}$, for all classes; we will construct DFL-1-A models as approximations for DFL systems in Section VII.

To distinguish variables within each system, we use superscripts. Use DFL for the DFL system and GEO for the DFL-1-A system (GEO for geometric decay). Suppose that we have been able to ensure that both systems behave the same for wafers prior to wafer $w$, that is, assume that $Y_{DFL}(w-1) = Y_{GEO}(w-1)$ and $X_{DFL}^{w-1} = X_{GEO}^{w-1}$. Since the entry time of wafer $w$ in each system is $X_{w=1}^{DFL} = \max\{a_{w}, V_{w=1,P_{w}}\} + \tau_{S}(w)$ and $X_{GEO}^{w-1} = \max\{a_{w}, Y_{GEO}(w-1)\} + \tau_{S}(w)$, we study the vacation times. We immediately have

$$V_{DFL}^{w=1,P_{w}} = X_{DFL}^{w=1} + \sum_{j=1}^{P_{w}} \tau_{j}^{DFL} + \sum_{j=1}^{P_{w}} \tau_{j}^{DFL}$$

and similarly for $Y_{GEO}^{w=1,P_{w}}$. Since cumulative delay and start times are the same for wafer $w$ and $d_{DFL}^{w=1} + c_{DFL}^{w=1} + \cdots + d_{DFL}^{w=1} + P_{w} = Y_{DFL}(w-1) = S_{DFL}^{w+1}(w-1)$, we obtain

$$V_{DFL}^{w=1,P_{w}} = V_{GEO}^{w=1,P_{w}} + \sum_{j=1}^{P_{w}} \left[\tau_{j}^{GEO}(w-1) - \tau_{j}^{DFL}(w-1)\right]$$

$$+ \left\{Y_{GEO}(w-1) - S_{DFL}^{P_{w}+1}(w-1)\right\}^+$$

$$- \left\{Y_{GEO}(w-1) - S_{DFL}^{P_{w}+1}(w-1)\right\}^+.$$
Now suppose that the distinguished setup module is fixed, so that either \( P(w) = 1 \) or \( P(w) = P > 1 \). For \( P(w) = P \) and if \( S^{B,DFL}_{P+1}(w-1) \leq S^{B,GE0}_{P+1}(w-1) \), then
\[
\left\{ Y^{GEO}(w-1) - S^{B,DFL}_{P+1}(w-1) \right\}^+ 
- \left\{ Y^{GEO}(w-1) - S^{B,GE0}_{P+1}(w-1) \right\}^+ 
= \min \left\{ S^{B,DFL}_{P+1}(w-1) - S^{B,DFL}_{P+1}(w-1), \right. \\
\left. Y^{GEO}(w-1) - S^{B,DFL}_{P+1}(w-1) \right\}^+.
\]
The term \( \sum_{j=P+1}^{B-1} \left[ \tau^{e(u-1)}_j - \nu^{e(u-1)}_j \right] \). Assuming that the DFL system has zero process times for modules \( m_{P+1}, \ldots, m_{B-1} \) (that is, these are buffer modules), and since the \( \nu^{e(u-1)}_j \) terms are very small (for typical systems with \( P \approx 10 \) and \( \eta \approx 0.75 \)), the latter sum is close to zero. So that
\[
\left\{ Y^{GEO}(w-1) - S^{B,DFL}_{P+1}(w-1) \right\}^+ 
- \left\{ Y^{GEO}(w-1) - S^{B,GE0}_{P+1}(w-1) \right\}^+ \approx 0.
\]
Similarly, for the case \( P(w) = P \) and \( S^{B,DFL}_{P+1}(w-1) > S^{B,GE0}_{P+1}(w-1) \). Thus
\[
A_{w-1,P} := V^{DFL}_{w-1,P} - V^{GEO}_{w-1,P} \approx \sum_{j=1}^{P} \left[ \tau^{e(u-1)}_j - \nu^{e(u-1)}_j \right]
\]
when we have a setup with \( P(w) = P \).

For \( P(w) = 1 \) and if \( S^{B,DFL}_{P+1}(w-1) \leq S^{B,GE0}_{P+1}(w-1) \), we obtain
\[
A_{w-1,1} := V^{DFL}_{w-1,1} - V^{GEO}_{w-1,1} 
= \tau^{e(u-1)}_1 - \nu^{e(u-1)}_1 
+ \min \left\{ \sum_{j=2}^{B-1} \left[ \tau^{e(u-1)}_j - \nu^{e(u-1)}_j \right], \right. \\
\left. \left\{ Y^{GEO}(w-1) - \sum_{j=2}^{B-1} \tau^{e(u-1)}_j \right\}^+ \right. \\
\left. - \sum_{j=w-B+1}^{w-2} \nu^{B}(j) \right\} \right. \\
\left. + \sum_{j=w-B+1}^{w-1} \tau^{B}(j) \right\}^+ \right. \\
\left. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \right. \右
single-channel. DFL-System 6 is intended to test the importance of our assumptions that lead to the “no contention after the bottleneck” results. Here, by setting $\tau_{ij}^c = \tau_{Bi}^c$, for $j > B$, we expect contention after the bottleneck.

We consider four setup scenarios. For the first wafer of every lot, set $\tau_{Bi}(w) = \tau_B(w) + s(w)$, where $s(w) \geq 0$. When changing lot class, the first wafer of the new lot experiences a setup with $P(w) = P$ and $\tau_S(w)$. For such wafers, we set $s(w)$, $P(w)$ and $\tau_S(w)$ based on the following four cases:

1. $P(w) = 9$, $\tau_S(w) \sim U[120, 300]$, $s(w) \sim U[240, 420]$;
2. $P(w) = 1$, $\tau_S(w) = 0$, $s(w) \sim U[240, 420]$;
3. $P(w) = 9$, $\tau_S(w) \sim U[120, 300]$, $s(w) = 0$;
4. $P(w) = 1$, $\tau_S(w) = 0$, $s(w) = 0$.

All other wafers have $P(w) = 1$, $\tau_S(w) = 0$ and $s(w) = 0$. The notation $X \sim U[a, b]$ means that the random variable $X$ is uniformly distributed over the range $[a, b]$.

B. DFL-1-A Systems

To evaluate the quality of a model that satisfies Assumptions A1 and A2, we must construct such models. The following approach first appeared in conference form in [18]. Note again our assumption that all classes have the same bottleneck module in the DFL model; we could however still use the same approach described here, otherwise. Recall that we use $\nu_k^f$ for the process times in a DFL-1-A system. Suppose that the wafer classes are labeled in order of decreasing $\tau_k^B$. Thus, $\tau_k^B \geq \tau_{k+1}^B$, $\forall k \geq 1$.

For a given DFL 1-A system, set:

- $\nu_k^f = \tau_k^B$, for $k$;
- $\nu_k^f = \max\{k, j < B\}$ $\tau_j^B$;
- $\eta = \tau_B^f/\tau_B^B$ and $\eta = \tau_{j+1}^B/\tau_j^B$, $\forall i = 1, \ldots, K - 1$;
- $\mu_i^f = \nu_i^f \left(\eta^{j-1} \prod_{j=1}^{i-1} \eta \right)$, $\forall i < j$;
- $\nu_k^F_{POST} = \sum_{j=k+1}^{M} \tau_j^B - \sum_{j=k+1}^{M} \tau_j^F$, $\forall k$. Here, $\nu_k^F_{POST}$ will be used as the sum of the DFL-1-A post bottleneck process times. It accounts for the fact that the DFL-1-A system will, for all practical examples, have smaller total process time than the DFL system from which it is derived. Also, since our DFL-1-A system will be assumed to have no contention after the bottleneck, we do not need to evolve the wafer advancement. We can simply add $\nu_k^F_{POST}$ to the end of the exit time from the bottleneck. An example is given in Appendix II to illustrate the resulting process times. Note that Assumptions A1 and A2 hold. Hereafter, we refer to the DFL-1-A model derived for DFL-System 1 as DFL-1-A System 1. Similarly for the others.

C. Simulation Results

For each system and setup case, we first simulate with a very fast lot arrival rate to determine the maximum system throughput $\lambda_{MAX}$. We call this the Just-In-Time (JIT) case. For the subsequent simulations, we adjust the rate $\lambda$ of the Poisson arrival process to achieve loading values of $\rho = 0.9, 0.8, \ldots, 0.1$; that is, set $\lambda = \rho \lambda_{MAX}$. Denote the completion time of lot $L$ from the tool as $T_C(L)$ and the start time of lot $L$ to the first module as $T_S(L)$. For the DFL systems, let $CT^{DFL}(L) = T_C^{DFL}(L) - T_S^{DFL}(L)$ be the cycle time of a lot and $TT^{DFL}(L) = \min\{CT^{DFL}(L) - T_C^{DFL}(L), TT^{DFL}(L) = T_C^{DFL}(L) - T_C^{DFL}(L - 1)\}$ be the so-called throughput time. Similarly, for the DFL-1-A (GEO) systems. We compare average $CT^{DFL}$ and $TT^{DFL}$ values for the true DFL system against the results from the DFL-1-A models.

The results for System 2 are depicted in Fig. 3. The circles and diamonds are the average values for $CT^{DFL}$ and $TT^{DFL}$, respectively, over all replications run for that system (each consisting of 15,000 lots data). The dotted and solid lines are the average $CT^{GEO}$ and $TT^{DFL}$, respectively. They are virtually indistinguishable.

The results for System 3, 4, 5 and 6 are very similar to Fig. 3. They are omitted for brevity. Fig. 4 depicts the simulation results for System 6; it shows that the DFL-1-A model can give predictions that are not as good. Similarly, for System 4. These errors arise because our DFL-1-A models have no post-bottleneck contention, however, the DFL Systems 4 and 6 will.

The summary results for the systems are depicted in Table VIII. To describe the results, consider System 1 and Fig. 3. Let $e_C$ be the average of absolute error between the DFL CT and GEO CT values for all 40 cases depicted in Fig. 3; its value is $e_C = 0.38\%$. Similarly, let $e_T$ denote the absolute error average between DFL TT and GEO TT; its value is $e_T = 0.43\%$. In System 1 simulations, a single simulation replication for the DFL system took on average 98.4 CPU seconds, while for the GEO it required 17.1 CPU seconds. Let $\alpha_{CPU}$ denote this ratio; its value is $\alpha_{CPU} = 5.77$. Thus, the FS approach took 5.77 times as long as the GEO approach with Corollary 2.

When using a DFL-1-A (GEO) system as a model for a more general DFL system, the simulations indicate acceptable performance. For models that have no post-bottleneck contention, the predictions are quite good. The presence of post-bottleneck contention appears to degrade the prediction to about 3-5% error. Systems 1, 2, 3, and 5 give about the expected computational reduction, accounting for the fact that we must compute the correction terms of Section VI-B. Note that System 2 has 12 fewer modules so less computation for FS. Systems 4 and 5 give computational reduction as expected, noting that we must simulate the post-bottleneck modules (where we lump them together as a single additive post-bottleneck process time in all DFL-1-A models and the other DFL systems). Thus, the proposed models well predict cycle time and throughput for practically structured flow lines. Reduction in CPU time is about as predicted in Section VI.

VIII. CONCLUDING REMARKS

For deterministic multiclass flow lines, we identified a class of process times allowing for the decomposition of the system into channels. A recursion for the wafer delay was then developed. By considering batch arrivals and single channel systems, an order of magnitude reduction in the computation required to simulate the system was possible.

In semiconductor wafer fabrication, tools such as the clustered photolithography tool possess internal wafer buffers and state dependent setups. Tool models in existing simulation software may not be sufficient for a high mix, small lot manufacturing environment. Here we developed multiclass flow line models to address these shortcomings. In tests against production data, the models gave throughput and cycle time values...
within 1\% and 4\% of actual, respectively. Our approach can require about one order of magnitude less computation than the elementary evolution equations. The models are thus good candidates for use in fab-level simulations to improve the fidelity of existing equipment models and reduce computation.

There are numerous directions for future work. To reduce the computation further, it is possible to develop a simplified model, with only a few additional parameters beyond the typical ones, that captures the behavior? One possible parameter could be stored work, akin to our in-system delay $Y(w)$. Another direction is to attempt to employ the structural properties we have exposed in conjunction with hybrid simulation models or with decomposition/aggregation methods. Finally, can robot limited systems be modeled via these approaches?

APPENDIX I

PROOFS

Here, we prove the results stated in this paper. First, two easy but essential lemmas are given.

Lemma 3: Consider a one channel flow line under Assumptions A1 and A2 as in Theorem 1. The process times satisfy $\tau_{j+1}^k \leq \tau_j^k$, for all $j < B - 1$ and all classes $k, k'$.

Proof of Lemma 3: By Assumption A2, $\tau_{j+1}^k \leq \eta_j^k$. If $k' < k$, then Assumption A1 gives $\tau_j^k = \tau_j^{k'} + \eta_j^{k'} + \cdots + \eta_{k-1}$. Hence, $\tau_j^k = \tau_j^{k'} + \eta_j^{k'} + \cdots + \eta_{k-1} < \tau_j^{k'}$, since all $\eta_j \eta_j^k < 1$. Similarly, for $k > k'$. The result is immediate for $k = k'$.

Lemma 4: Under Assumptions A1 and A2, $\tau_B^k \geq \eta_B^k$, for all classes $k \neq k'$.

Proof of Lemma 4: If $k > k'$, $\tau_B^k = \tau_B^{k'} + \eta_B^{k'} + \cdots + \eta_{k-1} \geq \eta_B^{k'}$. If $k < k'$, $\tau_B^k = \tau_B^{k'} + \eta_B^{k'} + \cdots + \eta_{k-1} \geq \eta_B^{k'}$.

Proof of Lemmas 1 and 2: No Contention After the Bottleneck: The results can be proven simultaneously by nested induction on the wafer index and the module index. However, by Lemma 3, it is intuitively clear that no contention is possible. The result follows immediately in that case.

Proof of Theorems 1 and 2: Delay in a Single Channel Multiclass Flow Line: First note that (3) follows simply by the definition of $d_{w,j}$ since $X_{w} = \max\{a_w, X_{w-1,2}\} = \max\{a_w, X_{w-1,1} + \tau_1^{c(w)} + d_{w-1,1}\}$.

We prove results (1) and (2) jointly; Theorem 1 will follow from (2). In (1), it is more convenient to use $X_{w-1,1} = a_w + d_{w,0} + \tau_1^{c(w)} + d_{w,1}$.

Fig. 3. Simulation results for System 1.
For the initialization step we must show that (1) and (2) hold
for all \( j = 1, \ldots, B \), and \( Y(0) = 0 \). The elementary evolution equations immediately give 
\( d_{1,j} = 0 \), for all \( j = 1, \ldots, B - 1 \); hence, \( Y(1) = 0 \). This is exactly what the recursion (1) gives, since \( \min \{ S_0^B(1), -\infty \}^+ = 0 \). Also, \( \min \{ \tau_B(1) - \tau_j(1), 0 - S_0^B(w) \}^+ = 0 \). This shows the initial \( w = 1 \) case.

Next, we seek to show that if (1) and (2) hold for \( w > 1 \), then they also hold for \( w + 1 \).

We start with the case \( Y(w) = 0 \) and consider \( a_{w+1} \leq a_w + d_{w,0} + \tau_1(w) + d_{w,1} \) (Case C1). Since we assume \( Y(w) = 0 \), the condition becomes \( a_{w+1} \leq a_w + d_{w,0} + \tau_1(w) \). Since \( Y(w) = 0 \), we immediately obtain \( X_{w,j} = a_w + d_{w,0} + \tau_1(w) \), for \( j = 1, \ldots, B \), where \( \tau_1(w) := 0 \). Also, \( C_{w,B} = a_w + d_{w,0} + \tau_1(w) \). From the elementary evolution equations, Lemma 3 and \( a_{w+1} \leq a_w + d_{w,0} + \tau_1(w) \), we obtain \( X_{w+1,j} = a_{w+1} + d_{w,0} + \tau_1(w) \), for \( j = 1, \ldots, B - 1 \). Similarly, \( X_{w+1,B} = \max \{ a_{w+1} + d_{w,0} + \tau_1(w) + \tau_B^{-1}(w + 1), a_w + d_{w,0} + \tau_B^{-1}(w) \} \). From this we obtain 
\( d_{w+1,j} = 0 \), for \( j = 1, \ldots, B - 2 \) and \( d_{w+1,B-1} = \max \{ a_{w+1}, \tau_B(w) - \tau_1(w) + \tau_B^{-1}(w + 1) \} \). Summing these terms gives \( Y(w + 1) \). This shows both (1) and (2).

There is no case C2 with \( Y(w) = 0 \), since then \( S_0^B(w + 1) > Y(w) + \tau_B(w) - \tau_1(w) \) using Lemma 4.

---

**TABLE VIII**

<table>
<thead>
<tr>
<th></th>
<th>( e_C )</th>
<th>( e_T )</th>
<th>( \alpha_{C,PU} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS1</td>
<td>0.38%</td>
<td>0.43%</td>
<td>5.77</td>
</tr>
<tr>
<td>SYS2</td>
<td>0.21%</td>
<td>0.47%</td>
<td>3.77</td>
</tr>
<tr>
<td>SYS3</td>
<td>0.80%</td>
<td>0.40%</td>
<td>5.65</td>
</tr>
<tr>
<td>SYS4</td>
<td>5.35%</td>
<td>3.13%</td>
<td>7.40</td>
</tr>
<tr>
<td>SYS5</td>
<td>2.67%</td>
<td>0.16%</td>
<td>5.87</td>
</tr>
<tr>
<td>SYS6</td>
<td>5.06%</td>
<td>2.54%</td>
<td>7.75</td>
</tr>
</tbody>
</table>

---

*Fig. 4. Simulation results for System 6.*
For case C3 with $Y(0)$, we have $a_{w+1} > a_w + d_{w;B} + \tau_1^{(w)}$. Using this assumption, the elementary evolution equations and Lemma 3, we obtain $X_{w+1,j} = a_{w+1} + T_F^{-1}(w+1)$, for $j = 1, \ldots, B-1$ and $X_{w+1,B} = \max\{a_{w+1} + T_F^{-1}(w+1), a_w + d_{w;0} + T_F^{-1}(w)\}$. The delays immediately follow as $d_{w,j+1} = 0$, for $j = 1, \ldots, B-2$ and $d_{w,0,B-1} = \max\{0, \tau_B^{(w)} - a_{w+1}, a_w + d_{w,0} + T_F^{-1}(w) - T_F^{-1}(w+1)\}$. This suffices to show (1) and (2) for this case.

The cases for $Y(0) > 0$ are more involved. We first show that $S_B^{(w)}(w)$ is strictly increasing in $j$. By definition, $S_B^{(w)}(w) = \sum_{j=0}^{B-1} \tau_B^{(w-B+j)} - \tau_1^{(w)}$. Note that $\tau_B^{(w)} \geq \tau_1^{(w)}$ by Lemma 4 and the fact that $\tau_1^{(w)} = \tau_1^{(w)}$. Thus, each term is strictly positive and $S_B^{(w)}(w)$ is strictly increasing. Since we have assumed our induction hypothesis that (2) holds, this allows us to conclude Theorem 1 for wafer $w$. Once we show (2) for wafer $w + 1$, Theorem 1 will hold for that wafer as well (and hence all wafers).

We now turn to case C1 with $Y(0) > 0$. Using Theorem 1 for wafer $w$ as just shown, we immediately have:

- $X_{w,j} = X_{w,j-1} + T_F^{-1}(w) = T_F^{-1}(w) = T_F^{-1}(w)$ for $j = 1, \ldots, j^*$;
- $X_{w,j+1} = X_{w,j} + T_F^{-1}(w) + d_{w,j+1};$
- $X_{w,j} = X_{w,j+1} + T_F^{-1}(w) + d_{w,j+1} + \sum_{j^*+1}^{j+1} \tau_B^{(w+j-B)}$ for all $j = j^* + 1, \ldots, B-2$.

Using the C1 assumption on $a_{w+1}$ and Lemma 3, we obtain:

- $X_{w,j} = X_{w,j-1} + \tau_1^{(w)} + \tau_1^{(w)} = T_F^{-1}(w) + \tau_1^{(w)}$ for all $j = 1, \ldots, j^*$;
- $X_{w,j} = X_{w,j-1} + \tau_1^{(w)} + \tau_1^{(w)} = T_F^{-1}(w) + \tau_1^{(w)}$ for all $j = j^* + 1, \ldots, B-1$.

From these, we obtain the delays as:

- $d_{w,j+1} = 0$ for all $j = 1, \ldots, j^*$;
- $d_{w,j+1} = \{d_{w,j+1} - d_{w,j+1} - d_{w,j} - T_F^{-1}(w) + \tau_1^{(w)}\}$;
- $d_{w,j+1} = \{d_{w,j+1} - d_{w,j+1} - d_{w,j} - T_F^{-1}(w) + \tau_1^{(w)}\}$ for all $j = j^* + 1, \ldots, B-1$.

By adding these terms together and telescoping we find $Y(w + 1) = \{Y(w) + T_F^{-1}(w) + \tau_1^{(w)} + T_F^{-1}(w) - T_F^{-1}(w+1)\}$. Using $Y(w)$ as the sum of $d_{w,j}$, we can simplify as $Y(w + 1) = \{Y(w) + T_F^{-1}(w) - \tau_1^{(w)} + T_F^{-1}(w) - T_F^{-1}(w+1)\}$. This concludes the proof of (1) for case C1.

To show (2) for case C1 with $Y(0) > 0$, we can use the $d_{w,j+1}$ forms just obtained. Consider $j > j^*$, so that $d_{w,j+1} = \{d_{w,j} + d_{w,j} - T_F^{-1}(w) + \tau_1^{(w)}\}$ for all $j = j^* + 1, \ldots, B-1$. There are three cases to consider. Note that $T_F^{-1}(w) = 1 + B + j > \tau_1^{(w)}$.

- If $Y(w + 1) = \{Y(w) + T_F^{-1}(w) + \tau_1^{(w)}\}$, then both $\{\cdot\}$ terms in $d_{w,j+1}$ are greater than or equal to zero. So, $d_{w,j+1} = T_F^{-1}(w) + \tau_1^{(w)}$.
- If $0 < Y(w) < \tau_1^{(w)}$, then only the left-hand $\tau_1^{(w)}$ term in $d_{w,j+1}$ is positive so that $d_{w,j+1} = Y(w) + \tau_1^{(w)}$.
- If $Y(w + 1) > 0$, then both terms in $d_{w,j+1}$ are zero. Thus, (2) holds. Since $S_B^{(w)}(w+1)$ is strictly increasing in $j$, Theorem 1 also holds for wafer $w+1$. This concludes the proof for case C1.

The proofs for cases C2 and C3 are similar. Readers may consult the proofs in [12] for some ideas useful for C3.

**Proof of Theorem 3: Recursion for Delay in a Multiclass Flow Line:** Consider channel $\alpha$ in a multiclass flow line under Assumptions A1 and A2. Consider the arrival time to the channel as $X_{w,j}(\alpha)$ and the service time of module $m_{j}(\alpha+1)$ as $\tau^{(\alpha)} + d_{w,j}(\alpha+1) + m_{j}(\alpha+1)$. With these definitions, each channel obeys Theorem 2. Concatenating the equations for each channel gives the result.

**Corollaries 1 and 2 follow immediately.**

**Proof of Theorem 4: Computation Required:** We consider addition and subtraction to be the same and denote them as ADD. Similarly, maximization, minimization, and $\{\cdot\}$ are denoted as MAX. MULT is used for multiplication.

For FS, we will advance GW wafers through B modules. No initializations are required. The first wafer never faces contention so we can simply add the process times to calculate $X_{i,j}$ using $B-1$ additions. For each subsequent wafer, $X_{i,j}$ requires $2$ MAX. $X_{w,j}$ requires $1 MAX$ and $1 ADD$ for $j = 2, \ldots, B-1$, and $X_{w,0,B-1}$ requires $1 MAX$ and $2 ADD$. The first lot requires $(B-1) + (B-1)B$ ADD and $0 + (B-1)B$ MAX. For the remaining $G-1$ lots, we require a total of $(G-1)B$ ADD and MAX. This completes FS.

For TH3, we conduct five initialization operations: $1, \ldots, 15$. In 11, our goal is to calculate $S_{\beta}^{(\alpha)}(\alpha)$ for $j = 1, \ldots, 15$. For $j = 1, \ldots, 15$, we must add the term $a_{j}(B-\beta-1)$; this takes one ADD and one MULT. I1 thus requires $2B-\sigma$ ADD and 1 MULT. For 12, we recursively calculate $2 \sum a_{j}^{(\alpha)}$ for all $K$ classes and modules $j = 1, \ldots, B$, using $K(B-2)$ ADD.

For I3, we calculate $\tau_{\alpha}^{(\alpha)\alpha}$ for each class and $\tau_{\alpha}^{(\alpha)\alpha}$ for each class in $K(\sigma-1)$ ADD. For $i4$, calculate $S_{\alpha}^{(\alpha)}(\alpha)$ minus the result of I3 for each $\alpha$ in $\sigma-2$ ADD. For $i5$, obtain $\sum_{\alpha=1}^{\beta} (\alpha) - \sum_{\alpha=1}^{\beta} (\alpha)$ for all $k \neq k'$ and all $\alpha$ in $K(\sigma-1)$ ADD.

For TH3, we conduct five recursion steps: $R1, \ldots, R5$. For $R1$, assuming we know each input value, we calculate $Y^{(w)}$, for each $\alpha$ in $3(\sigma-1)$ MAX and $3(\sigma-1)$ ADD. The remaining steps are to calculate the input values for $Y^{(w)}$ (or alternately the next recursion of $Y(w+1)$). R2 calculates $S_{\alpha}^{(\alpha)}(\alpha+1)$ recursively from $S_{\alpha}^{(\alpha)}(\alpha)$ for all $\alpha$ in $3(\sigma-1)$ ADD. For use in the R2 recursion, R3 obtains $T_{\alpha}^{(\alpha)}(w+1)$ and $d_{w,j}(\alpha+1)$ with 1 ADD for each $\alpha = 1, \ldots, \sigma-2$ (there is no delay at the end of last channel). This requires that we evaluate $d_{w,j}(\alpha+1)$ which takes 2 MAX and 3 ADD for each of the relevant $\alpha$. R3 requires a total of $3(\sigma-2)$ ADD and $2(\sigma-2)$ MAX. R4 can determine $a_{j}^{(w)}$ in 3 ADD per each of $\alpha = 2, \ldots, \sigma-1$; totaling $3(\sigma-2)$ ADD. Finally, R5 calculates $X_{w,j}$ in 2 ADD and 1 MAX. The total for the recursions is $17\sigma - 20$ ADD and $5\sigma - 6$ MAX.
The approach is similar for C2, but much simpler because there is only a single channel and $d_{w,B} = 0$.

**APPENDIX II**

**EXPERIMENTAL DATA**

Here, we provide details of the systems from Section VII. Process times for DFL-Systems 1 and 2 are given in Table IX. Process times for DFL-System 1 are given in Table XIII. Process times for DFL-Systems 3 and 4 are given in Table X. Process times for DFL-Systems 5 and 6 are given in Table XI and XII.

The buffer modules have...
TABLE XIII
PROCESS TIME DATA FOR DFL-1-A SYSTEM 1

<table>
<thead>
<tr>
<th>Class ((k))</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\nu_1)</td>
<td>52</td>
<td>52(\eta_1) = 48</td>
<td>52(\eta_1)(\eta_2) = 44</td>
</tr>
<tr>
<td>(\nu_2)</td>
<td>52(\eta_1)(\eta_1) = 1</td>
<td>52(\eta_1)(\eta_2)(\eta_1) = 1</td>
<td></td>
</tr>
<tr>
<td>(\nu_3)</td>
<td>52(\eta_1)(\eta_2)(\eta_1) = 1</td>
<td>52(\eta_1)(\eta_2)(\eta_2)(\eta_1) = 1</td>
<td></td>
</tr>
<tr>
<td>(\ldots)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(\nu_{32})</td>
<td>52(\eta_1)(\eta_1)(\eta_2)(\eta_1)(\eta_1) = 2</td>
<td>52(\eta_1)(\eta_2)(\eta_1)(\eta_1)(\eta_1) = 2</td>
<td></td>
</tr>
<tr>
<td>(\nu_{34})</td>
<td>65</td>
<td>60</td>
<td>55</td>
</tr>
<tr>
<td>(\nu_{POST})</td>
<td>454.4</td>
<td>451.3</td>
<td>469.2</td>
</tr>
</tbody>
</table>

\(\nu_1 = 52, \eta = 55/65 = 11/13, \eta_1 = 60/65 = 12/13, \text{ and } \eta_2 = 55/60 = 11/12.\) Sample values follow: \(\nu_1 = 52(\eta) = 44, \nu_2 = 52(\eta)\(\eta_1\) = 40.62, \nu_3 = 52\(\eta_1\)\(\eta_2\)\(\eta_1\) = 37.23, \nu_3 = 52\(\eta_1\)\(\eta_2\)\(\eta_1\)\(\eta_1\) = 34.37, \text{ and } \nu_3 = 52\(\eta_1\)\(\eta_2\)\(\eta_1\)\(\eta_1\)\(\eta_1\) = 31.50.

REFERENCES


James R. Morrison (S’97–M’00) received one B.S. degree in electrical engineering and a second B.S. degree in mathematics from the University of Maryland, College Park, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana–Champaign, Urbana, in 1997 and 2000, respectively.

He was with the Fab Operations Engineering Department at the IBM Corporation from 2000 to 2005. He is currently an Assistant Professor in the Department of Industrial and Systems Engineering, KAIST, South Korea. His research interests include semiconductor wafer fabrication, system design, queueing networks and stochastic control.