AN EFFICIENT DISCRETE EVENT MODEL OF UNRELIABLE PRODUCTION LINES

Vassilis S. Kouikoglou

Department of Production Engineering and Management
Technical University of Crete
Chania 73100, Greece
Phone: +30 821 37238, e-mail: kouik@dpem.tuc.gr

Abstract: *This paper develops a discrete event simulation model of production lines with finite buffers and unreliable machines. The model is an extension of a continuous flow model developed by D'Angelo et al. [1], which simulates the system only when a machine's production rate is altered. The events causing changes in the production rates are: machine fails or is repaired and buffer fills or empties. During operation between two events the system runs deterministically. Thus given the state of the system (machine cumulative production, buffer levels, and their statistics) at the time of occurrence of some event, the corresponding state variables upon the occurrence of the next event can be updated using analysis. The proposed model does not use repair events. This is achieved by considering the machine downtimes as transient times in which no parts are produced and developing more complex update equations. A number of numerical results verify that the proposed model is faster than the model of [1] and extremely faster than a conventional piece-by-piece simulator.*

Keywords: *Discrete event simulation, production lines, continuous flow models.*

1. INTRODUCTION

Production lines are commonly used in manufacturing systems producing high volumes of a small number of products. A production line is a serial arrangement of machines. During the operation of the line, a machine may break down and also the other machines are forced to stop production either immediately or after a few production cycles due to blockage or starvation. To avoid delays in production, storage spaces or buffers are introduced between machines and hold semi-finished parts.

The problem of analysis of production lines entails the computation of a number of performance measures, such as the expected throughput, mean buffer levels, mean time in the system, utilization of machines and so on, which are related to profit. Markov chain models are commonly used to analyze production lines. However, exact results concerning the performance measures can be obtained only for lines with buffers of zero or infinite capacities and two-stage systems with a single buffer of finite capacity [2]. For longer lines, the number of states of these models grows geometrically and the analysis demands enormous computational resources. Approximate decomposition models (see e.g. [2] and the references therein) are fast but often unreliable as far as accuracy is concerned. Conventional piece-by-piece simulation is an alternative modeling tool but it spends a lot of time to observe the traffic of parts at each machine and buffer.

This paper develops a hybrid simulation/analytic model of production lines by approximating discrete traffic by a continuous flow. This model is an extension of a continuous flow simulator developed by D'Angelo *et al.* [1], which simulates the system only when a machine's production rate is altered. The events causing changes in the production rates are: machine fails or is repaired and buffer fills or empties. The state of the system at each time instant comprises machine production rates and cumulative production, buffer levels, and their statistics. During operation between two events the system runs deterministically. Thus given the state of the system at the time of occurrence of some event, the state variables upon the occurrence of the next event can be updated using analysis. The proposed model does not use repair events. This is achieved by considering the machine downtimes as transient times in which no parts are produced and developing more complex update equations. A number of numerical results verify that the proposed model is 10-40% faster than the model of [1] and extremely faster than a conventional piece-by-piece simulator, and its errors due to the continuous flow approximation are very small.

2. SYSTEM DESCRIPTION AND CONTINUOUS FLOW APPROXIMATION

A production line is a serial arrangement of n machines M_1, M_2, ..., M_n, with $n-1$ intermediate buffers B_1, ..., B_{n-1}, as shown in Fig. 1. Workpieces enter each machine in sequence and finally exit the last machine as finished products. The production line may be open or closed. In open systems there is an infinite source of raw parts in front of the first machine and an infinite sink for products at the end of the line. If the system is closed, then M_n and M_1 are connected through a buffer.

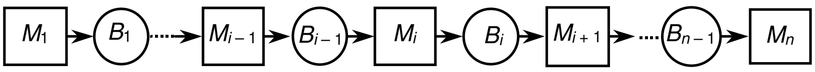

Fig. 1 An open production line.

The storage capacity of B_i is $\mathrm{BC}_i < \infty$. Machine M_i requires $1/R_{M_i}$ time units to complete each part. Hence, the *nominal production rate* of M_i is R_{M_i}. The machines are unreliable and incur failures caused by machine and tool deterioration or they may undergo preventive maintenance after a scheduled amount of production. All these stoppages are operation-dependent because they occur after a random or specific amount of production. Other stoppages caused by exogenous factors, such as power supply failures, can be taken into account in the model without difficulty. To keep the model simple we consider only operation-dependent failures. Furthermore, we assume that machine M_i survives the production of one workpiece with probability $1-p_i$ or breaks down with probability p_i. Let F_i denote the number of parts-to-failure of M_i. Then F_i has a geometric distribution on 0, 1, … with probability mass function $P(F_i = k) = (1-p_i)^k\, p_i$. Finally, we assume that the time-to-repair of M_i is an exponential random variable with mean $1/r_i$, which means that the probability density function is given by $f(x) = r_i\, e^{-r_i x}$, $x \geq 0$. The parameter p_i is the *failure probability* and r_i the *repair rate* of machine M_i. However, the model admits any other distribution of uptimes and downtimes.

In addition to failures, machines may be forced down temporarily due to blockage or starvation. Machine M_i becomes blocked when it is ready to release an item to buffer B_i and the buffer is full. For example, suppose that machine M_{i+1} is slower than M_i and that M_i attempts to release one item to buffer B_i but the buffer is full. Then M_i stays blocked until M_{i+1} releases its part downstream and removes an item from the buffer. Then, both machines start processing new items but M_i finishes its workpiece earlier and becomes blocked again. This alternation of M_i between starved and busy states may go on over a long period until some other event cancels starvation—for example, a failure of M_i. In a dual fashion, the machine becomes starved when it is ready to receive an item and buffer B_{i-1} is empty.

The model we shall develop here approximates discrete traffic by a continuous flow. An immediate consequence of the continuous flow approximation is that, when a buffer B_i becomes full, machine M_i becomes blocked *instantly* and its production rate is reduced to the rate of the downstream machine M_{i+1}. Similarly, the machines M_{i-1}, M_{i-2}, …, may be slowed down instantly if the intermediate buffers B_{i-1}, B_{i-2}, …, happen to be full. Here we have a chain of blocked machines producing synchronously at a reduced rate. In a dual fashion, there may be chains of empty buffers connecting starved machines that produce synchronously at a reduced rate. In discrete part systems, transitions to starved and blocked states and rate reductions do not occur immediately after the buffer-empty and buffer-full states but *after an elapsed time*. For example, if B_i is full then M_i becomes blocked after it finishes its workpiece. Such phenomena have been examined in [3] and are very important for the analysis of systems with small production volumes (e.g. when the system starts empty and during the initial transient period of operation). The assumption of continuous flow implies that a machine can produce infinitesimal quantities within infinitesimal periods of time. Thus, machine cumulative production and buffer levels may be fractional quantities and the transient phenomena associated with discrete traffic are ignored.

3. DISCRETE EVENT MODEL

3.1. Model Overview

The model observes the occurrence of the following events: "a machine fails", "a buffer fills" (*buffer-full event*), and "a buffer empties" (*buffer-empty event*). As we discussed in the previous section, the buffer-full and buffer-empty events force the faster machine slow down to the rate of the slower machine. Therefore, at any time t, the actual production rate of M_i is at most equal to the nominal rate R_{M_i}. The model presented in [1] uses one more event, namely, "a machine is repaired" (*repair event*). In that model, when a machine breaks down, its actual production rate is set to zero and when the machine is repaired it assumes the nominal value. The model we develop here is more efficient than [1] because it does not simulate the repair events.

Elimination of the repair event is achieved as follows. In the model, the operation of machine M_i *from time t on* is described by the following two state variables:

– a state variable $s_i(t)$, $s_i(t) \geq 0$, called the *transient time of M_i at time t*, which denotes the remaining time until the machine starts producing;
– a state variable $R_i(t)$, $R_i(t) \geq 0$, called the *production rate of M_i at time t*, which is the actual production rate of M_i *after* time $t + s_i(t)$.

In the interval $[t, t + s_i(t))$ the machine is idle.

If at time t the machine is operational and produces at its nominal or a reduced rate then we have $s_i(t) = 0$ and $R_i(t)$ is the actual production rate of M_i.

If M_i breaks down at time t, then $s_i(t)$ is set equal to the time until M_i is repaired, which is given by the random variate generator [4]

$$s_i(t) = -\frac{\ln u}{r_i} \tag{1}$$

where u is a random number with uniform distribution on (0,1). Furthermore, the production rate of M_i is set to its nominal rate, that is,

$$R_i(t) = R_{M_i} \tag{2}$$

Similarly, for each buffer B_i the model uses two transient times, namely, $a_i(t)$ which denotes the *remaining time until material starts flowing into the buffer* and $d_i(t)$ denoting the *remaining time until material starts flowing out of the buffer*. Since buffer B_i connects machines M_i and M_{i+1}, $a_i(t) = s_i(t)$ and $d_i(t) = s_{i+1}(t)$. The reason for introducing the transient times of buffer B_i, although, at a given time instant t, they are equal to the corresponding state variables of the adjacent machines M_i and M_{i+1}, is that the model does not record these quantities continuously but updates them only when an event affects the corresponding component (machine or buffer). For example, if at time t machine M_{i+1} breaks down, the model uses Eq. (1) to compute the transient time $s_{i+1}(t)$ of M_{i+1} and adjusts the transient time of B_i as specified above, that is, $d_i(t) = s_{i+1}(t)$. Now, if at some future time $T \geq t$ during the repair period buffer B_i happens to fill, the model computes the transient times of B_i at that time, but does not update the transient time of M_{i+1} because this machine is not affected by the buffer-full event. Hence, at time T the recorded transient times will be $s_{i+1} = s_{i+1}(t)$ and $d_{i+1} = d_{i+1}(T)$. In general, if $t \neq T$ then $s_{i+1} \neq d_{i+1}$. By introducing buffer-related transient times, the model updates only the state variables of the components that are affected by each event and avoids unnecessary computation of the whole state vector.

Finally, the *inflow* and *outflow rates* of buffer B_i are defined by

$$[\text{inflow rate of } B_i \text{ at time } t]=R_i(t)\ ,\ \ [\text{outflow rate of } B_i \text{ at time } t]=R_{i+1}(t)$$

From the above definitions, it turns out that the *inflow rate* of B_i is the actual input rate from time $t + a_i(t)$ on, whereas in the interval $[t, t + a_i(t))$ the buffer does not receive any material. In a dual fashion, the outflow rate of B_i is the actual output rate from time $t + d_i(t)$ on, whereas in the interval $[t, t + d_i(t))$ there is no material flow from B_i to M_{i+1}.

In the sequel, for the sake of brevity, we suppress the argument t in the notation of the state variables. The other state variables used in the model are the following:

F_i remaining number of parts until M_i fails

P_i cumulative production of M_i
BL_i level of buffer B_i, $0 \le BL_i \le BC_i$
T_{Mi} time of next failure at machine M_i
T_{Bi} time of next event at buffer B_i
τ_{Mi} time of the most recent event that affected the production rate or transient time of M_i
τ_{Bi} time of the most recent event that affected the flow rates or transient times of B_i.

The model proceeds as follows.

(a) In the beginning of the simulation, the model initializes the simulation clock and the times of the most recent event at each component (machine or buffer) of the system, that is, $t = \tau_{Mi} = \tau_{Bi} = 0$. Then, the model uses analysis to compute the times of next events at each component. The corresponding equations will be referred to as the *scheduling* equations.
(b) The model determines the most imminent event of the system, namely, the one with the earliest occurrence time and advances t to that time.
(c) Right before the execution of this event, the model *updates* the state variables of the components (machines and buffers) that will be affected by this event. The update equations give the new values of machine cumulative production, transient times, buffer levels, and their statistics using values of the state variables at the time of the most recent (previous) event—this time is τ_{Mi} for machine M_i and τ_{Bi} for buffer B_i.
(d) Upon the occurrence of an event at, say, time t, the production rates and transient times of the affected components are appropriately *adjusted*. Time t is now the new time of the most recent event for these components, i.e., $\tau_{Mi} = t$ and $\tau_{Bi} = t$.
(e) The next events at the affected components are re-scheduled by computing new times of next events and returns to step (b).

The above procedure is repeated until a specified amount of parts are produced.

The model is *hybrid* because it combines analysis and discrete event simulation. Thus piece-by-piece computations are avoided. In the next subsections, we consider the following problems: Given the buffer levels, transient times, parts-to-failure, and production rates at some time instant,

- compute the cumulative production of the machines, the buffer levels, and the transient times right before the occurrence of an event (*update equations*)
- adjust the transient times and the flow rates of the components that are affected by that event (*adjusting equations*)
- and schedule the next events at each affected component (*scheduling equations*).

3.2. Update Equations

Suppose that at time t an event takes place that will affect machine M_i and buffer B_i. The state variables of the affected components are updated right before the occurrence of this event. Let primed quantities denote the values of the state variables at the time of the most recent event that affected the production rate or transient time of the corresponding component. Specifically, for machine M_i we have the following state variables:

τ_{Mi} time of the most recent event that affected machine M_i, $\tau_{Mi} \le t$
F_i (F_i') number of parts-to-failure of M_i at time t (τ_{Mi})

P_i (P_i') cumulative production of M_i at time t (τ_{Mi})
s_i' transient time of M_i (remaining time until M_i starts producing) at time τ_{Mi}
R_i' production rate of M_i at time τ_{Mi}.

If $t \le \tau_{Mi} + s_i'$, then the machine will not produce any items in the interval [τ_{Mi}, t); otherwise it will produce a total of $(t - \tau_{Mi} - s_i')R_i'$ parts. Note that this quantity may be fractional due to the continuous flow assumption. Thus the update equations for M_i are:

$$P_i = \begin{cases} P_i' + (t - \tau_{Mi} - s_i')R_i' & \text{if } (t - \tau_{Mi} - s_i') > 0 \\ P_i' & \text{if } (t - \tau_{Mi} - s_i') \le 0 \end{cases}, \quad F_i = \begin{cases} F_i' - (t - \tau_{Mi} - s_i')R_i' & \text{if } (t - \tau_{Mi} - s_i') > 0 \\ F_i' & \text{if } (t - \tau_{Mi} - s_i') \le 0 \end{cases}$$

For buffer B_i we have the following state variables

τ_{Bi} time of the most recent event that affected buffer B_i, $\tau_{Bi} \le t$
BL_i (BL_i') level of B_i at time t (τ_{Bi})
a_i', d_i' transient times of B_i at time τ_{Bi}
R_i', R_{i+1}' inflow and outflow rate of B_i at time τ_{Bi}.

The level of B_i at time t is computed from

$$\begin{aligned} \mathrm{BL}_i &= \mathrm{BL}_i' + \left[\text{input into } B_i \text{ during } [\tau_{Bi}, t)\right] - \left[\text{output from } B_i \text{ during } [\tau_{Bi}, t)\right] \\ &= \mathrm{BL}_i' + \begin{Bmatrix} (t - \tau_{Bi} - a_i')R_i' & \text{if } (t - \tau_{Bi} - a_i') > 0 \\ 0 & \text{if } (t - \tau_{Bi} - a_i') \le 0 \end{Bmatrix} - \begin{Bmatrix} (t - \tau_{Bi} - d_i')R_{i+1}' & \text{if } (t - \tau_{Bi} - d_i') > 0 \\ 0 & \text{if } (t - \tau_{Bi} - d_i') \le 0 \end{Bmatrix} \end{aligned}$$

Working similarly, we can derive the update equation of the average buffer level in the interval [τ_{Bi}, t). This equation is omitted for the sake of brevity.

Next we derive the update equations for the transient time s_i of M_i at time t. Since M_i is idle in [τ_{Mi}, $\tau_{Mi} + s_i'$], s_i is positive only if t lies in this interval. Hence

$$s_i = \begin{cases} \tau_{Mi} + s_i' - t & \text{if } \tau_{Mi} + s_i' > t \\ 0 & \text{if } t \ge \tau_{Mi} + s_i' \end{cases}$$

The update equations of the transient times of B_i are derived similarly.

Finally, the throughput TH of the system during a period [0, $t_{\max}$] is given by the average throughput rate of the last machine, that is,

$$\mathrm{TH} = \frac{P_n}{t_{\max}}$$

3.3. Adjusting Equations

The model observes changes of flow rates and transient times caused by machine failures, blockage and starvation. Due to the continuous flow assumption, these changes propagate instantly to the beginning and the end of the production system through *chains* of consecutive blocked and starved machines. Specifically, we have the following cases.

(1) If the rate of M_{i+1} is smaller than that of M_i, then B_i will become full. When this happens the rate of M_{i+1} is not altered, but machine M_i becomes blocked instantly and is forced to run at a slower rate $R_i = R_{i+1}$. If there is a chain of blocked machines M_{i-1}, M_{i-2},

…, M_{i-k}, the algorithm is repeated upstream by setting $R_{i-m} = R_{i-m+1}$ for $m = 1, 2, \ldots, k$, until a non-full buffer B_{i-k-1} is reached. Here we have a block of machines collapsed into one with rate $R_{i-m} = R_{i+1}$. Since the rate of a blocked machine is less than its nominal production rate, the above recursion can be written equivalently as

$$R_{i-m} = \min \{R_{i-m+1}, R_{M_{i-m}}\} \tag{3}$$

for $m = 1, 2, \ldots, k$. In some cases, blockage of M_i can be caused by a prolonged idle period of M_{i+1}, although the production rate of M_{i+1} may be greater than that of M_i. This phenomenon happens when M_{i+1} breaks down. Then, the transient times a_i, s_i, and d_{i-1} of buffer B_i, machine M_i, and buffer B_{i-1} are set equal to d_i. Indeed, since buffer B_i is full and does not send any parts downstream during the next d_i time units, it also cannot accept any items during this period. Hence $a_i = d_i$. Similarly, we adjust the transient time of M_i, $s_i = d_i$, and also the transient time of B_{i-1}, $d_{i-1} = d_i$. If there is a chain of blocked machines M_{i-1}, M_{i-2}, …, M_{i-k}, the algorithm is repeated upstream by setting

$$d_{i-m-1} = s_{i-m} = a_{i-m} = d_{i-m}$$

for $m = 1, 2, \ldots, k$, until a non-full buffer B_{i-k-1} is reached. Note that B_{i-m} was full before the occurrence of the buffer-full event at B_i; hence, its transient times right before time t are equal, that is, $a'_{i-m} = d'_{i-m}$. Since the new transient time d_{i-m} is greater than or equal to the transient time before the occurrence of the buffer-full event, the above recursion can be written equivalently as

$$\begin{aligned} a_{i-m} &= \max \{d_{i-m}, a'_{i-m}\} \\ d_{i-m-1} &= s_{i-m} = a_{i-m} \end{aligned} \tag{4}$$

for $m = 1, 2, \ldots, k$. Equations (3) and (4) will be referred to as the *buffer-full algorithm*.

(2) When buffer B_i becomes empty, in a dual fashion, we reduce the rate of M_{i+1} to the rate of M_i and increase the transient times d_i, s_{i+1}, and a_{i+1}. If there is a chain of starved machines M_{i+2}, M_{i+3}, …, downstream of M_{i+1}, then these machines are also forced to run at a slower rate and incur the delay, if any, of M_i. The *buffer-empty algorithm* is as follows:

$$R_{i+m} = \min \{R_{i+m-1}, R_{M_{i+m}}\} \tag{5}$$

$$\begin{aligned} d_{i+m-1} &= \max \{a_{i+m-1}, d'_{i+m-1}\} \\ a_{i+m} &= s_{i+m} = d_{i+m-1} \end{aligned} \tag{6}$$

for $m = 1, 2, \ldots, k$, until a non-empty buffer B_{i+k} is reached.

(3) When M_i breaks down we adjust its transient time and production rate according to Eqs. (1) and (2). The latter equation sets the production rate of M_i to its nominal value, whereas the former sets the transient time of M_i equal to the time-to-repair. A random variate generator is invoked to compute the remaining number of parts until the next failure of M_i. Since F_i has a geometric distribution with parameter p_i, its random variate generator is given by [4] $F_i = \lfloor (\ln u)/[\ln (1-p_i)] \rfloor$, where $\lfloor x \rfloor$ denotes the greatest integer less than x. Next, we adjust the transient time d_{i-1} of buffer B_{i-1} by setting $d_{i-1} = s_i$ and apply the buffer-full algorithm to the components B_{i-1}, M_{i-1}, B_{i-2}, …, until a non-full buffer is reached. In a dual fashion, we adjust the transient time a_i of buffer B_i by setting $a_i = s_i$ and apply the buffer-empty algorithm to the components B_i, M_{i+1}, B_{i+1}, …, until a non-empty

buffer is reached.

3.4. Scheduling Equations

After adjusting the state variables, the model computes the times of next events at the affected components. Let T_{Mi} denote the time of next event at machine M_i and T_{Bi} the time of next event at buffer B_i. First, we derive the event scheduling equations of buffer B_i. We consider the following cases:

(1) If the transient period d_i (the time until M_{i+1} starts removing material from B_i) is sufficiently long, then B_i will fill at time

$$T_{Bi} = t + a_i + \frac{\mathrm{BC}_i - \mathrm{BL}_i}{R_i}$$

The condition for this event is $t + d_i > T_{Bi}$, which is equivalent to

$$d_i - a_i > \frac{\mathrm{BC}_i - \mathrm{BL}_i}{R_i} \tag{7}$$

(2) If the transient period a_i (the time until M_i starts sending material to B_i) is sufficiently long, then B_i will empty at time

$$T_{Bi} = t + d_i + \frac{\mathrm{BL}_i}{R_{i+1}}$$

The condition for this event is $t + a_i > T_{Bi}$, which is equivalent to

$$a_i - d_i > \frac{\mathrm{BL}_i}{R_{i+1}} \tag{8}$$

(3) If none of (7) and (8) holds then we have three possibilities:

(3i) If $R_i > R_{i+1}$, then B_i will become full. From the conservation of flow we have

$$\mathrm{BL}_i(T_{Bi}) = \mathrm{BC}_i = \mathrm{BL}_i(t) + \begin{pmatrix}\text{input into } B_i \\ \text{during } [t + a_i,\, T_{Bi})\end{pmatrix} - \begin{pmatrix}\text{output from } B_i \\ \text{during } [t + d_i,\, T_{Bi})\end{pmatrix}$$

From the above after a little algebra we obtain

$$T_{Bi} = t + \frac{\mathrm{BC}_i - \mathrm{BL}_i + a_i R_i - d_i R_{i+1}}{R_i - R_{i+1}}$$

(3ii) If $R_i < R_{i+1}$, then B_i will become empty at time T_{Bi} when $\mathrm{BL}_i(T_{Bi}) = 0$. From the conservation of flow after a little algebra we obtain

$$T_{Bi} = t + \frac{\mathrm{BL}_i - a_i R_i + d_i R_{i+1}}{R_{i+1} - R_i}$$

(3iii) If both machines produce at the same rate, after a transient period of length max $\{a_i, d_i\}$, the buffer will stay at the same level. In this case, we schedule a fictitious event at time $T_{Bi} = \infty$.

Next, we derive the event scheduling equation of machine M_i. The machine will break down when it produces F_i parts. This event, will occur at time

$$T_{Mi} = t + s_i + \frac{F_i}{R_i}$$

4. NUMERICAL RESULTS AND CONCLUSION

This section reports on the computer implementation and efficiency of the hybrid model. The following simulators were developed in FORTRAN 77 code: a piece-by-piece (PP) simulator and two hybrid, continuous flow models, namely, CF1, which was proposed in [1], and CF2, which was described in the previous section. To compare the models under the same experimental conditions, we use common sequences of random numbers, each one dedicated to a certain random variable (number of parts-to-failure or time-to-repair) and a given machine. The issues investigated are those of accuracy and computational efficiency. The relative speed, RS, of the hybrid models with respect to the PP model is measured by

$$\text{RS} = \frac{\text{CPU time of PP model}}{\text{CPU time of hybrid model}}$$

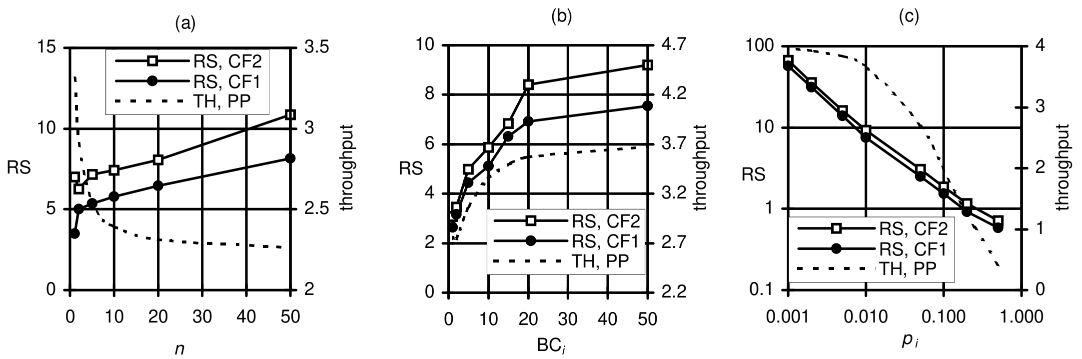

Figure 2. Relative speed of the hybrid models versus system size (a), buffer capacities (b), and failure probabilities (c).

First, we investigate how n, the length of the line, affects the relative speed of the hybrid models. We consider a production line consisting of identical machines and buffers with parameters $R_{Mi} = 5$, $p_i = 0.02$, $r_i = 0.2$, and $\text{BC}_i = 50$. The total production for terminating the simulation is 1,000,000 items. Figure 2(a) shows the relative speed and the throughput TH for lines with varying lengths. From the figure we see that the proposed model is faster than CF1, whereas the speeds of both models with respect to the PP simulator are increasing with the line length. The throughput errors of the hybrid models due to the continuous flow approximation are less than 0.3% in all cases. The errors in estimating the average buffer levels are less than 0.6%.

Next we consider a ten-machine production line consisting of *two identical blocks* of five machines. The first five machines have nominal rates $R_{M1} = 5$, $R_{M2} = 4$, $R_{M3} = 6$, $R_{M4} = 7$, $R_{M5} = 8$. The repair rates are $r_i = 0.5$. Two experiments are conducted to determine the effects of buffer capacities and failure frequency on the performance of the hybrid models. In the first experiment we simulate the line with $p_i = 0.01$ and various buffer capacities. The relative speeds of the hybrid models are given in Fig. 2(b). For small capacities ($BC_i = 1$) the throughput error of the hybrid models is 3.93% and the errors of the average buffer levels were between 0.00% and 6.78%. As buffer capacities increase, the accuracy of the hybrid models increases. For $BC_i \geq 10$, the maximum errors are 0.48% for the throughput and 2.85% for the average buffer level. The CF2 model is faster than CF1 whereas the efficiency of both models improves as buffer capacities increase.

The key condition under which the hybrid algorithms will deliver superior computational efficiency is that events occur at a frequency with order of magnitude smaller than the machine production rates. The latter determine the frequency of events for the PP simulator. Indeed, when buffer capacities are large, blockings occur rarely and the efficiency of the hybrid models is remarkable. To further support the above conjecture we investigate the effect of increasing failure probabilities on RS. We used the same line with space for 50 items in each buffer. The throughput errors of the hybrid model were less than 0.25% in all cases. The results, shown in Fig. 2(c), suggest that there is a critical frequency of failures, below which we have RS ≥ 1, that is, the hybrid models are faster than the PP benchmark. For the particular experiment herein the critical failure probability for the CF2 model is $p_i \approx 0.3$, which implies that the machines break down once every $E(F_i) = (1 - p_i)/p_i = 2$ parts, on the average. For the CF1 model, the corresponding critical values are $p_i \approx 0.21$ and $E(F_i) = 3.75$ parts. However, such frequent failures are rarely the case in actual production lines. If the line is completely reliable, after a transient period, all machines are slowed down to the rate of the slowest machine and no event takes place thereafter. In this case, the hybrid models are infinitely faster than the PP simulator. Therefore, the hybrid models can efficiently analyze relatively reliable systems.

The proposed model is more suitable for large production systems for which traditional simulation is computationally inefficient. The model can be extended to more complex systems involving assembly and disassembly operations, batch production, and random processing times. Discrete and random traffic can be approximated by a *piecewise deterministic fluid flow* using a simple method proposed in [5]. These issues are the subjects of ongoing research.

REFERENCES

[1] H. D'Angelo, H., M. Caramanis, S. Finger, A. Mavretic, Y. A. Phillis, and E. Ramsden, "Event-driven model of unreliable production lines with storage," *Int. J. Prod. Res.* vol. 26, 1173-1182, 1988.

[2] J. A. Buzacott and J. G. Shanthikumar, *Stochastic Models of Manufacturing Systems*, Englewood Cliffs, NJ: Prentice Hall, 1993.

[3] V. S. Kouikoglou and Y. A. Phillis, "An exact discrete-event model and control policies for production lines with buffers," *IEEE T. Automat. Contr.*, vol. 36, 515-527, 1991.

[4] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*, 2nd ed., New York: McGraw-Hill, 1991.

[5] V. S. Kouikoglou and Y. A. Phillis, "Discrete event modeling and optimization of production lines with random rates," *IEEE T. Robotic. Autom.*, vol. 10, 153-159, 1994.