Flow Lines with Regular Service Times: Evolution of Delay, State Dependent Failures and Semiconductor Wafer Fabrication

James R. Morrison, *Member, IEEE*

***Abstract*—For the class of flow lines (alternately referred to as tandem queues or assembly lines) possessing deterministic service times, this paper elucidates the evolution of customers within the line. It is shown that a natural decomposition of the servers exists that allows one to express the delay each customer faces without individually assessing the behavior at each server. The decomposition is characterized by successive bottlenecks and leads to a series of resettable monotone channels (RMCs). The delay evolution enables us to characterize state dependent restrictions on when customers may enter service with the flow line. The model is directly applicable to an important class of semiconductor wafer manufacturing tools (serial processing cluster tools). These tools may experience a setup when changing from one recipe to another, and in this case, generally delay the admission of a product lot into the tool until the preceding lot has vacated an initial portion of the cluster and a setup has been conducted.**

I. Introduction

FLOW lines, also termed assembly lines or tandem queues, have been studied for many years and there exists a rich and diverse theory addressing their evaluation and design. They may be used to model assembly line manufacturing (as in the automotive industry) or communication systems where a message is transmitted along successive links. They may also be used as a model for serial processing cluster tools in semiconductor wafer fabrication, and it is from this application that we draw our motivation.

In semiconductor wafer fabrication, cluster tools such as chemical vapor deposition and clustered photolithography tools play a central role. Cluster tools consist of multiple process modules clustered into a single physical (or logical) chassis. Modules are served by one or more wafer transport robots. The class of such tools in which all wafers follow the same path as they proceed sequentially from one process operation to the next (perhaps omitting some process module or operation) is referred to as a serial processing cluster tool. When such a tool must change from conducting operations on one class of wafer to another, a setup may be required (e.g., the module temperature and chemical must change). The setup may require that an initial portion of the tool be vacant of wafers prior to the initiation of the setup.

Ignoring contention for a wafer transport robot, serial processing cluster tools are essentially asynchronous flow lines with regular (deterministic) service times, reliable machines (process modules) and discrete customers (the wafers); see [1]. In [2], it is shown that with a generic arrival process the total delay (non-process time) in the system is the same as that of a G/D/1 queue (with service time equal to that of the bottleneck module) *regardless of the server order*. One consequence is that buffers may be viewed as process modules with zero service time. Extensions for batch arrivals and practical production complications are discussed in [3]-[5]. Analyses of the maximum throughput, order of the servers and buffer locations and sizes for variations of the system under consideration may be found in [6] – [10]. None characterizes the evolution of customers within the line beyond the initial defining relations.

The intent of this paper is to glean a deeper understanding of the procession of customers through the flow line and further, to propose a new state dependent failure mode. It is shown that the evolution of customer delay is dictated by *successive bottlenecks* (servers whose process time is strictly greater than those preceding them), each of which forms what we term a resettable monotone channel (RMC). Rather than tracking the interaction of each customer at each server, we need only attend to the status of the RMCs. If our interest lies solely with one of the RMCs, we show that there is an evolution for it alone. If the one of interest is just prior to the bottleneck, a further simplified recursion for delay evolution may be obtained. The simplified recursions allow us to readily incorporate setups, which may be considered a state dependent failure mode (as setup initiation occurs once the interior of the tool has reached a certain state).

There is a connection with the work of [11], which pursues optimal control policies in a fluid version of the flow line subject to holding costs. They use a successive bottleneck concept in obtaining an optimal control.

Throughout, no proofs are given due to space limitations, however it is worth noting that induction is essential and in some cases the (max,+) algebra ([12]) was employed.

The remainder of the paper is organized as follows. In Section II, the flow line model is introduced and existing fundamental results reviewed. A recursion for the evolution of the delay experienced by customers is obtained in Section III. Section IV focuses on reduced evolution recursions, their application to our particular class of state dependent delay and batch arrivals. Concluding remarks are made in Section V.

Manuscript received February 29, 2008.

James R. Morrison is with the Department of Industrial and Systems Engineering and the KAIST Institute for the Design of Complex Systems, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea (e-mail: james.morrison@kaist.edu; phone: +82 – 42 – 869 - 3127; fax: +82 – 42 – 869 - 3110).

II. Flow Lines with Regular Service Times

A flow line consists of M servers at which customers receive service sequentially. The *service time* of a customer in server i is denoted as τ_i. Assume they are deterministic and customer independent (i.e., there is a single class of customer). There may be buffers between the servers where customers await service; model these as servers with zero service time ([2]). The flow line is subject to manufacturing blocking, in that once a customer has received service, it may proceed only if the subsequent server is empty. No blocking occurs at the final server (customers exit the system). There is a queue of infinite capacity prior to the first server and customer arrivals to the system occur at the arrival times $a_1, a_2, \ldots$ (where $a_i \le a_{i+1}$). For definiteness, assume that all processes (e.g., the service and interarrival times) are right-continuous with left limits. Such a system is depicted in Fig. 1.

Practical systems may have multiple servers dedicated to long duration processes. For example, three servers may be devoted to a process requiring 120 seconds. These servers can produce three parts per 120 seconds, or one part every 40 seconds. This case can be modeled in our framework with a *series* of three servers with 40 second service times. Each part will spend 120 seconds in the servers devoted to this process (as in the original system) and the throughput potential remains three parts per 120 seconds. Further, up to three parts may be in service with the process at any given instant in the original system and in the serial model.

Customer movement through the flow line is dictated by the evolution equations detailed next. Let $x_i(k)$ denote the start time of the service of customer k at server i. We have

$$\begin{aligned} x_1(k+1) &= \max\{a_{k+1}, x_2(k)\}, \\ x_i(k+1) &= \max\{\tau_{i-1} + x_{i-1}(k+1), x_{i+1}(k)\}, \quad 2 \le i \le M-1, \\ x_M(k+1) &= \max\{\tau_{M-1} + x_{M-1}(k+1), \tau_M + x_M(k)\}, \end{aligned} \tag{1}$$

with initial conditions $x_i(0) = -\infty$, for i = 1, …, M.

Let $c_i(k)$ denote the completion time of customer k at server i; that is, $c_i(k) = \tau_i + x_i(k)$. (Time spent waiting for the subsequent server to be available is not included.) Let B denote the index of that server with $\tau_B = \max\{\tau_i, \text{ for all } i\}$, and $B \le j$ for all other servers j with $\tau_j = \tau_B$. This server is termed the *bottleneck server* (it is the first server whose service time is greater than or equal to all others. The next two lemmas may be used to prove Theorem 1, which is a key result of [2].

Lemma 1: No contention for servers after the bottleneck. After the bottleneck server B,

$$\begin{aligned} x_{i+1}(k) &= \tau_i + x_i(k), \quad B \le i \le M, \\ c_M(k) &= \tau_M + x_M(k). \end{aligned} \tag{2}$$

Lemma 2: Customer start times after the bottleneck. After the bottleneck server B,

$$\begin{aligned} x_i(k+1) &\ge \tau_B + x_i(k), \quad B \le i \le M, \\ c_M(k) &\ge \tau_B + c_M(k). \end{aligned} \tag{3}$$

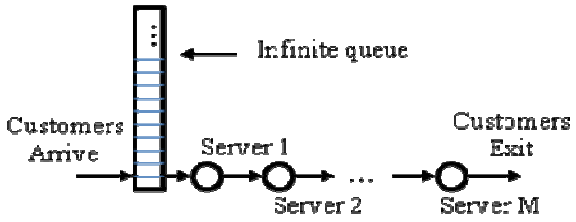

Fig. 1. A flow line. Intermediate buffers are considered as a server with zero service time.

Theorem 1: Customer completion times. Let $\tau = (\tau_1, \tau_2, \ldots, \tau_M)^T$ denote the column vector of the service times and $e = (1, 1, \ldots, 1)^T$ the column vector of 1's. Starting from an empty system (that is, let $c_M(0) = -\infty$),

$$c_M(k+1) = \max\{a_{k+1} + e^T\tau, c_M(k) + \tau_B\}. \tag{4}$$

Simply put, the theorem states that either an arriving customer encounters no contention for a server during its sojourn through the flow line or it departs the system τ_B time units subsequent to its predecessor. An interesting consequence of this is that the order of the servers is not relevant to the completion times. (From this observation, one can infer that buffers may be modeled as servers with zero service time.) The next two corollaries and theorem readily follow, see [2] – [4].

Corollary 1: Completion time for batches of customers. Suppose customers arrive in batches of size W and batch j arrives at time a_j. Let $c_M(j,w)$ denote the completion time of the w-th customer of batch j from module M. For an initially empty tool (that is, let $c_M(0,W) = -\infty$),

$$\begin{aligned} c_M(j+1, w) = \max\{&a_{j+1} + e^T\tau, c_M(j, W) + \tau_B\} \\ &+ (w-1)\tau_B. \end{aligned} \tag{5}$$

Corollary 2: Idle system required. If batch j+1 requires that all previous customers have exited the flow line before entering service,

$$c_M(j+1, w) = \max\{a_{i+1}, c_M(j, W)\} + e^T\tau + (w-1)\tau_B. \tag{6}$$

Theorem 2: Queueing delay with random arrivals.

i) The total time spent in the system by customer k is

$$\sum_{i=1}^{M} \tau_k + d(k), \tag{7}$$

where d(k) is time the same customer would have spent waiting in the queue of a single server flow line with deterministic service time τ_B. In particular, if the interarrival times are exponential with rate λ, then $d(k) = \tau_B\rho/(1-\rho)$, where $\rho := \lambda\tau_B$.

ii) For the case of batch arrivals, the total time spent in the system by customer w of batch j (each consisting of W customers) is

$$\sum_{i=1}^{M} \tau_k + (w-1)\tau_B + d_W(j), \tag{8}$$

where $d_W(j)$ is the delay experienced by a batch arrival to a single server queue with deterministic service time τ_B. For exponential batch interarrival times of rate λ, $d_W(j) = W\tau_B\rho/(1-\rho)$, where $\rho := \lambda\tau_B W$.

Note that the above equality results (as opposed to the inequalities of Lemmas 1 and 2) all discuss completion times from the final server. It is easy to extend them to completion times from the bottleneck.

III. The Evolution of Delay

Internal to a flow line with regular service times, the dynamics of customer movement have not received great attention in the literature. For completion times, the results of Section II are sufficient. However, should the system behavior depend upon the internal state (as in the case of setups in serial processing cluster tools, where a collection of wafers may not begin service until all preceding ones have vacated a certain server) additional insight is required. The results hinge upon the progression of customers progress through RMCs.

Definition 1. A flow line with M servers, deterministic service times $\tau_1, \tau_2, \ldots, \tau_{M-1}$ and deterministic but customer dependent service times $\tau_M(k)$ is termed a *resettable monotone channel* (RMC) if two conditions hold

i) $\tau_M(k) > \tau_1 > \tau_i$, for all i = 2, …, M-1, for all k, and

ii) $\tau_M(k+1) \geq \tau_M(k)$ for all customers k+1 such that

$$f_{k+1} := a_{k+1} + \sum_{i=1}^{M-1} \tau_i - c_M(k) \leq 0.$$

Otherwise, $\tau_M(k+1) \geq \max\{\tau_M(1), \tau_M(k) - f_{k+1}\}$.

Simply put, only the last server is slower than the first and, further, it is progressively slower per customer (this is the monotonicity) until a customer arrives to server M after its predecessor has exited server M. After which, the service time may reset to as fast as the initial $\tau_M(1)$ (the resettable feature). Note that *server M is always the channel bottleneck.*

The following theorem may be proved similarly to those of Section II, with care taken to account for the resettable monotonicity of $\tau_M(k)$.

Theorem 3: Customer completion times in an RMC. Let $\tau(k) = (\tau_1, \tau_2, \ldots, \tau_M(k))^T$ denote the column vector of the service times and recall that $e = (1, 1, \ldots, 1)^T$. Starting from an empty system (that is, let $c_M(0) = -\infty$),

$$c_M(k+1) = \max\{a_{k+1} + e^T\tau(k+1), c_M(k) + \tau_M(k+1)\}. \quad (9)$$

Corollary 3: Total delay in an RMC. Let $Y'(k) := c_M(k) - (a_k + e^T\tau(k))$ denote the delay a customer experiences in an RMC. Then, we have

$$Y'(k+1) = \max\{0, Y'(k) + \tau_M(k) + a_k - a_{k+1}\}. \quad (10)$$

Next we state our first main result characterizing the delay in an RMC; there are two key points. First, for a delayed customer in an RMC, there is an initial server at which a customer experiences delay and for all subsequent servers the delay is dictated by the pace of server M. Second, it is sufficient to know (and there is a recursion to calculate) the total delay in the system because all individual server delays may be obtained from the total.

First, we define the following variables. Let

$$\begin{aligned} d_0(k) &:= x_1(k) - a_k, \\ d_i(k) &:= x_{i+1}(k) - (x_i(k) + \tau_i), \quad 1 \leq i < M, \end{aligned} \quad (11)$$

denote the delay a customer experiences waiting for server 1 and server i+1, respectively. Use Y(k) to denote the total delay once entering the first server of the RMC, that is

$$Y(k) := \sum_{i=1}^{M-1} d_i(k). \quad (12)$$

Finally, let

$$S(k) := \sum_{n=1}^{M-1} [\tau_M(k-M+n) - \tau_n], \quad (13)$$

with $\tau_M(k) = \tau_M(1)$ for $k \leq 0$. S(k) is the maximum possible delay that may be incurred by customer k internal to an RMC.

Theorem 4: Customer delay in an RMC. For an initially empty system (that is, let $c_M(0) = -\infty$), Y(k) obeys the following recursion

$$Y(k) = \max\left\{0, \min\left\{\begin{matrix} S(k), Y(k-1) + \tau_M(k-1) \\ -\max\{\tau_1, a_k - a_{k-1} - d_0(k-1)\} \end{matrix}\right\}\right\}. \quad (14)$$

In addition, the server delays may be found as

$$\begin{aligned} d_0(k) &= \max\{0, a_{k-1} - a_k + \tau_1 + d_0(k-1) + d_1(k-1)\}, \\ d_j(k) &= \max\left\{0, \min\left\{\begin{matrix} \tau_M(k-M+j) - \tau_j, \\ Y(k) - \sum_{n=j+1}^{M-1} [\tau_M(k-M+n) - \tau_j] \end{matrix}\right\}\right\}, \end{aligned} \quad (15)$$

for j = 1, …, M-1 and we set $\tau_M(k) = \tau_M(1)$, $k \leq 0$. The initial conditions are $Y(1) = 0$, $d_j(1) = 0$.

Now, we are poised to deduce the manner in which delay evolves in a flow line with deterministic service times.

Definition 2. Each server i whose service time $\tau_i > \tau_j$, for all $j < i$, is termed a *successive bottleneck.* Suppose there are σ such servers and let $\beta(1), \beta(2), \ldots, \beta(\sigma)$ denote their server indices. (Thus, $\beta(\sigma) = B$).

Example 1: Successive bottlenecks. Consider a flow line consisting of M = 7 servers with $\tau_1 = 45$, $\tau_2 = 15$, $\tau_3 = 30$, $\tau_4 = 60$, $\tau_5 = 45$, $\tau_6 = 45$,and $\tau_7 = 90$. Servers 1, 4 and 7 are the successive bottlenecks and server 7 is the system bottleneck. □

The servers between and including each adjacent pair of successive bottlenecks behave as an RMC. The service time of the final server in such an RMC retains its nominal value until all servers excepting the initial one of the downstream RMC

have filled with customers. At this point, the service time of the RMC's final server begins to follow that of the downstream RMC service time. The service times reset when the system empties sufficiently. Invoking Theorem 4 successively from RMC to RMC, we obtain the subsequent theorem; it is our second main result.

Additional notation is helpful. Define

$$S^{\alpha}(k) := \sum_{j=\beta(\alpha)}^{\beta(\alpha+1)-1} \left[\tau_{\beta(\alpha+1)} + d_{\beta(\alpha+1)}(k+j-\beta(\alpha+1)) - \tau_j\right]. \quad (16)$$

$S^{\alpha}(k)$ is the maximum delay possible for customer k in the RMC between servers $\beta(\alpha)$ and $\beta(\alpha+1)$-1 (which does not include any delay in the final server of the RMC). Let $Y^{\alpha}(k)$ denote the delay incurred by customer k in servers $\beta(\alpha)$, …, $\beta(\alpha+1)$-1, that is

$$Y^{\alpha}(k) := \sum_{j=\beta(\alpha)}^{\beta(\alpha+1)-1} d_j(k). \quad (17)$$

We may now state the theorem. We are only concerned with the evolution prior to the bottleneck, since there is no delay after this point (by Lemma 1).

Theorem 5: Delay evolution in a flow line with regular process times. For each successive bottleneck α, $1 \le i < \sigma$,

$$Y^{\alpha}(k) = \left[\min\begin{Bmatrix} S^{\alpha}(k), Y^{\alpha}(k-1) \\ +\tau_{\beta(\alpha+1)} + d_{\beta(\alpha+1)}(k-1) \\ -\max\{\tau_{\beta(\alpha)}, a_k^{\alpha} - a_{k-1}^{\alpha} - d_0^{\alpha}(k-1)\} \end{Bmatrix}\right]^+, \quad (18)$$

and for servers j, $\beta(\alpha) \le j < \beta(\alpha+1)$,

$$d_j(k) = \left[\min\begin{Bmatrix} \tau_{\beta(\alpha+1)} + d_{\beta(\alpha+1)}(k+j-\beta(\alpha+1)) - \tau_j, \\ Y^{\alpha}(k) - \sum_{n=j+1}^{\beta(\alpha+1)-1} \begin{bmatrix} \tau_{\beta(\alpha+1)} - \tau_n \\ +d_{\beta(\alpha+1)}(k+n-\beta(\alpha+1)) \end{bmatrix} \end{Bmatrix}\right]^+, \quad (19)$$

where $[.]^+ = \max\{0,.\}$ and

$$d_0^1(k) = [a_{k-1} - a_k + \tau_1 + d_0(k-1) + d_1(k-1)]^+, \quad (20)$$
$$d_0^{\alpha}(k) = 0, \quad \alpha > 1,$$
$$a_k^1 = a_k,$$
$$a_k^{\alpha} = a_k + d_0^1(k) + \sum_{\gamma=1}^{\alpha-1} Y^{\gamma}(k) + \sum_{n=1}^{\beta(\alpha)-1} \tau_n. \quad (21)$$

The initial conditions for the recursive relationship are $Y^{\alpha}(1) = 0$, $d_j(k) = 0$ for $k \le 1$ and $d_0^1(1) = 0$.

The result shows that within each RMC described by a pair of adjacent successive bottlenecks, the delay behaves as in Theorem 4. Further, it characterizes the manner in which these connected RMCs fill and empty of delay.

Example 2: Delay evolution in an example flow line. Consider the initially empty flow line of Example 1. One can readily calculate that $S^1(0) = 90$ and $S^2(k) = 120$, for all k. The arrival time, start time at each server and completion time for eleven customers are provided in Table 1. These have been calculated using the system defining evolution equations (1) of Section 2. For example, customer 1 arrives to the empty system at time $a_1 = 0$ and immediately enters server 1, $x_1(1) = 0$. Since there is no preceding customer (i.e., $x_3(0) = -\infty$), customer 1 enters server 2 immediately after it completes service at time $x_2(1) = \max(x_1(1)+45, x_3(0)) = 45$. Customer 2 arrives at time $a_2 = 45$ and enters the recently vacated server 1, $x_1(2) = 45$. Similarly, $x_2(2) = \max(x_1(2)+45, x_3(1)) = 90$.

In Table 2, the results of Theorem 5 are displayed. Table 2 depicts the values for the cumulative delay $Y^{\alpha}(k)$ in the RMCs of which the flow line consists. Notice that $S^1(k)$ (the sum of maximum delays possible in servers 1, 2 and 3) has an initial value of 90, subsequently increases to 120 (as the downstream channel fills) and resets to the value of 90 once the first RMC drains of customers. Note that within each RCM, the delay occurs first in downstream servers. For example, customer 1 has initial condition $Y^1(1) = 0 = Y^2(1)$ and all delays at the individual servers are also zero, $d_i(1) = 0$. Customer 2 then has $Y^2(2) = \tau_7 - \tau_4 = 30$ and $Y^1(1) = \tau_4 - \tau_1 = 15$. All of these delays are experienced in the module just prior to the end of the channel, that is $d_6(2) = 30$ and $d_3(2) = 15$.

To assist in visualizing the behavior, Figure 2 depicts the total residency time $r_j(k) := d_j(k) + x_j(k)$, for each customer in each server. The y-axis is the time of residency. Note that for customers 1 through 5, the residency times are increasing to 90 starting from the end of the line. Similarly, $r_3(k)$ and $r_2(k)$ increase to 60 then to 90. □

Customer	a_k	$x_1(k)$	$x_2(k)$	$x_3(k)$	$x_4(k)$	$x_5(k)$	$x_6(k)$	$x_7(k)$	$c_1(k)$
1	0	0	45	60	90	150	195	240	330
2	15	45	90	105	150	210	255	330	420
3	45	90	135	150	210	270	330	420	510
4	60	135	180	210	270	330	420	510	600
5	105	180	225	270	330	420	510	600	690
6	435	435	480	495	525	585	630	690	780
7	465	480	525	540	585	645	690	780	870
8	480	525	570	585	645	705	780	870	960
9	570	570	615	645	705	780	870	960	1050
10	690	690	735	750	780	870	960	1050	1140
11	735	735	780	795	870	960	1050	1140	1230

Table 1. Customer evolution based on (1) for the network of Example 2.

Customer	$d_0(k)$	$S^1(k)$	$Y^1(k)$	$d_1(k)$	$d_2(k)$	$d_3(k)$	$S^1(k)$	$Y^2(w)$	$d_4(k)$	$d_5(k)$	$d_6(k)$	$d_7(k)$
1	0	90	0	0	0	0	120	0	0	0	0	0
2	30	90	15	0	0	15	120	30	0	0	30	0
3	45	90	30	0	0	30	120	60	0	15	45	0
4	75	90	45	0	15	30	120	90	0	45	45	0
5	75	90	60	0	30	30	120	120	30	45	45	0
6	0	120	0	0	0	0	120	15	0	0	15	0
7	15	120	15	0	0	15	120	45	0	0	45	0
8	45	120	30	0	0	30	120	75	0	30	45	0
9	0	90	45	0	15	30	120	105	15	45	45	0
10	0	105	0	0	0	0	120	120	30	45	45	0
11	0	135	45	0	0	45	120	120	30	45	45	0

Table 2. Delay evolution within the network of Example 2.

In addition to elucidating the manner in which delay (and hence a customer) evolves within a deterministic flow line, Theorem 5 enables us to assess the consequences of state dependent setups. This is an important class of events to assess as they are common in certain semiconductor

manufacturing tools.

A setup requires that all servers prior to a distinguished server be vacant before the setup may commence. Once the ensuing setup duration has elapsed, the next customer (if it has arrived) will enter production with server 1. Not all customers may elicit a setup from the flow line. Let P(k) denote the server that must be vacant prior to the initiation of a setup of duration $\tau_S(k)$. For customers requiring no setup, simply set P(k) = 1 and $\tau_S(k) = 0$. This phenomenon is readily incorporated into the evolution of Theorem 5.

Let $V_{k-1,P(k)}$ denote the time instant when customer k-1 vacates server P(k). There are two possible restrictions on the initiation of the setup. If the setup prior to customer k must wait until both customer k has arrived and customer k-1 has vacated server P(k), let

$$a_k^* = \max\{a_k, V_{k-1,P(k)}\} + \tau_S(k). \tag{22}$$

If the setup need not wait for the arrival of customer k, set

$$a_k^* = \max\{a_k, V_{k-1,P(k)} + \tau_S(k)\}. \tag{23}$$

Corollary 4 follows directly by application of Theorem 5.

Corollary 4: Vacation time and setups. Let $\alpha_{P(k)}$ denote the index of the successive bottleneck prior to P(k), that is, $\beta(\alpha_{P(k)}) \le P(k) < \beta(\alpha_{P(k)}+1)$. The delay faced by customers subject to setups is as in Theorem 5 with a_k replaced by a_k^* and

$$V_{k-1,P(k)} = a_{k-1}^* + d_0^1(k-1) + \sum_{n=1}^{P(k)} \tau_n + \sum_{\alpha=1}^{\alpha_{P(k)}} Y^{\alpha}(k-1) - \left[\begin{matrix} Y^{\alpha_{P(k)}}(k-1) \\ - \sum_{n=P(k)+1}^{\beta(\alpha_{P(k)}+1)-1} \begin{pmatrix} \tau_{\beta(\alpha_{P(k)}+1)} - \tau_n \\ + d_{\beta(\alpha_{P(k)}+1)}(k-1+n-\beta(\alpha_{P(k)}+1)) \end{pmatrix} \end{matrix} \right]^+ . \tag{24}$$

We have thus far detailed the manner in which customers traverse every RMC of the flow line (which then extends to every server). For each customer, one must maintain a record of the state of each RMC (i.e., $Y^{\alpha}(k)$, $d_{\beta(\alpha)}(k)$, $d_0^{\alpha}(k)$).

IV. Bottleneck RMC and Batch Arrivals

If server P(k) resides in the last RMC, that is, $\beta(\sigma\text{-}1) \le P(k) \le \beta(\sigma)$, one need not carry out the calculations for the evolution of delay in all RMCs to obtain the vacation time, only the last one. This is stated in the next corollary.

Corollary 5: Delay and vacation time in the last RMC. The following recursive evolution exists for the delay in servers j, $\beta(\sigma\text{-}1) \le j < \beta(\sigma)$,

$$Y^{\sigma-1}(k) = \left[\min \begin{Bmatrix} S^{\sigma-1}, Y^{\sigma-1}(k-1) + \tau_B \\ - \max\{\tau_{\beta(\sigma-1)}, a_k^{\sigma-1} - a_{k-1}^{\sigma-1}\} \end{Bmatrix} \right]^+, \tag{25}$$

$$d_j(k) = \left[\min \left\{ \tau_B - \tau_j, Y^{\sigma-1}(k) - \sum_{n=j+1}^{B-1} [\tau_B - \tau_n] \right\} \right]^+, \tag{26}$$

$$d_B(k) = 0, \tag{27}$$

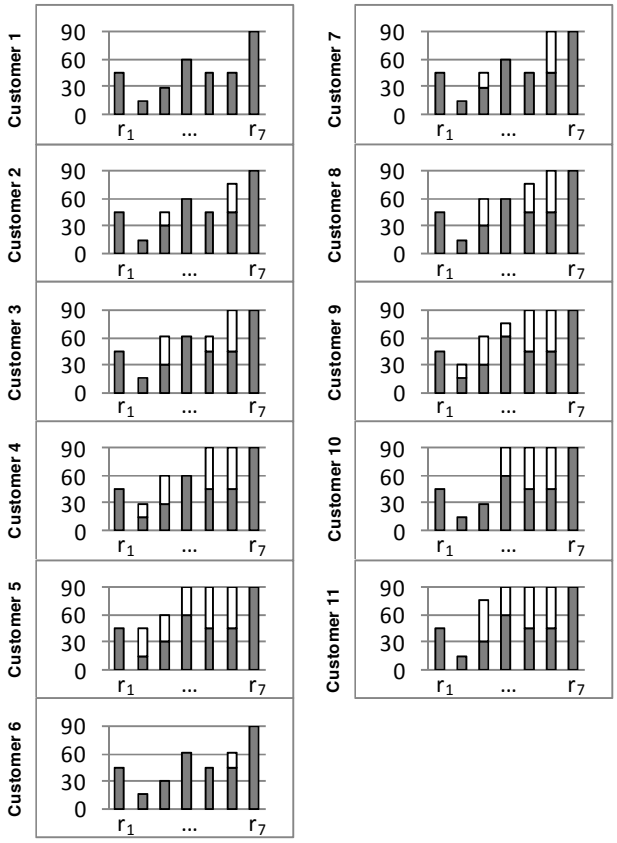

Fig. 2. Residency times for the system of Example 2. The shaded portion indicates service time while the clear portion is the delay.

where

$$a_k^{\sigma-1} := x_{\beta(\sigma-1)}(k) = \max\left\{ a_k + \sum_{n=1}^{\beta(\sigma-1)-1} \tau_n, c_{\beta(\sigma-1)}(k-1) \right\}, \tag{28}$$

$$c_{\beta(\sigma-1)}(k) = a_k^{\sigma-1} + \tau_{\beta(\sigma-1)} + d_{\beta(\sigma-1)}(k).$$

Further, the vacation time $V_{k-1,P(k)}$ is given by

$$V_{k,P(k+1)} = a_k^{\sigma-1} + \sum_{n=\beta(\sigma-1)}^{P(k+1)} \tau_n + \sum_{n=\beta(\sigma-1)}^{P(k+1)} d_n(k). \tag{29}$$

The initial conditions are $Y^{\sigma-1}(1)=0$, $d_j(1)=0$ and $c_{\beta(\sigma-1)}(0)=-\infty$. Note that since $S^{\sigma-1}(k)$ is independent of k, we write only $S^{\sigma-1}$.

Setups may be incorporated by replacing a_k with a_k^* as in Corollary 4 (equation (22) or (23)) when a setup occurs. When there is no setup, use $a_k^* = a_k$.

In semiconductor manufacturing, it is wafers (upon which the semiconductor integrated circuits are being constructed) that act as customers. Such wafers are generally grouped into collections of twelve to twenty-five and called a lot. Our next corollary states that one need not evolve each customer, but may iterate from batch to batch.

First, recall that $c_i(g,w)$ denotes the completion time of the w-th customer of batch g from server i. Let batch g consist of W(g) customers. Abusing notation slightly, we let a_g denote the arrival time of batch g and $\omega_{g,w}$ the customer index of the w-th customer of the g-th group.

Corollary 6: Delay evolution in the last RMC for batch arrivals. The following recursive evolution exists for the delay in servers j, $\beta(\sigma\text{-}1) \le j < \beta(\sigma)$,

$$Y^{\sigma-1}\left(\omega_{g+1,1}\right) = \left[\min \begin{Bmatrix} S^{\sigma-1}, Y^{\sigma-1}\left(\omega_{g,W(g)}\right) + \tau_B \\ -\max\left\{\tau_{\beta(\sigma-1)}, a^{\sigma-1}_{\omega_{g+1,1}} - a^{\sigma-1}_{\omega_{g,W(g)}}\right\} \end{Bmatrix} \right]^+, \quad (30)$$

$$Y^{\sigma-1}\left(\omega_{g+1,W(g+1)}\right) = \min \begin{Bmatrix} S^{\sigma-1}, Y^{\sigma-1}\left(\omega_{g+1,1}\right) \\ +\left(W(g+1)-1\right)\left(\tau_B - \tau_{\beta(\sigma-1)}\right) \end{Bmatrix}, \quad (31)$$

where $d_j(\omega_{g,w})$ obeys the same relation as in Corollary 5 and

$$\begin{aligned} a^{\sigma-1}_{\omega_{g+1,1}} &:= x_{\beta(\sigma-1)}\left(\omega_{g+1,1}\right) \\ &= \max\left\{ a_{g+1} + \sum_{n=1}^{\beta(\sigma-1)-1} \tau_n, c_{\beta(\sigma-1)}\left(\omega_{g,W(g)}\right) \right\}, \end{aligned} \quad (32)$$

$$\begin{aligned} a^{\sigma-1}_{\omega_{g,W(g)}} &:= x_{\beta(\sigma-1)}\left(\omega_{g,W(g)}\right) \\ &= c_{\beta(\sigma-1)}\left(\omega_{g,W(g)}\right) - \tau_{\beta(\sigma-1)} - d_{\beta(\sigma-1)}\left(\omega_{g,W(g)}\right), \end{aligned} \quad (33)$$

$$c_{\beta(\sigma-1)}\left(\omega_{g,W(g)}\right) = a^{\sigma-1}_{W(g)} + W(g)\tau_{\beta(\sigma-1)} + f(g). \quad (34)$$

The function f(g) denotes the sum of the delays experienced by all customers from group g in server $\beta(\sigma\text{-}1)$ and

$$\begin{aligned} f(g) &:= \left(W(g) - k^*_g\right)\left(\tau_B - \tau_{\beta(\sigma-1)}\right) \\ &\quad + \left[Y^{\sigma-1}\left(\omega_{g,1}\right) - S^{\sigma-1} + k^*_g\left(\tau_B - \tau_{\beta(\sigma-1)}\right)\right]^+. \end{aligned} \quad (35)$$

Here $k_g^* \in \{1, \ldots, W(g)\}$ denotes the index of the first customer in batch g to experience delay in server $\beta(\sigma\text{-}1)$ and is given as

$$k^*_g = \min\left\{ W(g), 1 + \max\left\{ k \mid k \le \frac{S^{\sigma-1} - Y^{\sigma-1}\left(\omega_{g,1}\right)}{\tau_B - \tau_{\beta(\sigma-1)}} \right\} \right\}^+. \quad (36)$$

The initial conditions are $Y^{\sigma-1}(\omega_{1,1}) = 0$, $d_j(\omega_{1,1}) = 0$ and $c_{\beta(\sigma-1)}(\omega_{0,W(0)}) = -\infty$. Note that since $S^{\sigma-1}(k)$ is independent of k, we write only $S^{\sigma-1}$.

For each group of customers g, the first customer may require a setup (but no others in the group). To incorporate setups, one proceeds analogously to Corollary 5 to determine a_k^* and $V_{\omega(g,W(g)),P(g+1)}$.

Example 3: Delay evolution for batch arrivals. Consider the flow line of Example 1 with batch arrivals of customers. Let W(g)=12 and set the batch arrival times to a_1=0, a_2=999, a_3=2014 and a_4=3357. Table 3 shows the details of the recursion of Corollary 5.

To illustrate, consider group 1. The first wafer is processed on an initially empty tool and so faces no delay $Y^2(\omega_{1,1}) = 0$. The last wafer of group 1 does experience contention for service, $Y^2(\omega_{1,12}) = \min\{S^2, Y^2(\omega_{1,1}) + [W(1)\text{-}1](\tau_7\text{-}\tau_4)\} = \min\{120, 0+[11](30)\} = 120$. The delay experienced by the last wafer of group 1 in server 4 is $d_4(\omega_{1,12}) = \min\{\tau_7\text{-}\tau_4, Y^2(\omega_{1,12}) - (\tau_7\text{-}\tau_5) - (\tau_7\text{-}\tau_7)\}^+ = \min\{30, 120 - (45) - (45)\}^+ = 30$. Arrival times to the last RMC, $a^2(\omega_{1,1}) = \max\{a_1+\tau_1+\tau_2+\tau_3, -\infty\} = 90$ and $a^2(\omega_{1,12}) = a^2(\omega_{1,1}) + [W(g)\text{-}1]\tau_4 + f(1) - d_4(\omega_{1,12}) = 90 + [11]60+240-30 = 960$. □

Batch (g)	W(g)	a_g	S^2	$Y^2(w_{g,1})$	$Y^2(w_{g,W(g)})$	$d_4(\omega_{g,W(g)})$	$a^2(w_{g,1})$	$a^2(w_{g,W(g)})$	$c_4(w_{g,W(g)})$	k*	f(g)
1	12	0	120	0	120	30	90	960	1050	5	240
2	12	999	120	81	120	30	1089	2040	2130	2	321
3	12	2014	120	120	120	30	2130	3120	3210	1	360
4	12	3357	120	0	120	30	3447	4317	4407	5	240

Table 3. Delay for batches in the last RMC.

V. Concluding Remarks

For flow lines with regular service times, we have detailed the manner in which customers traverse the servers. In particular, it has been shown that the line may be decomposed into what we term resettable monotone channels (RMCs). Within each such channel, the delay a customer faces is predictable and structured. In addition to elucidating the internal dynamics of such a system, we have introduced an additional failure mode that has been inspired by serial processing cluster tool setups in semiconductor wafer fabrication. Knowledge of customer movement is essential for determining the consequences of this state-dependent failure.

Future work could study maximum throughput and design principles. Does any of the structure remain if we include transport robots? Finally, how does delay for customers with different service times evolve?

References

[1] Y. Dallery and S. B. Gershwin, "Manufacturing flow line systems: a review of models and analytical results," *Queueing Systems*, vol. 12, 1992, pp. 3 – 94.

[2] B. Avi-Itzhak, "A sequence of service stations with arbitrary input and regular service times," *Management Science*, vol. 11, no. 5, 1965, pp. 565 – 571.

[3] J. R. Morrison and M. K. Mutnuri, "On the throughput of clustered photolithography tools: Wafer advancement and intrinsic equipment loss," *Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering*, September 2007, pp. 88 – 93.

[4] J. R. Morrison and D. P. Martin, "Performance evaluation of photolithography cluster tools: Queueing and throughput models," *OR Spectrum*, vol. 11, no. 4, 2007, pp. 375 – 389.

[5] J. R. Morrison and D. P. Martin, "Practical extensions to approximations for the G/G/m queue with applications," IEEE Transactions on Automation Science and Engineering, vol. 4, no. 4, 2007, pp. 523 – 532.

[6] F. P. Kelly, "The throughput of a series of buffers," *Advances in Applied Probability*, vol. 14, no. 3, 1982, pp. 633 – 653.

[7] B. Avi-Itzhak and S. Halfin, "Servers in tandem with communication and manufacturing blocking," *Journal of Applied Probability*, vol. 30, 1993, pp. 429 – 437.

[8] B. Avi-Itzhak, "Servers in tandem with k-stage blocking and communications-type flow," *Journal of Applied Probability*, vol. 31, 1994, pp. 1061 – 1069.

[9] B. Avi-Itzhak and M. Yadin, "A sequence of servers with arbitrary input and regular service times revisited," Management Science, vol. 41, no. 6, 1995, pp 1039 – 1047.

[10] B. Avi-Itzhak and H. Levy, "Buffer requirements and server ordering in a tandem queue with correlated service times," Mathematics of Operations Research, vol. 26, no. 2, 2001, pp. 358 – 374.

[11] J. R. Perkins and P. R. Kumar, "Optimal control of pull manufacturing systems," *IEEE Transactions on Automatic Control*, vol. 40, no. 12, 1995, pp. 2040 – 2051.

[12] F. Baccelli, G. Cohen, G. J. Olsder and J. P. Quadrat, *Synchronization and Linearity*, New York : Wiley, 1992.