EURANDOM PREPRINT SERIES
2011-025

Approximate performance analysis of production lines with continuous material flows and finite buffers

Remco Bierbooms, Ivo J.B.F. Adan, Marcel van Vuuren
ISSN 1389-2355

APPROXIMATE PERFORMANCE ANALYSIS OF PRODUCTION LINES WITH CONTINUOUS MATERIAL FLOWS AND FINITE BUFFERS

Remco Bierbooms, Ivo J.B.F. Adan, and Marcel van Vuuren
Eindhoven University of Technology, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands
E-mail: r.bierbooms@tue.nl, iadan@tue.nl, vanvuuren@cqm.nl

Abstract: *In this paper, we analyze production lines consisting of a number of machines or servers in series with a finite buffer between each pair of machines. The flow of products through the machines is continuous. Each machine suffers from breakdowns, because of, for example, failures, cleaning and changeover. The up- and downtimes are independent and generally distributed. We develop a new method to efficiently and accurately estimate the throughput and the mean buffer content of the production line. This method relies on decomposition of the production line into two-stage, one-buffer subsystems aggregating the up- and downstream part of the line. For each subsystem, the parameters of the aggregate up- and downtimes are determined iteratively by employing matrix-analytic techniques. The proposed method performs very well on a large test set consisting of over 49,000 cases. Remarkably, the performance of the method does not deteriorate in case of highly unpredictable up- and downtimes, as often seen in practice. We apply the method to a bottling line at brewery Heineken Den Bosch and an assembly line at NXP Semiconductors.*

Keywords: *approximation, decomposition, finite buffer, fluid flow, matrix-analytic method, production line*

1 Introduction

This paper considers production lines consisting of a number of machines in series. Each pair of machines is separated by a finite buffer. The flow through the machines is continuous. Figure 1 shows an example of a four-machine production line, where M_i is the ith machine and B_i is the ith buffer in between M_{i-1} and M_i. The size of buffer B_i is b_i. Machine M_i produces at a maximum speed of s_i units per time unit, but M_i adjusts its speeds to the speed of M_{i-1} when it depletes upstream buffer B_{i-1}, and to the speed of M_{i+1} when it fills up downstream buffer B_i. Each machine suffers from breakdowns. When a breakdown occurs, the machine is immediately taken into repair. During repair, the machine is not able to produce (i.e., it is down), possibly causing starvation of downstream machines and blocking of upstream machines. Successive up- and downtimes are assumed to be independent and generally distributed. Further, the lengths of the uptimes are assumed to be independent of the machine speed. The lenght of an uptime U_i of M_i is characterized by rate λ_{U_i} and coefficient of variation (cv) c_{U_i}, where the scv is defined as the variance divided by the squared mean. The length of a downtime (or repair time) V_i of M_i is characterized by rate λ_{V_i} and cv c_{V_i}. We assume that a machine cannot break down while it is not producing because of starvation or blocking, referred to as "operational dependent failures" [5]. Hence, during starvation or blocking, the uptime of M_i is "frozen" and resumed when M_i starts producing again. The production lines described above are too complex to analyze exactly. Therefore, we aim to find a robust and efficient method to approximate the performance of such lines.

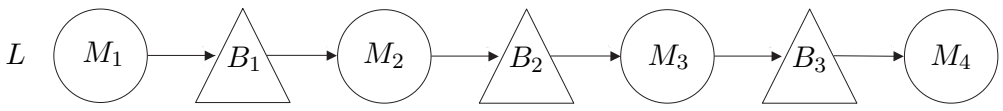

Figure 1: A production line L with four servers, labeled M_1 up to M_4

Fluid flow models are natural to describe production systems producing continuous material, but such

models also appear to be very useful in describing discrete production systems, especially when discrete parts are produced at very high speed. The advantage of continuous models is that these models simplify and speed up discrete-event simulation (in comparison to discrete systems), and, as demonstrated in this paper, their performance can also be accurately and efficiently assessed analytically. Examples of discrete systems that can be adequately described by fluid flow models can be found in data communication networks and semiconductor manufacturing. In fact, our research is inspired by two real-life discrete production lines. The first one is a bottling line at the Heineken brewery in 's-Hertogenbosch. This line consists of eleven machines in series, in which retour bottles are filled and processed. The machine speeds are very high compared to the uptimes of the machines, making it natural to model the flow of bottles as being continuous. A conveyor belt transports bottles from one machine to the next, and it also acts as buffer in case of machine breakdowns due to, for example, failures, cleaning and changeover. The up- and downtimes are obtained from large files or data, showing high variations in up- and downtimes; especially the squared coefficients of variation of downtimes are very high. The second one is an assembly line at NXP semiconductors in Guangdong, China, consisting of eight machines in series. A tape (or lead frame) flows through the machines attaching small electronic components. The tape between each pair of machines can continuously vary in length, up to maximum, which can be seen as the buffer size. Also in this case, we find high variations in up- and downtimes.

There is a huge literature on production lines with continuous flows. Lines consisting of two machines and exponential (or phase type) up- and downtimes can be analyzed exactly by modeling it as a Markov chain, the steady-state distribution of which satsifies a set of linear differential equations. A straightforward approach to solve these equations is by using spectral analysis or matrix exponential functions, see, e.g., [11, 12, 19, 23, 25]. However, since numerical problems arise in case of large buffers, another approach has been developed recently. This approach is initiated by a connection between fluid-flow production lines and discrete quasi-birth-and-death-processes (QBDs) as found by Adan et al. [1] and Ramaswami [20]. Soares and Latouche [21] construct a numerically stable approach for two-stage production lines with continuous flows and time-dependent failures, using a mixture of two matrix exponential functions. Using matrix-analytical techniques, as typically used for discrete QBDs, they determine the steady-state distribution. Their method is adapted in [22] for the case of operational dependent failures. We adopt this method as a building block in our approximation for longer production lines.

For production lines consisting of more than two machines, we have to rely on approximative methods. De Koster [15] made a first attempt by repeatedly aggregating two machines and one buffer into a single machine, until a two-stage production line remains to be analyzed. This method works well for lines with equal machine speeds. However, for lines with non-equal machine speeds, it can produce large errors. The idea of decomposition was introduced by Gershwin [11] to analyze production lines with equal machine speeds, and it was later improved by Dallery et al. [6]. The first approximation for lines with non-equal machine speeds and exponentially distributed up- and downtimes was constructed by Burman [4]. This paper assumes a linear dependence between machine speeds and the rate at which machines break down. The approximation of Burman [4] suffers from convergence problems, which may be avoided by (ad-hoc) modifications in the algorithm. Bierbooms et al. [2] propose another decomposition method, where new decomposition equations are combined with the use of matrix-analytic techniques to solve the two-stage subsystems. This method has no convergence problems, however it assumes exponential up- and downtimes. Levantesi et al. [17] deal with phase-type up- and downtimes. This approximation performs well, but it is unable to handle long production lines because of a state space explosion.

Another approach is based on the use of homogenization techniques, see e.g. [7, 10, 18]. These techniques attempt to replace a non-homogeneous production line with non-identical machine speeds by an equivalent homogeneous line. Dallery and Le Bihan [8] propose a combination of several homogenization techniques. Strong results are reported using these techniques. However, homogenization techniques are "case-specific," prohibiting implementation and use in practice.

The novelty of our approach is that it is able to analyze long production lines with *generally distributed* up- and downtimes. The approach relies on decomposition of the production line into subsystems, each subsystem consisting of an "arrival" server, "departure" server, and a buffer in between. The arrival server mimics (or "aggregates") the behavior of the whole upstream part of the production line and the departure server is doing this for the downstream part. This is the key to our approach: instead of keeping track of the phases of the up- and downtimes of all machines and all buffer contents in the upstream and downstream part of the production line, we use aggregation to avoid a state space explosion for long production lines. The unknown parameters of the up- and downtime distributions for the arrival and departure servers are determined iteratively. The equations used to update the parameters are new and provide the crucial ingredient to the iterative algorithm.

We test our method on a large test set of 49,152 cases including long production lines and high variations in up- and downtimes. Over the whole test set, we find an average error of 1.36% in throughput and 0.62% in mean total buffer content compared to simulation. The method in [2], assuming exponential up- and downtime distributions, produces an average error of 5.15% in throughput on this test set. Furthermore, we find strong results for the practical cases of Heineken and NXP. We experience that our method converges rapidly and that conservation of flow is satisfied automatically, without the need of explicitly adding a conservation of flow equation in the algorithm.

In Section 2, we decompose the production line into two-stage, one-buffer subsystems and we specify the elements of a subsystem. Section 3 constructs the iterative method to obtain the throughput and mean total buffer content of the production line as a whole. In Section 4, we analyze a subsystem by going through the steps of the iterative algorithm. Section 5 is devoted to experimental results and discussion on the test cases and the two practical cases. Finally, Section 6 concludes the paper and gives directions for further research.

2 Decomposition

We decompose the production line L into two-stage subsystems, as illustrated in Figure 2. Each subsystem L_i consists of three elements:

- *Arrival server* A_i. In the description of the arrival server we include the influence of the *upstream* part of the production line on machine M_i, comprising starvation and speed adaption. We describe the behavior of this server as a continuous-time Markov chain with $k_A^{(i)}$ states and generator $\mathbf{Q}_A^{(i)}$. Each state of the Markov chain has a corresponding production speed. We define $\mathbf{r}_A^{(i)}$ as the speed vector, the jth element of which denotes the speed in state j, $j = 1, \ldots, k_A^{(i)}$.
- *Buffer* B_i of size b_i.
- *Departure server* D_i. In the description of the departure server we include the influence of blocking and speed adaption caused by the *downstream* part of the production line on machine M_{i+1}. The departure server is modeled as a continuous-time Markov chain with $k_D^{(i)}$ states, generator $\mathbf{Q}_D^{(i)}$, and speed vector $\mathbf{r}_D^{(i)}$.

In the description of A_i and D_i, starvation and blocking play important roles. We formally define starvation and blocking as follows:

- Arrival server A_i is *starved* by the upstream part of the production line if A_{i-1} is down or starved and B_{i-1} is empty.
- Departure server D_i is *blocked* by the downstream part of the production line if D_{i+1} is down or blocked and B_{i+1} is full.

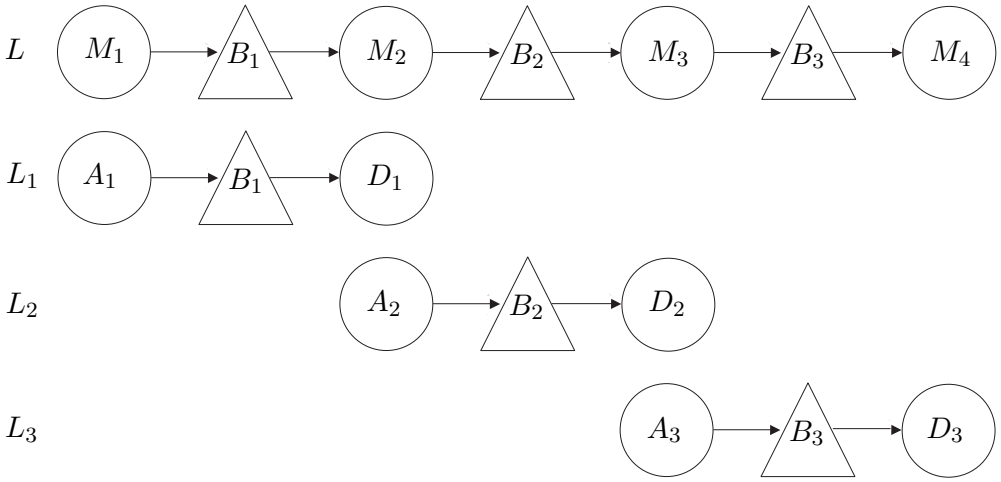

Figure 2: Decomposition of a production line L in three two-stage subsystems L_1, L_2 and L_3

The challenge is to determine the structure of the Markov chains and the elements of $\mathbf{Q}_A^{(i)}$, $\mathbf{Q}_D^{(i)}$, $\mathbf{r}_A^{(i)}$, and $\mathbf{r}_D^{(i)}$, which we do in an iterative way. In the next section, we present the iterative method to obtain these elements, and ultimately, the throughput and mean buffer content of the system.

3 Iterative method

In this section, we construct an iterative method to obtain the throughput and mean buffer content distribution of N-stage production line L. This algorithm relies on the decomposition into subsystems $L_1, ..., L_{N-1}$ as explained in the previous section.

Step 0: Initialization

We assume that initially A_i is not affected by starvation and speed adaption and D_i is not affected by blocking and speed adaption, $i = 1, ..., N$. The corresponding parameters are set accordingly.

Step 1: Evaluation of subsystems

In this step, we analyze all subsystems, starting with L_1 and going onwards to L_N.

(a) Construction of Markov chains for A_i and D_i
Using information from the preceding subsystem, we construct a continuous-time Markov chain describing the behavior of A_i. The elements of $\mathbf{Q}_A^{(i)}$ and $\mathbf{r}_A^{(i)}$ corresponding to this Markov chain are determined. Similarly, we construct a Markov chain for D_i using information from the succeeding subsystem, and we determine the elements of $\mathbf{Q}_D^{(i)}$ and $\mathbf{r}_D^{(i)}$. This step is explained in detail in Section 4.2.

(b) Determination of steady state distribution
We merge the Markov chains for A_i and D_i into a Markov chain with state space $S^{(i)}$ of size $k^{(i)} = k_A^{(i)} \times k_D^{(i)}$, generator $\mathbf{Q}^{(i)}$, and *net speed* vector $\mathbf{r}^{(i)}$. Next, we determine the steady state distribution of subsystem L_i. The cdf $F_j^{(i)}(x)$ of this distribution denotes the probability that subsystem L_i is in state $j \in S^{(i)}$ and the content of buffer B_i does not exceed x, $0 \leq x \leq b_i$. We elaborate further on this step in

Section 4.3.

(c) Determination of throughput estimate
Using the steady state distribution from the previous step, we obtain the estimated throughput $t_h^{(i)}$ of subsystem L_i. The subscript h indicates that the estimate is obtained from iteration h.

(d) Update starvation, blocking, and speed parameters
Following the definition of starvation in Section 2, we obtain information on starvation of A_{i+1} using the steady state distribution of subsystem L_i. If $i < N-1$, we obtain the rate at which A_{i+1} gets starved and the rate and coefficient of variation of the starvation time of A_{i+1}. Since the coefficient of variation of the uptime till starvation is hard to determine, we assume that this transition occurs at an exponential rate. We also determine the speed $s_A^{(i+1)}$ at which A_{i+1} is producing (when A_{i+1} is up) as the *average* of speed s_{i+1}, when D_i is up and B_i is non-empty (note that M_{i+1} is the underlying machine of departure server D_i), and $s_A^{(i)}$, when D_i and A_i are both up and B_i is empty (see (12)). Furthermore, we obtain information on blocking of D_{i-1} using the steady state distribution of L_i. If $i > 1$, we update the rate at which D_{i-1} becomes blocked and the rate and coefficient of variation of the blocking time D_{i-1}. Also for D_{i-1}, we determine the speed $s_D^{(i-1)}$, similar as done for $s_A^{(i)}$ (see (13)). This step is explained in detail in Section 4.4.

Step 2: Repeat
We repeat step 1 until the throughput for all subsystems has converged. If for some small ϵ it holds that

$$\frac{t_h^{(i)} - t_{h-1}^{(i)}}{t_{h-1}^{(i)}} < \epsilon \quad \text{for all} \quad i \leq N-1$$

we stop, otherwise we perform another iteration.

Note that this algorithm does not guarantee that the throughput values for all subsystems are equal. However, this appeared to be true in all experiments performed in Section 5. In the next section, we explain the steps of the iterative algorithm in more detail.

4 Subsystem analysis

This section describes the analysis of subsystem L_i, $i = 1, ..., N-1$, by consecutively going through steps **1(a)-1(d)** in detail. We start by describing a class of distributions called phase-type distributions. Section 4.1 describes how to fit a phase-type distribution on a given rate and (squared) coefficient of variation. Section 4.2 models the behavior of arrival server A_i and departure server D_i as continuous-time Markov chains. The steady state distribution of subsystem L_i satisfies a set of linear differential equations, which are solved in Section 4.3 by using matrix-analytic techniques as in [22]. From this distribution, we update parameters for the arrival server of subsystem L_{i+1} and the departure server of L_{i-1}.

4.1 Phase-type distributions

For every random variable $Y \geq 0$ with rate $\lambda_Y = 1/E(Y)$ and cv $c_Y = \sigma(Y)/E(Y)$, we can determine a phase-type distribution with the same rate and cv, according to the following recipe, see [24]. If $c_Y \leq 1$, we use the Erlang$_{k-1,k}$ distribution to match λ_Y and c_Y, where the integer $k > 1$ satisfies

$1/k < c_Y^2 \leq 1/(k-1)$. As illustrated in the left part of Figure 3, an Erlang$_{k-1,k}$ random variable is with probability p the sum of $k-1$ exponential phases and with probability $1-p$ the sum of k exponential phases. By choosing

$$p = \frac{1}{1+c_Y^2}(kc_Y^2 - (k(1+c_Y^2) - k^2c_Y^2)^{1/2}), \qquad \mu = (k-p)\lambda_Y,$$

the rate and cv of the Erlang$_{k-1,k}$ distribution matches λ_Y and c_Y. If $c_Y > 1$, we fit a Coxian$_2$ distribution with $k = 2$ phases. A Coxian$_2$ random variable is with probability p an exponential phase with rate μ_1 and, with complementary probability, it is an exponential phase with rate μ_1 followed by an exponential phase with rate μ_2; see the right part of Figure 3. To match λ_Y and c_Y, we choose

$$\mu_1 = 2\lambda_Y, \qquad p = 1 - 0.5/c_Y^2, \qquad \mu_2 = \mu_1(1-p).$$

It is also possible to use other phase-type distributions to match λ_Y and c_Y. However, numerical experiments show that the use of other distributions does not significantly affect the results, see [14]. The Erlang$_{k-1,k}$ and Coxian$_2$ distributions of Y can be represented as a transient Markov process with states $1, \ldots, k$, and generator $\mathbf{\Phi}_Y$; the column vector $\boldsymbol{\psi}_Y = -\mathbf{\Phi}_Y \mathbf{1}_k$ contains the rates at which the Markov process exits, where $\mathbf{1}_k$ is the all-one column vector of size k. Then Y is the time till exit, starting in phase 1.

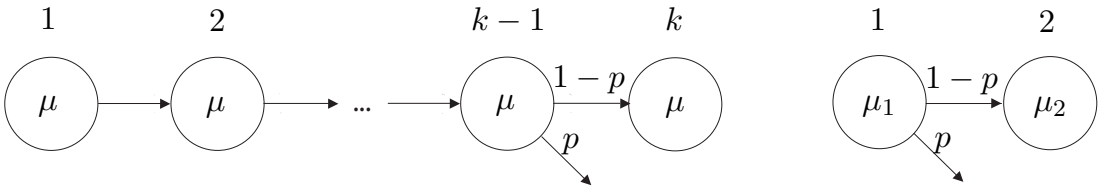

Figure 3: Phase diagram of Erlang$_{k-1,k}$ distribution (left) and Coxian$_2$ distribution (right).

For example, we have the following $\mathbf{\Phi}_Y$ and $\boldsymbol{\psi}_Y$ for the Erlang$_{3,4}$ distribution:

$$\mathbf{\Phi}_Y = \begin{pmatrix} -\mu & \mu & 0 & 0 \\ 0 & -\mu & \mu & 0 \\ 0 & 0 & -\mu & (1-p)\mu \\ 0 & 0 & 0 & -\mu \end{pmatrix}, \qquad \boldsymbol{\psi}_Y = \begin{pmatrix} 0 \\ 0 \\ p\mu \\ \mu \end{pmatrix}.$$

For the Coxian$_2$ distribution, we have

$$\mathbf{\Phi}_Y = \begin{pmatrix} -\mu_1 & p\mu_1 \\ 0 & -\mu_2 \end{pmatrix}, \qquad \boldsymbol{\psi}_Y = \begin{pmatrix} (1-p)\mu_1 \\ \mu_2 \end{pmatrix}.$$

This representation will be used in the next section.

4.2 Behavior of arrival and departure server

In this section, we subsequently model the behavior of arrival server A_i and departure server D_i as a continuous-time Markov chain. For A_i, we divide the state space into three sets: states where A_i is producing (*up*), states where A_i is not producing because of a breakdown of the underlying machine M_i (*down*), and states where A_i is not producing because it has no input (*starved*). This division of states is illustrated in the left part of Figure 4. The three sets are formally defined as follows:

- The set of up-states $S_{A,u}^{(i)}$: A_i is *up* when M_i is up, and either B_{i-1} is not empty or B_{i-1} is empty and A_{i-1} is up.

- The set of down-states $S^{(i)}_{A,d}$: A_i is *down* when M_i is down.
- The set of starved-states $S_{A,st}$: A_i is *starved* when M_i is up, B_{i-1} is empty, and A_{i-1} is down or starved.

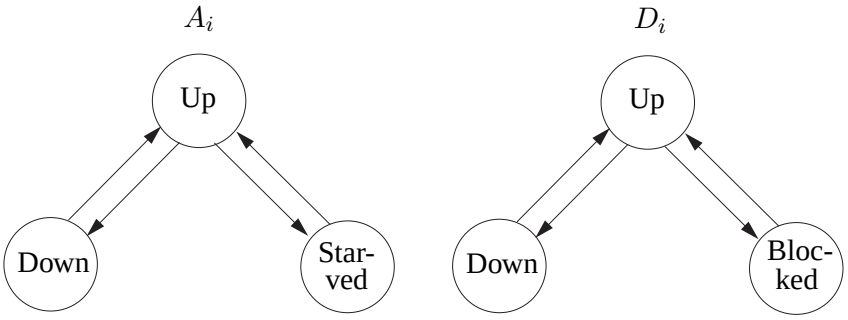

Figure 4: Division of states for A_i and D_i.

We define $\mathbf{Q}^{(i)}_A$ as the generator of the Markov process for A_i, containing the transition rates within and between the three sets of states. The state space of this Markov chain is given by $S^{(i)}_A = S^{(i)}_{A,u} \bigcup S^{(i)}_{A,d} \bigcup S^{(i)}_{A,st}$. Because of operational dependent failures, no transitions are possible from the starved-states to the down-states and vice versa. To obtain the transition rates, we first fit phase-type distributions on the rate and cv of the following random variables:

- The uptime U_i of M_i with rate λ_{U_i} and cv c_{U_i}.
- The repair time V_i of M_i with rate λ_{V_i} and cv c_{V_i}.
- The starvation time $SU^{(i)}$ with rate $\lambda^{(i)}_{SU}$ and cv $c^{(i)}_{SU}$, which follow from the analysis of L_{i-1} (see Section 4.4).

We also determine from the analysis of L_{i-1}:

- The rate $\lambda^{(i)}_{US}$ at which A_i jumps from up to starved; we act as if this rate is exponential.

The number of phases for U_i, V_i and $SU^{(i)}$ are k_{U_i}, k_{V_i}, and $k^{(i)}_{SU}$ respectively, with corresponding generators $\mathbf{\Phi}_{U_i}$, $\mathbf{\Phi}_{V_i}$, $\mathbf{\Phi}^{(i)}_{SU}$, and exit-rate vectors $\boldsymbol{\psi}_{U_i}$, $\boldsymbol{\psi}_{V_i}$, and $\boldsymbol{\psi}^{(i)}_{SU}$ (see Section 4.1).

Before specifying the generator $\mathbf{Q}^{(i)}_A$, we define the Kronecker product: If A is an $n_1 \times n_2$ matrix and B is an $n_3 \times n_4$ matrix, the Kronecker product is defined as

$$A \otimes B = \begin{pmatrix} A(1,1)B & \dots & A(1,n_2)B \\ \vdots & & \vdots \\ A(n_1,1)B & \dots & A(n_1,n_2)B \end{pmatrix}.$$

The generator $\mathbf{Q}^{(i)}_A$ is given by,

$$\mathbf{Q}^{(i)}_A = \begin{pmatrix} \mathbf{Q}^{(i)}_{A,(u,u)} & \mathbf{Q}^{(i)}_{A,(u,d)} & \mathbf{Q}^{(i)}_{A,(u,st)} \\ \mathbf{Q}^{(i)}_{A,(d,u)} & \mathbf{Q}^{(i)}_{A,(d,d)} & 0 \\ \mathbf{Q}^{(i)}_{A,(st,u)} & 0 & \mathbf{Q}^{(i)}_{A,(st,st)} \end{pmatrix},$$

where

$$\mathbf{Q}^{(i)}_{A,(u,u)} = \mathbf{\Phi}_{U_i} - \lambda^{(i)}_{US}\mathbf{I}(k_{U_i}), \quad \mathbf{Q}^{(i)}_{A,(u,d)} = \mathbf{e}_1(k_{V_i}) \otimes \boldsymbol{\psi}_{U_i}, \quad \mathbf{Q}^{(i)}_{A,(u,st)} = \lambda^{(i)}_{US}\mathbf{I}(k_{U_i}) \otimes \mathbf{e}_1(k^{(i)}_{SU}),$$
$$\mathbf{Q}^{(i)}_{A,(d,u)} = \mathbf{e}_1(k_{U_i}) \otimes \boldsymbol{\psi}_{V_i}, \qquad \mathbf{Q}^{(i)}_{A,(d,d)} = \mathbf{\Phi}_{V_i},$$
$$\mathbf{Q}^{(i)}_{A,(st,u)} = \mathbf{I}(k_{U_i}) \otimes \boldsymbol{\psi}^{(i)}_{SU}, \qquad \mathbf{Q}^{(i)}_{A,(st,st)} = \mathbf{I}(k_{U_i}) \otimes \mathbf{\Phi}^{(i)}_{SU}$$

In these expressions, $\mathbf{I}(y)$ is the y-size identity matrix and $\mathbf{e}_1(y)$ is a y-size row vector with first element one and other elements zero. On the diagonal blocks we find the transitions within either of the three sets of states. If A_i is starved, we store ("freeze") the phase of the uptime U_i, resuming in this phase as soon as A_i returns to up again. The total number of states of the Markov chain for A_i is given by

$$k^{(i)}_A = k_{U_i} + k_{V_i} + k^{(i)}_{SU}k_{U_i}.$$

The generator $\mathbf{Q}^{(i)}_D$ of departure server D_i is derived similarly. The following three sets are introduced for D_i (see right part of Figure 4):

- The set of up-states $S^{(i)}_{D,u}$: D_i is *up* when M_{i+1} is up, B_{i+1} is not full or B_{i+1} is full and D_{i+1} is up.
- The set of down-states $S^{(i)}_{D,d}$: D_i is *down* when M_{i+1} is down.
- The set of blocked-states $S^{(i)}_{D,bl}$: D_i is *blocked* when M_{i+1} is up, B_{i+1} is full and D_{i+1} is down or blocked.

The Markov chain of D_i has state space $S^{(i)}_D = S^{(i)}_{D,u} \bigcup S^{(i)}_{D,d} \bigcup S^{(i)}_{D,bl}$. To determine the transition rates of $\mathbf{Q}^{(i)}_D$, we first fit phase-type distributions on the rate and cv of the following random variables:

- The uptime U_{i+1} of M_{i+1} with rate $\lambda_{U_{i+1}}$ and cv $c_{U_{i+1}}$.
- The repair time V_{i+1} of M_{i+1} with rate $\lambda_{V_{i+1}}$ and cv $c_{V_{i+1}}$.
- The blocking time $BU^{(i)}$ with rate $\lambda^{(i)}_{BU}$ and cv $c^{(i)}_{BU}$, determined from the analysis of L_{i+1} (see Section 4.4).

Further, we determine from the analysis of L_{i+1}:

- The rate $\lambda^{(i)}_{UB}$ at which D_i jumps from up to blocked, and we act as if this rate is exponential.

The number of phases for U_{i+1}, V_{i+1} and $BU^{(i)}$ are $k_{U_{i+1}}$, $k_{V_{i+1}}$, and $k^{(i)}_{BU}$, respectively, and the corresponding generators are denoted by $\mathbf{\Phi}_{U_{i+1}}$, $\mathbf{\Phi}_{V_{i+1}}$, $\mathbf{\Phi}^{(i)}_{BU}$, and the exit-rate vectors $\boldsymbol{\psi}_{U_{i+1}}$, $\boldsymbol{\psi}_{V_{i+1}}$, and $\boldsymbol{\psi}^{(i)}_{BU}$. The generator $\mathbf{Q}^{(i)}_D$ is given by

$$\mathbf{Q}^{(i)}_D = \begin{pmatrix} \mathbf{Q}^{(i)}_{D,(u,u)} & \mathbf{Q}^{(i)}_{D,(u,d)} & \mathbf{Q}^{(i)}_{D,(u,bl)} \\ \mathbf{Q}^{(i)}_{D,(d,u)} & \mathbf{Q}^{(i)}_{D,(d,d)} & 0 \\ \mathbf{Q}^{(i)}_{D,(bl,u)} & 0 & \mathbf{Q}^{(i)}_{D,(bl,bl)} \end{pmatrix},$$

where

$$\mathbf{Q}^{(i)}_{D,(u,u)} = \mathbf{\Phi}_{U_{i+1}} - \lambda^{(i)}_{UB}\mathbf{I}(k_{U_{i+1}}), \quad \mathbf{Q}^{(i)}_{D,(u,d)} = \mathbf{e}_1(k_{V_{i+1}}) \otimes \boldsymbol{\psi}_{U_{i+1}}, \quad \mathbf{Q}^{(i)}_{D,(u,bl)} = \lambda^{(i)}_{UB}\mathbf{I}(k_{U_{i+1}}) \otimes \mathbf{e}_1(k^{(i)}_{BU})]$$
$$\mathbf{Q}^{(i)}_{D,(d,u)} = \mathbf{e}_1(k_{U_{i+1}}) \otimes \boldsymbol{\psi}_{V_{i+1}}, \qquad \mathbf{Q}^{(i)}_{D,(d,d)} = \mathbf{\Phi}_{V_{i+1}},$$
$$\mathbf{Q}^{(i)}_{D,(bl,u)} = \mathbf{I}(k_{U_{i+1}}) \otimes \boldsymbol{\psi}^{(i)}_{BU}, \qquad \mathbf{Q}^{(i)}_{D,(bl,bl)} = \mathbf{I}(k_{U_{i+1}}) \otimes \mathbf{\Phi}^{(i)}_{BU}.$$

The number of states is

$$k^{(i)}_D = k_{U_{i+1}} + k_{V_{i+1}} + k^{(i)}_{BU}k_{U_{i+1}}.$$

4.3 Steady state distribution

This section is devoted to the determination of the steady state distribution of subsystem L_i. First, for notational convenience, we drop the subscript i and superscript (i) referring to the ith subsystem. Based on the Markov chains for arrival server A and departure server D, we now construct the Markov chain describing the behavior of the whole subsystem. This Markov chain has state space $S = S_A \bigcup S_D$, generator $\mathbf{Q}$, and net speed vector $\mathbf{r}$. The number of states is given by $k = k_A \times k_D$. By ordering the states of A and D lexicographically, we get

$$\mathbf{Q} = \mathbf{Q}_A \otimes \mathbf{I}_{k_D} + \mathbf{I}_{k_A} \otimes \mathbf{Q}_D,$$

$$\mathbf{r} = \mathbf{r}_A \otimes \mathbf{1}_{k_D} - \mathbf{1}_{k_A} \otimes \mathbf{r}_D,$$

where $\mathbf{I}_n$ is an identity matrix of size $n \times n$ and $\mathbf{1}_n$ is the all-one column vector of size n.

Note that, because of the assumption of operational dependent failures, A cannot go down or starved whenever B is full and D is not producing, and D cannot go down or blocked whenever B is empty and A is not producing. This implies that the transition rates are different in case of an empty or full buffer. Therefore, we introduce a "full-buffer" generator $\mathbf{Q}^F$ and an "empty-buffer" generator $\mathbf{Q}^E$. For $\mathbf{Q}^F$, we argue that when D is in a state with zero-speed and B is full, A cannot jump to a state with zero-speed. In all other situations, the transition rates in $\mathbf{Q}^F$ are the same as the ones in $\mathbf{Q}$. Hence, denoting by $Q^F_{(i_A,i_D)\to(j_A,j_D)}$ the transition rate from state (j_A, j_D) to (j_A, j_D) when B is full, we get

$$Q^F_{(i_A,i_D)\to(j_A,j_D)} = \begin{cases} 0 & \text{if} \quad r_{j_A} = 0 \wedge r_{i_D} = 0, \\ Q_{(i_A,i_D)\to(j_A,j_D)} & \text{else.} \end{cases}$$

Similarly, when A is in a state with zero-speed and B is empty, D cannot jump to a state with zero-speed, yielding

$$Q^E_{(i_A,i_D)\to(j_A,j_D)} = \begin{cases} 0 & \text{if} \quad r_{j_D} = 0 \wedge r_{i_A} = 0, \\ Q_{(i_A,i_D)\to(j_A,j_D)} & \text{else.} \end{cases}$$

The state of subsystem L can be described by the pair of variables (i, x), where $i \in S$ is the state of the phase process and $0 \le x \le b$ is the fluid level of the buffer. We define $F_i(x)$ as the probability that the phase process is in state i and the fluid level does not exceed x, also referred to as the cumulative distribution function (cdf) of subsystem L. Since there is probability mass at the boundary levels 0 and b, we introduce $p_i^{(0)}$ as the probability of being in state $(i, 0)$, and $p_i^{(b)}$ as the probability of being in (i, b). So

$$p_i^{(0)} = F_i(0), \qquad p_i^{(b)} = F_i(b) - \lim_{x \uparrow b} F_i(x).$$

Since the buffer cannot be empty in a state with positive net speed, we have $p_i^{(0)} = 0$ for all $i \in S$ with $r_i > 0$. Similarly, $p_i^{(b)} = 0$ for all $i \in S$ with $r_i < 0$. Besides the boundary probabilities, we define $\pi_i = F_i(b)$ as the probability that the phase process is in state $i \in S$. To derive a system of (differential) equations for $F_i(x)$, we balance the probability flux into and out of the set of states $\{(i, y), 0 \le y \le x\}$:

$$r_i \frac{dF_i(x)}{dx} + \sum_{j \neq i} Q_{i,j} F_i(x) = \sum_{j \neq i} Q_{j,i} F_j(x).$$

Rearranging terms yields

$$\frac{dF_i(x)}{dx} = \sum_{j \in S} \frac{1}{r_i} Q_{j,i} F_j(x) \qquad \text{for all } i \in S \text{ with } r_i \neq 0. \tag{1}$$

and

$$\sum_{j \neq i} Q_{i,j} F_i(x) = \sum_{j \neq i} Q_{j,i} F_j(x) \qquad \text{for all } i \in S \text{ with } r_i = 0. \tag{2}$$

To solve the balance equations for $F_i(x)$, we first remove from (1) all the states $i \in S$ for which $r_i = 0$ by substituting (2); we include these states later. The reduced state space is denoted by $\tilde{S}$ of size $\tilde{k}$. Accordingly, we adjust $\mathbf{Q}$, $\mathbf{Q}^E$, $\mathbf{Q}^F$, and $\mathbf{r}$ to obtain $\tilde{\mathbf{Q}}$, $\tilde{\mathbf{Q}}^E$, $\tilde{\mathbf{Q}}^F$, and $\tilde{\mathbf{r}}$. Hence, (1) reduces to

$$\frac{d\tilde{F}_i(x)}{dx} = \sum_{j \in \tilde{S}} \frac{1}{\tilde{r}_i} \tilde{Q}_{j,i} \tilde{F}_j(x) \qquad \text{for all } i \in \tilde{S}. \tag{3}$$

The equations (3) can be interpreted as the balance equations for the Markov chain *conditional on being in* $\tilde{S}$. Now we define $\tilde{\mathbf{F}}(x)$ as the column vector with elements $\tilde{F}_i(x)$ and $\tilde{\mathbf{f}}(x) = \frac{d\tilde{\mathbf{F}}(x)}{dx}$ as the corresponding probability density vector (pdf). Further, we define $\mathbf{R}$ as the diagonal matrix with (non-zero) diagonal elements $R_{i,i} = \tilde{r}_i$. By writing (3) in matrix-notation and differentiating, we obtain

$$\frac{d\tilde{\mathbf{f}}(x)}{dx} = \mathbf{Z}\tilde{\mathbf{f}}(x),$$

where $\mathbf{Z} = \mathbf{R}^{-1}\tilde{\mathbf{Q}}$. The solution of this differential equation is the matrix-exponential function

$$\tilde{\mathbf{f}}(x) = \mathbf{c}e^{\mathbf{Z}x}, \tag{4}$$

where $\mathbf{c}$ is a $\tilde{k}$-size vector of constants to be determined. We define $\tilde{p}_i^{(0)}$ and $\tilde{p}_i^{(b)}$ as the probabilities of being in state $i \in \tilde{S}$ for the conditional Markov chain. By balancing the probability flux into and out of state $(i, 0)$, we get the following balance equation,

$$\sum_{j \neq i} \tilde{Q}_E(j,i) \tilde{p}_j^{(0)} = \sum_{j \neq i} \tilde{Q}_E(i,j) \tilde{p}_i^{(0)} + \tilde{r}_i \tilde{f}_i(0), \qquad \text{for all } i \in \tilde{S} \text{ with } \tilde{r}_i < 0,$$

and similarly, for state (i, b),

$$\sum_{j\neq i} \tilde{Q}_F(j,i)\tilde{p}_j^{(b)} + \tilde{r}_i\tilde{f}_i(b) = \sum_{j\neq i} \tilde{Q}_F(i,j)\tilde{p}_i^{(b)}, \qquad \text{for all } i \in \tilde{S} \text{ with } \tilde{r}_i > 0,$$

From the boundary equations and the normalization equation we obtain the following set of equations to solve for c, $\tilde{p}_i^{(0)}$ for $i \in \tilde{S}$ with $\tilde{r}_i < 0$, and $\tilde{p}_i^{(b)}$ for $i \in \tilde{S}$ with $\tilde{r}_i > 0$:

$$\sum_{j\in\tilde{S}} \tilde{Q}_E(j,i)\tilde{p}_j^{(0)} - \tilde{r}_i\tilde{f}_i(0) = 0, \qquad \text{for all } i \in \tilde{S} \text{ with } \tilde{r}_i < 0, \tag{5}$$

$$\sum_{j\in\tilde{S}} \tilde{Q}_F(j,i)\tilde{p}_j^{(b)} + \tilde{r}_i\tilde{f}_i(b) = 0, \qquad \text{for all } i \in \tilde{S} \text{ with } \tilde{r}_i > 0, \tag{6}$$

$$\sum_{i\in\tilde{S}} \tilde{F}_i(b) = 1, \tag{7}$$

where

$$\tilde{F}_i(b) = \tilde{p}_i^{(0)} + \int_0^b \tilde{f}_i(x)dx + \tilde{p}_i^{(b)}.$$

In principle, we can determine $\tilde{\mathbf{F}}(x)$ from (5)-(7). However, this solution suffers from numerical instability. The reason is that $e^{\mathbf{Z}b}$ in (4) can become very large, whenever b is large and $\mathbf{Z}$ has (large) positive eigenvalues. Scaling with $e^{-\mathbf{Z}b}$ does not help, since then the same problem appears with the negative eigenvalues. For the same reason, the use of spectral analysis can be numerically instable and time consuming, since $\mathbf{Z}$ may have defective eigenvalues. Therefore, we adopt another approach by splitting the state space in two parts: states with positive net speeds and states with negative net speeds. Each part has its corresponding transition rate matrix. To analyze each part separately (i.e., conditionally), we need "return probabilities." These probabilities can be determined using matrix-analytic techniques, as widely used for discrete QBDs. This method to determine $\tilde{\mathbf{F}}(x)$ is numerically stable, and it is developed in [21, 22] for both time and operational dependent failures. In [2], the method is used in a production line setting with exponential up- and downtimes. In [22, 2], it is also explained how the throughput and mean buffer content can be obtained from the matrix-analytic solution.

Once $\tilde{F}_i(x)$ is known for $i \in \tilde{S}$, the next step is to determine the *unconditional* $F_i(x)$ for $i \in S$. Note that

$$F_i(x) = (1 - \sum_{j\in S_0} F_j(b))\tilde{F}_i(x), \qquad i \in \tilde{S}, \tag{8}$$

where $S_0 = S\backslash\tilde{S}$ denotes the set of states with a net speed of zero. Substitution of (8) in (2) with $x = b$ yields a set of linear equations for the unknown probabilities $F_j(b)$, $j \in S_0$. Hence, the solution to this set of equations yields $F_i(x)$ by (8) and (2), and subsequently $p_i^{(0)}$, $p_i^{(b)}$, and π_i, for all $i \in S$.

4.4 Update parameters

In this section, we present a crucial step in our algorithm: the calculation of new (better) estimates of the parameters for A_{i+1} and D_{i-1}, based on the steady state distribution of subsystem L_i. More specifically, we use the following:

- $\mathbf{\Pi}_{j,l}^{(i)}$, matrix of steady state probabilities, the (m,n)th element of which is the probability that A_i is in state $m \in S_{A,j}^{(i)}$, $j \in \{u,d,st\}$, and D_i is in state $n \in S_{D,l}^{(i)}$, $l \in \{u,d,bl\}$. The scalar $\bar{\pi}_{j,l}^{(i)}$ is the sum of all probabilities in $\mathbf{\Pi}_{j,l}^{(i)}$.
- $\mathbf{P}_{j,l}^{(i)}(0)$ and $\mathbf{P}_{j,l}^{(i)}(b)$, matrices of boundary probabilities at the levels 0 and b. The subscript (j,l) has the same interpretation as for $\mathbf{\Pi}_{j,l}^{(i)}$. The scalars $\bar{p}_{j,l}^{(i)}(0)$ and $\bar{p}_{j,l}^{(i)}(b)$ are the sum of all probabilities in $\mathbf{P}_{j,l}^{(i)}(0)$ and $\mathbf{P}_{j,l}^{(i)}(b)$, respectively.
- $\mathbf{F}_{j,l}^{(i)}(0)$ and $\mathbf{F}_{j,l}^{(i)}(b)$, pdf matrices at the levels 0 and b. The scalars $\bar{f}_{j,l}^{(i)}(0)$ and $\bar{f}_{j,l}^{(i)}(b)$ are the sum of elements in $\mathbf{F}_{j,l}^{(i)}(0)$ and $\mathbf{F}_{j,l}^{(i)}(b)$ respectively. We can also see $\bar{f}_{j,l}^{(i)}(0)$ and $\bar{f}_{j,l}^{(i)}(b)$ as the number of level crossings at level 0 and b in state (j,l), $j \in \{u,d,st\}$, $l \in \{u,d,bl\}$.

We start to determine the rate $\lambda_{US}^{(i+1)}$ at which A_{i+1} jumps from up to starved and the rate $\lambda_{SU}^{(i+1)}$ and cv $c_{SU}^{(i+1)}$ of the starvation time $SU^{(i+1)}$. These parameters are used to specify the generator $\mathbf{Q}_A^{(i+1)}$ (see Section 4.2). Recall that A_{i+1} is starved when M_{i+1} is up, A_i is down or starved and B_i is empty. A jump from up to starved can be caused by either of the following two events (note that the "underlying machine" of departure server D_i is the same as the one of A_{i+1}, namely M_{i+1}):

(i) A_i is down or starved, D_i is up and draining the (non-empty) buffer B_i, and then, it happens that B_i becomes empty, and thus, A_{i+1} gets starved.

(ii) A_i and D_i are both up, and B_i is empty, and then, it happens that A_i goes down or starved, and thus, A_{i+1} gets starved.

The number of type-(i) jumps per time unit is equal to $(\bar{f}_{d,u}^{(i)}(0)+\bar{f}_{st,u}^{(i)}(0))s_D^{(i)}$. To obtain the rate at which type-(ii) jumps occur, note that the jth element of $\mathbf{P}_{u,u}^{(i)}(0)\mathbf{1}$ is the fraction of time that B_i is empty and D_i is in up-state j, and the jth element of $\begin{pmatrix} \mathbf{Q}_{A,(u,d)}^{(i)} & \mathbf{Q}_{A,(u,st)}^{(i)} \end{pmatrix}\mathbf{1}$ is the rate from up-state j to the set of down- and starved-states. Hence, the (inner) product of the two vectors yields the number of type-(ii) per time unit. The rate of type-(i) and type-(ii) events is also equal to $\lambda_{US}^{(i+1)}$ times the fraction of time D_i is generating output, which is $\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{d,u}^{(i)} - \bar{p}_{d,u}^{(i)}(0) + \bar{\pi}_{st,u}^{(i)} - \bar{p}_{st,u}^{(i)}(0)$. Hence,

$$\lambda_{US}^{(i+1)} = \frac{(\bar{f}_{d,u}^{(i)}(0) + \bar{f}_{st,u}^{(i)}(0))s_D^{(i)} + (\mathbf{P}_{u,u}^{(i)}(0)\mathbf{1})'\begin{pmatrix} \mathbf{Q}_{A,(u,d)}^{(i)} & \mathbf{Q}_{A,(u,st)}^{(i)} \end{pmatrix}\mathbf{1}}{\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{d,u}^{(i)} - \bar{p}_{d,u}^{(i)}(0) + \bar{\pi}_{st,u}^{(i)} - \bar{p}_{st,u}^{(i)}(0)}, \tag{9}$$

where $\mathbf{1}$ is a column vector of ones of appropriate size and $(\cdot)'$ is the transpose.

To determine the rate and cv of the starvation time $SU^{(i+1)}$, we first state the following result. Consider a transient Markov chain with k states, transition rate matrix $\mathbf{\Gamma}$ and entrance probability vector α, and let the random variable Y denote the time till exit. Then the nth moment of Y is given by (see, e.g., [16])

$$E(Y^n) = (-1)^n n! \alpha \mathbf{\Gamma}^{-n}\mathbf{1}, \tag{10}$$

from which the rate $\lambda_Y = 1/E(Y)$ and cv $c_Y = \sqrt{E(Y^2) - E^2(Y)}/E(Y)$ readily follow. The starvation time of A_{i+1} can be described as an exit time: the starvation time ends when A_i returns to up from either a down-state or starved-state. Thus, the starvation time $SU^{(i+1)}$ can be seen as the exit time of the Markov chain, the states of which are the down-states and starved-states of A_i, and with (transient) generator

$$\mathbf{\Gamma}_{SU}^{(i+1)} = \begin{pmatrix} \mathbf{Q}_{A,(d,d)}^{(i)} & 0 \\ 0 & \mathbf{Q}_{A,(st,st)}^{(i)} \end{pmatrix}.$$

Further, the entrance probability vector $\alpha_{SU}^{(i+1)}$ is given by

$$\alpha_{SU}^{(i+1)} = \xi_{SU}^{(i+1)} \left[\left(\begin{pmatrix} \mathbf{F}_{d,u}^{(i)}(0) \\ \mathbf{F}_{st,u}^{(i)}(0) \end{pmatrix} \mathbf{1} \right)' s_D^{(i)} + (\mathbf{P}_{u,u}^{(i)}(0)\mathbf{1})' \begin{pmatrix} \mathbf{Q}_{A,(u,d)}^{(i)} & \mathbf{Q}_{A,(u,st)}^{(i)} \end{pmatrix} \right], \tag{11}$$

where $\xi_{SU}^{(i+1)}$ is a normalization constant. Note the similarity between (11) and (9).

Next, we determine the speed $s_A^{(i+1)}$ at which A_{i+1} produces, whenever it is up. If the speed s_{i+1} of M_{i+1} is less or equal to the speed $s_A^{(i)}$ of A_i, then A_{i+1} can always produce at speed s_{i+1}, whenever up, so $s_A^{(i+1)} = s_{i+1}$. If $s_{i+1} > s_A^{(i)}$, then A_{i+1} has to lower its speed to $s_A^{(i)}$ when B_i is empty and A_i is up. The fraction of time, conditional on A_{i+1} (or equivalently, D_i) producing, is equal to $\bar{p}_{u,u}^{(i)}(0) / \left(\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{d,u}^{(i)} - \bar{p}_{d,u}^{(i)}(0) + \bar{\pi}_{st,u}^{(i)} - \bar{p}_{st,u}^{(i)}(0) \right)$. Hence,

$$s_A^{(i+1)} = \begin{cases} s_{i+1} & \text{if} \quad s_{i+1} \leq s_A^{(i)} \\ s_{i+1} - \dfrac{\bar{p}_{u,u}^{(i)}(0)}{\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{d,u}^{(i)} - \bar{p}_{d,u}^{(i)}(0) + \bar{\pi}_{st,u}^{(i)} - \bar{p}_{st,u}^{(i)}(0)} (s_{i+1} - s_A^{(i)}) & \text{if} \quad s_{i+1} > s_A^{(i)} \end{cases} \tag{12}$$

The expressions for the parameters of D_{i-1} are symmetrical to the ones for A_{i+1}. The rate at which D_{i-1} jumps from up to blocked is given by

$$\lambda_{UB}^{(i-1)} = \frac{(\bar{f}_{u,d}^{(i)}(b) + \bar{f}_{u,bl}^{(i)}(b)) s_A^{(i)} + \mathbf{1}' \mathbf{P}_{u,u}^{(i)}(b) \begin{pmatrix} \mathbf{Q}_{D,(u,d)}^{(i)} & \mathbf{Q}_{D,(u,bl)}^{(i)} \end{pmatrix} \mathbf{1}}{\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{u,d}^{(i)} - \bar{p}_{u,d}^{(i)}(b) + \bar{\pi}_{u,bl}^{(i)} - \bar{p}_{u,bl}^{(i)}(b)}$$

The rate $\lambda_{BU}^{(i-1)}$ and cv $c_{BU}^{(i-1)}$ of a blocking time $BU^{(i-1)}$ follow from (10) with

$$\mathbf{\Gamma}_{BU}^{(i-1)} = \begin{pmatrix} \mathbf{Q}_{D,(d,d)}^{(i)} & 0 \\ 0 & \mathbf{Q}_{D,(bl,bl)}^{(i)} \end{pmatrix}$$

and

$$\alpha_{BU}^{(i-1)} = \xi_{BU}^{(i-1)} \left[\mathbf{1}' \begin{pmatrix} \mathbf{F}_{u,d}^{(i)}(b) & \mathbf{F}_{u,bl}^{(i)}(b) \end{pmatrix} s_A^{(i)} + \mathbf{1}' \mathbf{P}_{u,u}^{(i)}(b) \begin{pmatrix} \mathbf{Q}_{D,(u,d)}^{(i)} & \mathbf{Q}_{D,(u,bl)}^{(i)} \end{pmatrix} \right].$$

Lastly, the average speed at which D_{i-1} produces in the up-states is given by

$$s_D^{(i-1)} = \begin{cases} s_i & \text{if} \quad s_i \leq s_D^{(i)} \\ s_i - \dfrac{\bar{p}_{u,u}^{(i)}(b)}{\bar{\pi}_{u,u}^{(i)} + \bar{\pi}_{u,d}^{(i)} - \bar{p}_{u,d}^{(i)}(b) + \bar{\pi}_{u,bl}^{(i)} - \bar{p}_{u,bl}^{(i)}(b)} (s_i - s_D^{(i)}) & \text{if} \quad s_i > s_D^{(i)} \end{cases} \tag{13}$$

5 Results and discussion

In this section we investigate the quality of the proposed method. We also compare the method to the one in [2]. This method assumes exponential up- and downtimes, thus neglecting possible non-exponential behavior, but it might be used as a first rough approximation. This section is divided into two parts. First, Subsection 5.1 shows the performance of the proposed method on a large test set. In Subsection 5.2 we apply our method to two case studies with data from practice.

5.1 Test set

We test the performance of our method on a large test set, in which we vary the values of seven input parameters: the number of machines in the line, mean uptimes, mean downtimes, squared coefficient of variation (scv) of uptimes, scv of downtimes, machine speed configuration, and buffer sizes. Table 1 lists the different settings for each input parameter. By making all combinations, we obtain a test set of $4 \times 4 \times 8 \times 4 \times 8 \times 3 \times 4 = 49,152$ cases. As shown in Table 1, cases are included with imbalance in mean up- and downtimes, and scv of up- and downtimes. We choose three different machine speed configurations, the first being a homogeneous speed configuration, in which each machine has the same speed. Secondly, we have a jumping speed configuration, in which each odd machine has a speed of 10 and each even machine has a speed of 15. The last setting, a V-shape speed configuration, is motivated by practice. In this setting, the first and last machine are the fastest, speeds decrease linearly in the first part of the production line, and speeds increase linearly in the second part of the production line. For instance, in a 6-machine production line, the machine speeds would be $\{15, 12.5, 10, 10, 12.5, 15\}$.

Table 1: Input parameter values of the test set

Input parameter	**Values**
Number of machines	4, 8, 12, 16
Mean uptimes	{10,10,10,10,...}, {10,5,10,5,...}, {20,20,20,20,...}, {20,10,20,10,...}
Mean downtimes	{1,1,1,1,...}, {1,0.5,1,0.5,...}, {2,2,2,2,...}, {2,1,2,1,...}
Scv of uptimes	{0.5,0.5,0.5,0.5,...}, {0.5,1,0.5,1,...}, {1,1,1,1,...}, {1,2,1,2,...}, {2,2,2,2,...}, {2,4,2,4,...}, {4,4,4,4,...}, {4,8,4,8,...}
Scv of downtimes	{0.5,0.5,0.5,0.5,...}, {0.5,1,0.5,1,...}, {1,1,1,1,...}, {1,2,1,2,...}, {2,2,2,2,...}, {2,4,2,4,...}, {4,4,4,4,...}, {4,8,4,8,...}
Machine speeds	{10,10,10,10,...}, {10,15,10,15,...}, {15,...,10,...,15}
Buffer size	{1,1,1,1,...}, {10,10,10,10,...}, {25,25,25,25,...}, {50,50,50,50,...}

We test the performance of our method by comparing the estimated throughput and mean total buffer content to the same quantities obtained from a discrete-event simulation model. The 95% confidence intervals in the simulation have a width of at most 0.5%. In Tables 2-8, we show the average relative errors of our approximation (column "Model PHT") and the one in [2] (column "Model EXP"). Each table row provides the average relative errors over all cases in which the input parameters have the values as specified in the first column. For instance, the first row of Table 2 gives the average errors over all 12,288 cases with four machines in the production line.

In Table 2 we see that our method produces reliable throughput estimates for production lines of up to 16 machines. The error in mean buffer content even decreases for longer production lines. In Tables 3 and 4, we see that the method performs slightly worse for lines with small mean uptimes and larg mean downtimes. This may be due to the fact that starvation and blocking are more prominent when downtimes are large compared to uptimes. Table 5 and 6 show that our method performs well for both small and large the scv of up- and downtimes, a feature which is definitely not shared by the method in [2]. In Table 7 we see that our method performs excellent for cases with a V-shape speed configuration, but slightly less for cases with a "jumping" speed configuration. Finally, Table 8 shows that our method performs well for cases with very small buffers up to large buffers.

Table 2: Results for production lines with different lengths

Line length	**Error (%) in the throughput**		**Error (%) in mean buffer content**	
	Model PHT	**Model EXP**	**Model PHT**	**Model EXP**
4	0.57	2.00	1.30	1.65
8	1.03	4.37	0.53	0.80
12	1.65	6.30	0.36	0.61
16	2.18	7.91	0.29	0.55

Table 3: Results for production lines with different mean uptimes

Mean uptimes	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
10,10,10,10,...	1.56	5.52	0.56	0.77
10,5,10,5,...	1.64	7.17	0.65	0.95
20,20,20,20,...	1.05	3.35	0.58	0.85
20,10,20,10,...	1.18	4.53	0.67	1.04

Table 4: Results for production lines with different mean downtimes

Mean downtimes	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
1,1,1,1,...	1.18	4.55	0.71	1.04
1,0.5,1,0.5,...	1.08	3.22	0.61	1.12
2,2,2,2,...	1.62	7.15	0.61	0.66
2,1,2,1,...	1.55	5.65	0.54	0.80

Table 5: Results for production lines with different scv of uptimes

Scv of uptimes	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
0.5,0.5,0.5,0.5,...	1.09	4.06	0.56	0.89
0.5,1,0.5,1,...	1.12	4.23	0.52	1.01
1,1,1,1,...	1.36	4.59	0.52	0.78
1,2,1,2,...	1.29	4.83	0.59	0.93
2,2,2,2,...	1.44	5.26	0.59	0.77
2,4,2,4,...	1.37	5.62	0.67	0.97
4,4,4,4,...	1.62	6.12	0.71	0.85
4,8,4,8,...	1.57	6.45	0.76	1.03

Table 6: Results for production lines with different scv of downtimes

Scv of downtimes	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
0.5,0.5,0.5,0.5,...	1.32	1.51	0.55	0.78
0.5,1,0.5,1,...	1.37	1.63	0.52	0.77
1,1,1,1,...	1.42	2.32	0.46	0.50
1,2,1,2,...	1.49	3.86	0.57	0.92
2,2,2,2,...	1.30	5.13	0.51	0.66
2,4,2,4,...	1.28	6.92	0.73	1.18
4,4,4,4,...	1.28	8.81	0.67	1.03
4,8,4,8,...	1.38	10.98	0.92	1.39

Table 7: Results for production lines with different machine speeds

Machine speeds	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
10,10,10,10,...	1.16	5.19	0.34	0.61
10,15,10,15,...	2.18	6.32	1.11	1.32
15,...,10,...,15	0.73	3.93	0.40	0.78

Table 8: Results for production lines with different buffer sizes

Buffer sizes	Error (%) in the throughput		Error (%) in avg buffer content	
	Model PHT	Model EXP	Model PHT	Model EXP
1	0.47	1.51	0.27	0.39
10	1.65	6.43	0.62	0.68
25	1.72	6.88	0.75	1.02
50	1.58	5.76	0.82	1.51

5.2 Practical cases

In this section we apply the proposed method to two practical cases. The first case is a production line at Heineken brewery in 's-Hertogenbosch, the Netherlands, where retour bottles are being re-used and refilled. Bottles enter the production line in crates on pallets and subsequently go through the (1) depallatizer, (2) logo-detection, (3) depacker, (4) bottle-washer, (5) EBI (Empty Bottle Inspector), (6) filler, (7) pasteurizer, (8) labeler, (9) packer, (10) cratemanco, and (11) palletizer. Since the number of bottles going through the machines per hour is very large, we approximate this flow as a fluid. In Table 9, all the input parameter values are listed for this production line. The mean and scv of up- and downtimes are retrieved directly from industrial data. Typical for this production line are the high variations in uptimes and (especially) in downtimes. Therefore, it is necessary to have an approximation method that properly takes into account this variation. Furthermore, the setup of machine speeds corresponds to the V-shape setup, as we have seen in the previous subsection. The buffer sizes reflect the maximum number of bottles on the conveyor belt in between two successive machines. In [3] the modeling of the bottling line is described in detail, and the resulting fluid flow model is extensively validated by means of industrial data. The purpose of this section is to show that the current method accurately predicts the throughput of the fluid flow model, the parameter values of which are based on the Heineken data.

Table 9: Data for the Heineken production line

Machine name	Mean uptime (hrs)	Scv of uptime	Mean downtime (hrs)	Scv of downtimes	Machine speed	Buffer size
Depalletizer	1.37	2.37	0.060	16.32	48349	3647
Logo-detection	0.58	2.16	0.026	38.15	43284	1823
Depacker	0.14	2.82	0.028	10.22	43284	6895
Bottle-washer	0.32	1.86	0.047	34.72	40389	5300
EBI	0.58	3.09	0.034	36.54	37407	270
Filler	0.42	2.65	0.036	18.99	37407	4874
Pasteurizer	3.94	6.31	0.081	3.30	40170	7014
Labeler	0.29	2.13	0.025	19.12	37094	6622
Packer	0.27	2.87	0.035	16.95	40988	4630
Cratemanco	2.82	4.07	0.152	3.27	41500	6945
Palletizer	1.96	2.34	0.069	25.45	42559	-

A discrete-event simulation using *empirical* up- and downtime data predicts a throughput of 27,735 bottles per hour. The method in [2], assuming exponential up- and downtimes provides a throughput estimate of 31,979 bottles per hour, which is an overestimate of 15.30% compared to simulation. Hence, neglecting the high variations in up- and downtimes leads to a poor estimate. The method in this paper gives a much better throughput estimate of 26,679 bottler per hour, which is an underestimate of 3.81%.

The second practical case is an assembly line of small electronic components in Guangdong, China. A tape (or lead frame) flows through eight machines at a high speed. At each machine small components (dies and wire bonds) are placed on the lead frame. If one of the machines goes down, the upstream machine can still produce until the length of the lead frame in between the two machines has reached a maximum level. The number of components that fit on this maximum length of lead frame can be seen as the maximum size of the buffer. Again, the processing speeds are high, so that we can treat the flow of products as being a fluid. The mean and scv of up- and downtimes are retrieved from actual machine data, showing high variability in up- and downtimes. The speeds of the eight machines only differ slightly, making the speed configuration almost homogeneous. The input parameter values for this case can be found in Table 10. The modeling of the assembly line as fluid flow model and the validation of this model are described in detail in [9].

By using discrete-event simulation with empirical up- and downtime data, we obtain a throughput estimate of 12,279 products per hour. The approximation of [2] gives a throughput estimate of 13,882 products per hour, which is an overestimation of 13.05%. The current approximation provides a much better throughput estimate of 12,029 products per hour, which is an underestimation of 2.04%. This result again indicates that the non-exponential character of up- and downtimes should be taken into account

Table 10: Data for the NXP production line

Machine name	Mean uptime (hrs)	C^2 of uptime	Mean downtime (hrs)	C^2 of downtimes	Machine speed	Buffer size
A1	0.36	2.30	0.054	8.45	17225	2250
A2	0.45	2.10	0.062	22.40	17270	2250
A3	0.51	2.03	0.048	11.60	16885	2250
A4	0.43	1.74	0.044	14.27	16795	2250
P1	0.77	2.14	0.064	5.80	16446	2250
P2	0.76	2.45	0.059	4.57	16559	5750
P3	0.54	2.56	0.059	5.06	16708	5750
P4	0.43	2.61	0.050	5.61	16785	-

to be able to produce accurate performance estimates.

6 Conclusions

In this paper, we construct an approximative method to analyze production lines with continuous flows. The distinguishing feature of this method is that it takes into account the generality of up- and downtime distributions without experiencing a state space collapse for longer production lines. The sophisticated use of aggregation techniques is crucial in our method. We find a strong performance of our method on a large test set and on two practical cases. The performance of our method is almost insensitive to the coefficient of variation in up- and downtimes, proving that our method takes into account the variations in up- and downtimes in a correct way.

The following limitation should be stated about this model. For cases with low coefficients of variation of up- and downtimes, the method might become slower, since we need phase-type distributions with more exponential phases for these cases. Further research might include the search for a state space reduction for these cases. However, when the coefficients of variation of up- and downtimes become very low, it might be better to treat the downs as planned downtimes, such that the production line can be analyzed using easier techniques. Further research can also include the adaption of this model to more complicated production lines, such as lines with multi-server workstations, assembly lines or feed-forward networks.

References

[1] I.J.B.F. Adan, V.G. Kulkarni, J.A.C. Resing (2003) Stochastic discretization for the long-run average reward in fluid models. *PEIS 17*, 251-265.

[2] R. Bierbooms, I.J.B.F. Adan, M. van Vuuren (2010) Performance analysis of production lines with continuous material flows and finite buffers. *EURANDOM report 2010-044*, Eindhoven University of Technology.

[3] R. Bierbooms, I.J.B.F. Adan (2011) Modeling and performance analysis of a bottling line; a case study. In preparartion.

[4] M.H. Burman (1995) New results in flow line analysis. *Ph.D. thesis*, Massachusetts Institute of Technology.

[5] J. A. Buzacott, J.G. Shanthikumar (1993) Stochastic Models of Manufacturing Systems. Prentice Hall.

[6] Y. Dallery, R. David, X.L. Xie (1988) An efficient algorithm for analysis of transfer lines with unreliable machines and random processing times *IIE Transactions 20 (3)*, 280-283.

[7] Y. Dallery, R. David, X.L. Xie (1989) Approximate analysis of transfer lines with unreliable machines and finite buffers *IEEE Transactions on Automatic Control 34 (9)*, 943-953.

[8] Y. Dallery and H. Le Bihan (1997) Homogenization techniques for the analysis of production lines with unreliable machines having different speeds. *European Journal of Control 3 (3)*, 200-215.

[9] G. Kesuma, R. Martono (2010) Dynamic line modeling. *Mathematics for Industry report*, Eindhoven University of Technology.

[10] M. Di Mascolo (1988) Méthode analytique d'évaluation des performances d'une ligne d'assemblage. *Rapport de DEA*, Laboratoire d'Automatique de Grenoble.

[11] S.B. Gershwin and I.C. Schick (1980) Continuous model of an unreliable two-machine material flow system with a finite interstage buffer. *Report LIDS-R-1039*, Massachussets Institute of Technology.

[12] S.B. Gershwin (1987) An efficient decomposition algorithm for the approximate evaluation of tandem queues with finite storage space and blocking *Operations Research 35 (2)*, 291-305.

[13] S.B. Gershwin (1987) Representation analysis of transfer lines with machines that have different processing rates. *Annals of Operations Research 9*, 511-530.

[14] M.A. Johnson (1993) An empirical study of queueing approximations based on phase-type distributions. *Commun. Statist.-Stochastic Models 9*, 531-561.

[15] M.B.M. de Koster (1987) Estimation of line efficiency by aggregation. *International Journal of Production Research 25 (4)*, 615-626.

[16] G. Latouche and V. Ramaswami (1999), Introduction to matrix-analytic methods in stochastic modeling. *ASA-SIAM Series on Statistics and Applied Probability*, Philadelphia.

[17] R. Levantesi, A. Matta, and T. Tolio (2003) Performance evaluation of continuous production lines with machines having different processing times and multiple failure modes. *Performance Evaluation 51*, 247-268.

[18] X.G. Liu and J.A. Buzacott (1990) Approximate models of assembly systems with finite banks. *European Journal of Operational Research 45*, 143-154.

[19] D. Mitra (1988) Stochastic theory of a fluid flow model of multiple failure-susceptible producers and consumers coupled by a buffer. *Advances in Applied Probability 20*, 646-676.

[20] V. Ramaswami (1999) Matrix analytic methods for stochastic fluid flows. *Teletraffic Engineering in a Competitive World (Proceedings of the 16th International Teletraffic Congress* Elsevier Science B.V., Edinburgh, UK, 1019-1023.

[21] A. da Silva Soares and G. Latouche (2006) Matrix-analytic methods for fluid queues with finite buffers. *Performance Evaluation 63*, 295-314.

[22] A. da Silva Soares and G. Latouche (2005) A matrix-analytic approach to fluid queues with feedback control. *International Journal of Simulation: Systems, Science and Technology 6 (1-2)*, 4-12.

[23] B. Tan and S.B. Gershwin (2009) Analysis of a general Markovian two-stage continuous-flow production system with a finite buffer. *International Journal of Production Economics 120*, 327-339.

[24] H.C. Tijms (1994) Stochastic models: an algorithmic approach. *John Wiley & Sons*, Chichester.

[25] J. Wijngaard (1979) The effect of interstage buffer storage on the output of two unreliable production units in series with different production rates. *AIIE Transactions 11 (1)*, 42-47.