TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Performance analysis of production lines with continuous material flows and finite buffers

Citation for published version (APA):
Bierbooms, R., Adan, I. J. B. F., & Vuuren, van, M. (2010). *Performance analysis of production lines with continuous material flows and finite buffers*. (Report Eurandom; Vol. 2010044). Eurandom.

Document status and date:
Published: 01/01/2010

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Download date: 04. Oct. 2023

EURANDOM PREPRINT SERIES
2010-044

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

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

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, i.j.b.f.adan@tue.nl, vanvuuren@cqm.nl

Abstract: *This paper deals with the approximative analysis of production lines with continuous material flow consisting of a number of machines or servers in series and finite buffers in between. Each server suffers from operational dependent breakdowns, characterized by exponentially distributed up- and down-times. We construct an iterative method to efficiently and accurately estimate performance characteristics such as throughput and mean total buffer content. The method is based on decomposition of the production line into single-buffer subsystems. Novel features of the method are (i) modeling of the aggregate servers in each subsystem, (ii) equations to iteratively determine the processing behavior of these servers, and (iii) use of modern matrix-analytic techniques to analyze each subsystem. The proposed method performs very well on a large test set, including long and imbalanced production lines. For production lines with imbalance in mean down-times, we show that a more refined modeling of the servers in each subsystem performs significantly better. Lastly, we apply the iterative method to predict the throughput of a bottle line at brewery Heineken Den Bosch yielding errors of less than two percent.*

Keywords: *production line, finite buffer, fluid flow, approximation, decomposition*

1 Introduction

In this paper we analyze production lines with continuous material flow consisting of a number of machines or servers in series and a finite buffer between each of the servers. We are interested in both throughput and buffer content distributions. Figure 1 illustrates a production line of length four, where M_i denotes the ith server and B_i denotes the ith buffer. Each server M_i has a maximum production rate and is subject to breakdowns, which are characterized by an up- and down cycle. First, the server is "up" for an exponentially distributed amount of time, after which it is "down" for an exponentially distributed amount of time. During an uptime the server is able to produce; during a downtime it is not. Each buffer has a finite capacity. When buffer B_i becomes full, server M_i will slow down its production rate to that of server M_{i+1}; similarly, when buffer B_{i-1} becomes empty, server M_i will slow down to the production rate of server M_{i-1}. In this system, blocking and starvation play significant roles: a server can become blocked when the downstream buffer is full, or it can become starved when the upstream buffer is empty. The servers can only break down when they are actually producing; these are called operationally dependent failures (ODFs). We assume that breakdown rates are constant and independent of the production rates. This assumption is not common in the literature on production lines with ODFs, typically assuming a (linear) relation between production and breakdown rates. However, in the practical case described below, there appeared to be no evidence for dependencies between production and breakdown rates.

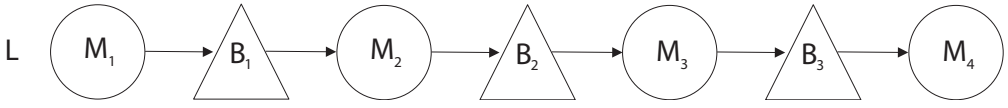

Figure 1: A production line L with four servers, labeled M_1 up to M_4

Our research is inspired by a production line at brewery Heineken Den Bosch, where retour bottles are being filled and processed by eleven machines in series having different machine speeds. Each machine

has its own up- and downtime distributions. A conveyor belt between each pair of machines is being used for transportation, but also for buffering in case of machine breakdowns. The number of bottles handled per hour by the machines is very large, which makes it natural to treat the material flows as fluid rather than as discrete items.

There are many papers considering production lines with fluid flows. Production lines with two machines (or two production stages) and one buffer can usually be analyzed exactly by modeling the system as a Markov chain. The dynamics of this Markov chain are described by a set of linear differential equations, see, e.g., [7, 19] for both production stages having two states (up and down) and [9, 14] for both stages having more than two states. Typically, a matrix exponential function or spectral analysis is used to solve the set of differential equations. However, since these procedures can lead to numerical problems in case of a large buffer, a more stable approach is needed. Adan et al. [1] and Ramaswami [15] found that there is a connection between production lines with fluid flows and discrete-state quasi-birth-and-death-processes (QBDs). Soares and Latouche [16] propose a numerically reliable method for the analysis of a single-buffer fluid flow model with time-dependent failures (TDFs), where a mixture of two matrix-exponentials is used to express the buffer content distribution. They employ matrix-analytic methods, typically used for discrete-state QBDs. Their method is extended in [17] to the case of ODFs. This matrix-analytic method also works well in case of a large buffer. In Section 2, we elaborate on the analysis of two-stage production lines with ODFs.

Production lines with more than two stages and exponential up- and down-times cannot be analyzed exactly, justifying the development of approximation methods. De Koster [10] suggests an aggregation method, where repeatedly two stages are aggregated into one stage, until a system of two stages and one buffer remains to be analyzed. The performance estimates are good for balanced lines; however for non-balanced lines the estimation error increases rapidly. Gershwin [8] introduces a decomposition algorithm for production lines with machines that take an equal amount of time to process parts; this idea is exploited by Dallery et al. [3] in the DDX algorithm. Burman [2] extends this algorithm to production lines where machines have different production rates. However, convergence problems may arise in this case, requiring (ad-hoc) modifications to secure convergence. Levantesi et al. [12] propose a more detailed decomposition method, where the upstream machine of each subsystem has a "starved-state" for each upstream subsystem and the downstream machine has a "blocked-state" for each downstream subsystem. Also, multiple failure states can be modeled in their approach. Although their method performs well, the state space of the Markov chain will collapse for longer production lines.

An alternative method is the use of homogenization techniques, see, e.g., [4, 6, 13]. In these papers, a non-homogeneous production line is replaced by an equivalent homogeneous line, in which all machines have the same processing times. The homogeneous line can be analyzed by, for example, the DDX algorithm. Dallery and Le Bihan [5] propose an algorithm in which several homogenization techniques are combined.

In Section 3 of this paper, we develop a new approach to approximate the performance of production lines with continuous materials. The approach is based on decomposition of the production line into single-buffer subsystems consisting of an "arrival server" describing the upstream part of the production line and a "departure server" describing the downstream part. The upstream and downstream parts of the production lines are aggregated into single servers to be able to analyze long production lines. We will propose two models for the servers, that differ in the level of aggregating the behavior of the upstream and downstream parts of the production line. The first and simplest model describes the arrival and departure server as a two-state Markov chain with an up and down state, where blocking and starvation are aggregated into the downtime. The novelty of this model lies in the determination of the transition rates and average speeds. In the second and more detailed model, blocking and starvation are described as a separate state, different from a "real" breakdown, so in this case we have a three-state Markov chain for the arrival and departure server. The rates of the Markov chains are determined iteratively. In our approach, we avoid numerical instability problems inherent to the use of matrix exponential functions or

spectral analysis, by adopting modern matrix-analytic methods as described in Section 2. Further, it will appear that no convergence problems occur in the iterative algorithm, and that conservation of flow is satisfied automatically without the need to explicitly add this law as extra constraints.

In Section 4, we discuss the performance of approximation method on a large test set of 1728 cases, containing long production lines and imbalance in all parameters. We show that the method performs well on this test set, with an average error in throughput over the whole test set of 1.7% and 1.5% in case the first, respectively second model is used. The second model performs significantly better than the first model for production lines with different repair rates, which justifies modeling starvation of the arrival machine and blocking of the departure machine as separate states. Furthermore, a practical case at Heineken Den Bosch is discussed in Section 4. For their 11-machine production line, the error of the throughput predicted by the approximation method, using the second model, is 1.44 $\%$. Section 5 discusses the results of the test bed. Finally, Section 6 contains conclusions and directions for further research.

2 Two-stage production lines with ODFs

This section treats the analysis of two-stage production lines with ODFs, for which exact methods are available. These methods form the basis of approximation methods for multi-stage production lines as analyzed in the next section. The system consists of an arrival server M_A, a departure server M_D, and a buffer B of size b in between; see Figure 2.

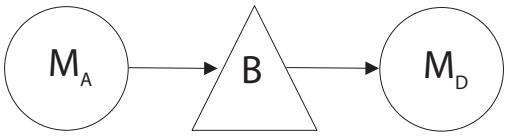

Figure 2: Two-stage production line with servers M_A and M_D, and buffer B in between

The phase process of each server is driven by a continuous-time Markov chain. For the arrival server, we are given a finite set of states S_A of size $N_A = |S_A|$ with transition matrix Q_A. In state $i_A \in S_A$, the arrival server provides the buffer with fluid at a speed of r_{A,i_A} per time unit. The finite set of states for the departure server is given by S_D of size $N_D = |S_D|$ with transition matrix Q_D. The departure server depletes the fluid buffer at a speed of r_{D,i_D} in state $i_D \in S_D$. The phase process of the two-stage system is given by $S = S_A \cup S_D$ with a total number of $N = N_A \times N_D$ states. To determine the generator of the phase process, we use the Kronecker product: when A is an $n_1 \times n_2$ matrix and B is an $n_3 \times n_4$ matrix, the Kronecker product $A \otimes B$ 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 Q of the phase process and *net speed* vector r are given by

$$Q = Q_A \otimes I_{N_D} + I_{N_A} \otimes Q_D,$$

$$r = r_A \otimes \mathbf{1}_{N_D} - \mathbf{1}_{N_A} \otimes r_D,$$

where I_n is an identity matrix of size $n \times n$ and $\mathbf{1}_n$ is a column vector of ones of size n.

We assume operationally dependent failures (ODFs), meaning that the arrival server cannot break down when it is not producing because of blocking, and the departure server cannot break down when it is not producing because of starvation. For the arrival server this implies that whenever the buffer is full and

the departure machine is in a state with zero-speed, the arrival machine is blocked and it cannot jump to a state with zero-speed. Formally we define a full-buffer process, with generator Q^F given by

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

Similarly, when the arrival server is in a state with zero-speed and the buffer is empty, the departure server is starved and it cannot jump to a state with zero-speed. The generator Q^E of the empty-buffer process is defined as

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

The two-stage production system can be described by a Markov process with states (i, x) where i is the state of the phase process and x is the fluid level of the buffer. We are interested in the steady state distribution of this Markov process. We define $F_i(x)$ as the steady state probability that the system is in state $i \in S$ and the buffer content is less or equal to x. This probability can be related to its derivative (or density) $\frac{dF_i(x)}{dx}$ and probabilities $F_j(x)$, $j \neq i$ in the following way,

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

This relation is based on a balance principle, stating that the probability flux into the set of states $\{(i, y), 0 \leq y \leq x\}$ is equal to the flux out of that set. Rearranging terms gives

$$r_i \frac{dF_i(x)}{dx} = \sum_{j\in S} Q_{j,i} F_j(x).$$

Defining $F(x)$ as the column vector with elements $F_i(x)$ and R as the diagonal matrix with elements $R_{i,i} = r_i$, we can write

$$\frac{dF(x)}{dx} = ZF(x), \tag{1}$$

where $Z = R^{-1}Q^T$. Here we assume that none of the rates r_i are equal to 0, so R is invertible (see also Remark 1). By differentiating (1) and introducing the probability density vector $f(x) = \frac{dF(x)}{dx}$, we get

$$\frac{df(x)}{dx} = Zf(x).$$

The solution to these linear differential equations is a matrix-exponential density function of the form

$$f(x) = ce^{Zx}, \tag{2}$$

where c is a vector of constants. Note that we have probability mass at the boundaries $x = 0$ and $x = b$. It is convenient to define $p_i^{(0)}$ as the probability that the system is in state $i \in S$ and the buffer is empty; $p_i^{(b)}$ is the probability that the system is in state $i \in S$ and the buffer is full. So

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

We can determine the probabilities $p_i^{(0)}$ and $p_i^{(b)}$, and the constants c_i, $i \in S$ in (2) by solving a set of boundary equations plus the normalizing equation. The boundary equations are determined by applying the balance principle to the boundary levels $x = 0$ and $x = b$, yielding

$$\sum_{j\in S} Q_E(j, i) p_j^{(0)} - r_i f_i(0) = 0, \qquad i \in S, \tag{3}$$

$$\sum_{j\in S} Q_F(j,i)p_j^{(b)} + r_i f_i(b) = 0, \qquad i \in S, \tag{4}$$

$$\sum_{i\in S} F_i(b) = 1. \tag{5}$$

Note that $F_i(b)$ can be expressed in terms of the density and boundary probabilities,

$$F_i(b) = p_i^{(0)} + \int_0^b f_i(x)dx + p_i^{(b)}.$$

We divide the state space in two parts, S_+ being the set of states with positive net speeds and S_- being the set of states with negative net speeds. Then $p_i^{(0)} = 0$ if $i \in S^+$ because the buffer immediately starts to fill in a state with positive net speed. Similarly, $p_i^{(b)} = 0$ if $i \in S^-$ since the buffer immediately empties in a state with negative net speed. This leaves us with $2N$ unknown quantities in the set of equations (3)-(5).

Remark 1 If there are $i \in S$ with $r_i = 0$, then the matrix R is a singular matrix. So we have to treat these states differently. Note that, for states i with $r_i = 0$, equation (1) reduces to

$$\sum_{j\neq i} Q_{i,j}F_i(x) = \sum_{j\neq i} Q_{j,i}F_j(x). \tag{6}$$

These equations define $F_i(x)$ in terms of the probabilities $F_j(x)$ with $r_j \neq 0$. Now we use these equations to eliminate the probabilities $F_i(x)$ and we adjust the generators Q, Q^E, and Q^F accordingly. Once the steady state distribution for states with non-zero speeds is determined, we obtain the steady state probabilities $F_i(x)$ with $r_i = 0$ from the balance equations (6).

The solution as described above suffers from numerical instability, since e^{Zx} becomes very large whenever b is large and Z has (large) positive eigenvalues. Scaling with a factor e^{-Zb} would solve this problem, however, then the negative eigenvalues cause similar problems. The use of spectral analysis to solve this problem can be very time consuming (and also numerically unstable), since Z may have defective eigenvalues.

A solution to this numerical problem is to split the state space in two parts and to analyse the Markov process restricted to one of these parts; one part consisting of all states with positive net speeds and the other part consisting of all states with negative net speeds, each part having its own transition rate matrix. The corresponding matrices Z_1 and Z_2 in the matrix-exponential density functions both have only negative eigenvalues, which solves the numerical problem. To analyze the Markov process restricted to one part, we need probabilities of returning to a certain fluid level in a particular state. Such return probabilities can be determined by using matrix-analytic methods, as for discrete-state Quasi-Birth-and-Death processes. In Appendix A, we explain in detail how to obtain the density function and performance measures of the two-stage system, using matrix-analytic methods as in the analysis of Soares and Latouche [17].

3 Decomposition method for N-stage production lines

We consider a production line L consisting of N servers $M_1, ..., M_N$ and $N-1$ buffers $B_1, ..., B_{N-1}$ as illustrated in Figure 1. We define the maximum speed of server M_i as v_i; the up- and downrates are given as λ_i and μ_i respectively. The size of buffer B_i is b_i.

Since this production line cannot be analyzed exactly, we develop a method to approximate the throughput and buffer content distribution. We decompose production line L into two-stage subsystems, each subsystem consisting of an arrival server A_i, a departure server D_i, and buffer B_i in between. The decomposition is illustrated in Figure 2. In the description of the (phase) behavior of the arrival server we

include possible starvation caused by the upstream part of the production line. Similarly, in the description of the behavior of the departure server we include possible blocking caused by the downstream part of the production line. The generator corresponding to the phase process for arrival server A_i of the ith subsystem is defined as $Q_A^{(i)}$. The vector of speeds is defined as $r_A^{(i)}$, where the jth element of this vector is the speed in state j. For the departure server, we define generator $Q_D^{(i)}$ and speed vector $r_D^{(i)}$.

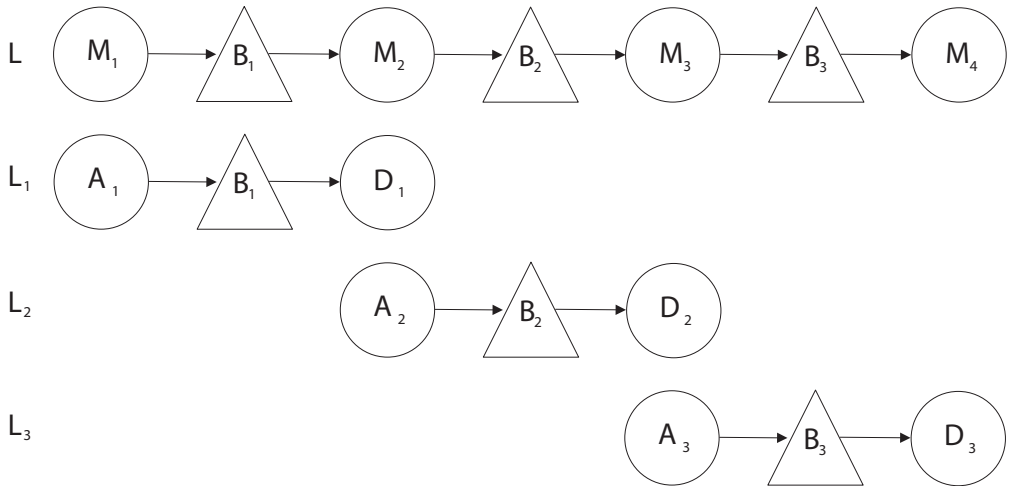

Figure 3: Decomposition of a production line L in three two-stage subsystems L_1, L_2 and L_3

The challenge is to iteratively determine the elements of $Q_A^{(i)}$, $r_A^{(i)}$, $Q_D^{(i)}$, and $r_D^{(i)}$. In the following subsections, two approaches are described. Firstly, Subsection 3.1 uses two-state Markov chains for A_i and D_i. In this case we have an "up"state and a "down"state; starvation and blocking are aggregated into the downstate. In Subsection 3.2, we treat starvation and blocking as separate states, which means that we have three-state Markov chains. Subsection 3.3 proposes an iterative algorithm to obtain the elements of $Q_A^{(i)}$, $r_A^{(i)}$, $Q_D^{(i)}$, and $r_D^{(i)}$.

3.1 Subsystem analysis using two-state Markov processes

In this subsection, we model the behavior of A_i and D_i of subsystem L_i, $i = 1, 2, \ldots, N-1$, as two-state Markov chains, the states and transitions of which are illustrated in Figure 4. Starvation of the arrival server and blocking of the departure server are included in the downstate of that server. In the remainder of this subsection we refer to this aggregated state as downstate.

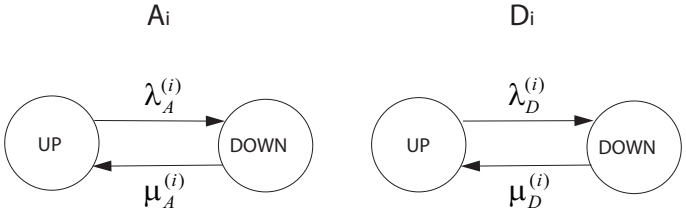

Figure 4: Two-state Markov chains describing the states and transitions of the arrival and departure server

For the arrival server, we define the generator $Q_A^{(i)}$ and speed vector $r_A^{(i)}$ as

$$Q_A^{(i)} = \begin{pmatrix} -\lambda_A^{(i)} & \lambda_A^{(i)} \\ \mu_A^{(i)} & -\mu_A^{(i)} \end{pmatrix}, \qquad r_A^{(i)} = \begin{pmatrix} v_A^{(i)} \\ 0 \end{pmatrix},$$

where $\lambda_A^{(i)}$ is the estimated rate of jumps from up to down, $\mu_A^{(i)}$ is the estimated rate of jumps from down to up, and $v_A^{(i)}$ is the average speed in the upstate. Note that this average speed is not necessarily equal to the maximum speed v_i, since it can be the case that server A_i has to adjust its speed sometimes because of a slower upstream server and an empty buffer.

The generator $Q_D^{(i)}$ and speed vector $r_D^{(i)}$ of the departure machine are defined as

$$Q_D^{(i)} = \begin{pmatrix} -\lambda_D^{(i)} & \lambda_D^{(i)} \\ \mu_D^{(i)} & -\mu_D^{(i)} \end{pmatrix}, \qquad r_D^{(i)} = \begin{pmatrix} v_D^{(i)} \\ 0 \end{pmatrix},$$

where $\lambda_D^{(i)}$, $\mu_D^{(i)}$, and $v_D^{(i)}$ have a similar interpretation as $\lambda_A^{(i)}$, $\mu_A^{(i)}$, and $v_A^{(i)}$ for the arrival server. We first derive the parameters for the arrival server, after which the departure server parameters will be treated.

To determine the parameters for the arrival server A_i, we use information from the upstream subsystem, assuming that we have this information available. Note that arrival server A_i of the subsystem of interest corresponds to departure server D_{i-1} of the upstream subsystem. We obtain the rates for A_i by linking these two together: we consider the arrival server as being up whenever the departure server of the upstream subsystem is producing and down whenever it is not producing. We use the following information from the upstream subsystem:

- $\pi_{j,j'}^{(i-1)}$: the probability that A_{i-1} is in state j and D_{i-1} is in state j' ($j, j' \in \{u, d\}$, where u=up and d=down).
- $p_{j,j'}^{(i-1)}(0)$: the probability that A_{i-1} is in state j, D_{i-1} is in state j' and B_{i-1} is empty.
- $f_{j,j'}^{(i-1)}(0)$: the density of the upstream subsystem's buffer content, for A_{i-1} being in state j, D_{i-1} being in state j' and B_{i-1} being empty.
- $Q_A^{(i-1)}(j, j')$: the rate of jumps from state j to state j' of A_{i-1}.

We can distinguish three different ways in which the state of arrival server A_i can jump from up to down. In the first situation, the down is caused by an actual breakdown of the arrival server; in the second and third situation the down is caused by a down of arrival server A_{i-1} and an empty buffer B_{i-1}. Figure 5 illustrates the two possible ways for the arrival server to get starved by the upstream subsystem.

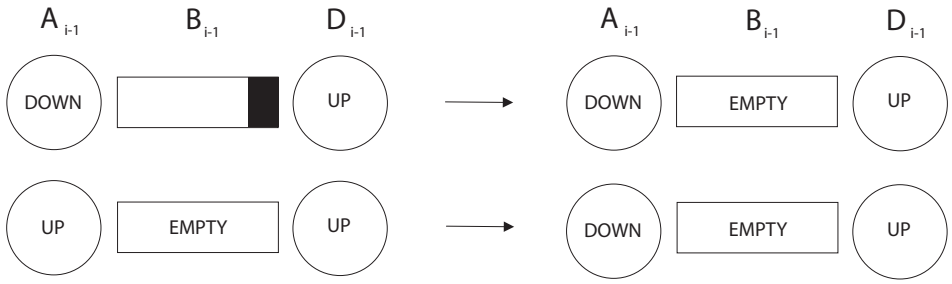

Figure 5: Two possible events in subsystem L_{i-1} causing A_i to get starved

We divide the rate $\lambda_A^{(i)}$ into three parts:

$$\lambda_A^{(i)} = \lambda_{A,1}^{(i)} + \lambda_{A,2}^{(i)} + \lambda_{A,3}^{(i)}.$$

The first part corresponds to an actual breakdown of the arrival server, which occurs at a rate of λ_i, so

$$\lambda_{A,1}^{(i)} = \lambda_i.$$

Secondly, A_i can go down when A_{i-1} is down and buffer B_{i-1} empties. The number of jumps per time unit of this type is obtained from the upstream subsystem as $f_{d,u}^{(i-1)}(0)$. This quantity is multiplied by the speed $v_D^{(i-1)}$ at which D_{i-1} empties buffer B_{i-1}. We have to condition on the probability that the upstream departure server is able to produce, which is given by $\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0)$. The expression for $\lambda_{A,2}^{(i)}$ becomes

$$\lambda_{A,2}^{(i)} = \frac{f_{d,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0)} v_D^{(i-1)}.$$

The last type of jump from up to down occurs when the usptream buffer is empty and the upstream arrival server is up, which is the case with probability $p_{u,u}^{(i-1)}(0)$. If in this case the upstream arrival server goes down, which happens with rate $Q_A^{(i-1)}(u,d)$, the arrival server of interest also goes down. Again we condition on the probability that the upstream departure server is able to produce, yielding

$$\lambda_{A,3}^{(i)} = \frac{p_{u,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0)} Q_A^{(i-1)}(u,d).$$

To determine the mean downtime $(\mu_A^{(i-1)})^{-1}$, we use the fact that a jump from up to down is with probability $\frac{\lambda_{A,1}^{(i)}}{\lambda_A^{(i)}}$ an actual breakdown and with probability $\frac{\lambda_{A,2}^{(i)}+\lambda_{A,3}^{(i)}}{\lambda_A^{(i)}}$ a starvation. In the first case we have a mean downtime of $(\mu_i)^{-1}$ and in the second case the mean downtime is obtained as the mean time in the downstate of the upstream arrival server, $(Q_A^{(i-1)}(d,u))^{-1}$. Thus, the estimated mean downtime of the arrival server is a weighted average over these two probabilities:

$$\frac{1}{\mu_A^{(i-1)}} = \frac{\lambda_{A,1}^{(i)}}{\lambda_A^{(i)}} \frac{1}{\mu_i} + \frac{\lambda_{A,2}^{(i)} + \lambda_{A,3}^{(i)}}{\lambda_A^{(i)}} \frac{1}{Q_A^{(i-1)}(d,u)}.$$

The average production rate in the upstate, $v_A^{(i)}$, is a weighted average over two possible speeds. Conditioned on the fact that D_{i-1} is producing (and thus, A_i is up), the upstream buffer B_{i-1} is empty with probability $\frac{p_{u,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)}+\pi_{d,u}^{(i-1)}-p_{d,u}^{(i-1)}(0)}$. In this case, the arrival server has to adjust its speed to its predecessor, which produces at $v_A^{(i-1)}$. With complementary probability, the arrival server is able to produce at its maximum speed v_i. This gives

$$v_A^{(i)} = v_i - \frac{p_{u,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0)} (v_i - v_A^{(i-1)}). \tag{7}$$

This completes the analysis of the arrival server.

For the departure server we obtain symmetrical expressions; blocking of the downstream subsystem is modeled as part of the downtime for this server. We define the following variables, obtained from the downstream subsystem:

- $\pi_{j,j'}^{(i+1)}$: the probability that A_{i+1} is in state j and D_{i+1} is in state j' ($j, j' \in \{u, d\}$, where u=up and d=down).
- $p_{j,j'}^{(i+1)}(b)$: the probability that A_{i+1} is in state j, D_{i+1} is in state j' and buffer B_{i+1} is full.
- $f_{j,j'}^{(i+1)}(b)$: the density of the buffer content of B_{i+1}, for A_{i+1} being in state j, D_{i+1} being in state j' and B_{i+1} being full.

- $Q_D^{(i+1)}(j, j')$: the rate of jumps from state j to state j' of D_{i+1}.

As for the arrival server, we can divide the transition rate from up to down of the departure server into three parts:

$$\lambda_D^{(i)} = \lambda_{D,1}^{(i)} + \lambda_{D,2}^{(i)} + \lambda_{D,3}^{(i)}.$$

The first part $\lambda_{D,1}^{(i)}$ corresponds to an actual breakdown of the departure server; the second and third part are jumps caused by blocking. Figure 6 illustrates the two different ways for the departure server to get blocked by the downstream subsystem.

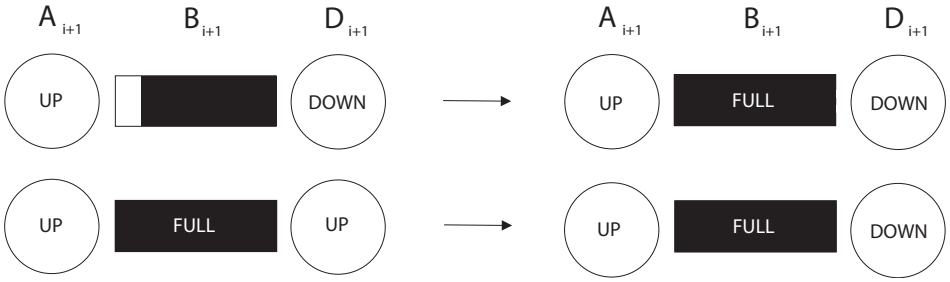

Figure 6: Two possible events in subsystem L_{i+1} causing D_i to get blocked

Since the analysis follows along the same (symmetrical) lines as for the arrival server, we only give the resulting expressions for the transition rates and speed;

$$\begin{aligned}
\lambda_{D,1}^{(i)} &= \lambda_{i+1}, \\
\lambda_{D,2}^{(i)} &= \frac{f_{u,d}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i-1)} - p_{u,d}^{(i-1)}(b)} v_D^{(i+1)}, \\
\lambda_{D,3}^{(i)} &= \frac{p_{u,u}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i+1)} - p_{u,d}^{(i+1)}(b)} Q_D^{(i+1)}(u, d), \\
\frac{1}{\mu_D^{(i)}} &= \frac{\lambda_{D,1}^{(i)}}{\lambda_D^{(i)}} \frac{1}{\mu_{i+1}} + \frac{\lambda_{D,2}^{(i)} + \lambda_{D,3}^{(i)}}{\lambda_D^{(i)}} \frac{1}{Q_D^{(i+1)}(d, u)}, \\
v_D^{(i)} &= v_{i+1} - \frac{p_{u,u}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i+1)} - p_{u,d}^{(i+1)}(b)} (v_{i+1} - v_D^{(i+1)}).
\end{aligned} \tag{8}$$

This completes the two-state analysis of subsystem L_i.

3.2 Subsystem analysis using three-state Markov processes

A more sophisticated approach is to model starvation in the arrival process and blocking in the departure process as separate states. This gives us a three-state Markov process for both servers. The states of the arrival server are defined as $\{u, d, st\}$: up, down, and starved; the states of the departure server are labeled $\{u, d, bl\}$: up, down, and blocked. Notice that the downstate in this subsection is not the same as the downstate in the previous subsection, where starvation and blocking were aggregated into the downstate. Figure 7 illustrates both Markov processes and corresponding transitions. Because of operationally dependent failures it is not possible to jump from starved to down or from blocked to down. It is readily seen that direct transitions in the opposite direction are also not possible.

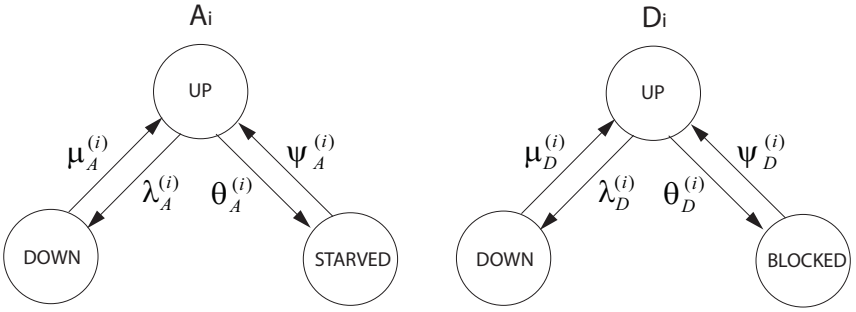

Figure 7: Three-state Markov chains describing the states and transitions of the arrival and departure server

The transition matrix and speed vector of the arrival server are given by

$$Q_A^{(i)} = \begin{pmatrix} -\lambda_A^{(i)} - \theta_A^{(i)} & \lambda_A^{(i)} & \theta_A^{(i)} \\ \mu_A^{(i)} & -\mu_A^{(i)} & 0 \\ \psi_A^{(i)} & 0 & -\psi_A^{(i)} \end{pmatrix}, \qquad r_A^{(i)} = \begin{pmatrix} v_A^{(i)} \\ 0 \\ 0 \end{pmatrix}.$$

The net speed in the upstate $v_A^{(i)}$ can be determined as, similar to (7),

$$v_A^{(i)} = v_i - \frac{p_{u,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0) + \pi_{st,u}^{(i-1)} - p_{st,u}^{(i-1)}(0)}(v_i - v_A^{(i-1)}).$$

This leaves us with obtaining the elements of $Q_A^{(i)}$. The transition rates from up to down and from down to up are retrieved easily from the input data:

$$\lambda_A^{(i)} = \lambda_i,$$

$$\mu_A^{(i)} = \mu_i.$$

We obtain the transition rate from up to starved in a similar way as $\lambda_{A,2}^{(i)} + \lambda_{A,3}^{(i)}$ in the previous subsection:

$$\begin{aligned} \theta_A^{(i)} \quad = \quad & \frac{f_{d,u}^{(i-1)}(0) + f_{st,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0) + \pi_{st,u}^{(i-1)} - p_{st,u}^{(i-1)}(0)} v_D^{(i-1)} \\ & + \frac{p_{u,u}^{(i-1)}(0)}{\pi_{u,u}^{(i-1)} + \pi_{d,u}^{(i-1)} - p_{d,u}^{(i-1)}(0) + \pi_{st,u}^{(i-1)} - p_{st,u}^{(i-1)}(0)} (Q_A^{(i-1)}(u,d) + Q_A^{(i-1)}(u,st)). \end{aligned}$$

When arrival server A_i is starved, the downstream buffer B_{i-1} is empty and the downstream arrival server A_{i-1} is either down (with probability $\frac{p_{d,u}^{(i-1)}(0)}{p_{d,u}^{(i-1)}(0)+p_{st,u}^{(i-1)}(0)}$) or starved (with probability $\frac{p_{st,u}^{(i-1)}(0)}{p_{d,u}^{(i-1)}(0)+p_{st,u}^{(i-1)}(0)}$). Arrival server A_i jumps from starved to up whenever A_{i-1} jumps to up. This gives the following transition rate from starved to up:

$$\psi_A^{(i)} = \frac{p_{d,u}^{(i-1)}(0)}{p_{d,u}^{(i-1)}(0) + p_{st,u}^{(i-1)}(0)} Q_A^{(i-1)}(d,u) + \frac{p_{st,u}^{(i-1)}(0)}{p_{d,u}^{(i-1)}(0) + p_{st,u}^{(i-1)}(0)} Q_A^{(i-1)}(st,u).$$

The transition matrix and net speed vector of departure server D_i are given by

$$Q_D^{(i)} = \begin{pmatrix} -\lambda_D^{(i)} - \theta_D^{(i)} & \lambda_D^{(i)} & \theta_D^{(i)} \\ \mu_D^{(i)} & -\mu_D^{(i)} & 0 \\ \psi_D^{(i)} & 0 & -\psi_D^{(i)} \end{pmatrix}, \qquad r_D^{(i)} = \begin{pmatrix} v_D^{(i)} \\ 0 \\ 0 \end{pmatrix}.$$

We can obtain $v_D^{(i)}$ as in (8),

$$v_D^{(i)} = v_{i+1} - \frac{p_{u,u}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i+1)} - p_{u,d}^{(i+1)}(b) + \pi_{u,bl}^{(i+1)} - p_{u,bl}^{(i+1)}(b)}(v_{i+1} - v_D^{(i+1)}).$$

In a symmetrical way as for the arrival server, we can obtain the transition rates for the departure machine:

$$\lambda_D^{(i)} = \lambda_{i+1},$$

$$\mu_D^{(i)} = \mu_{i+1},$$

$$\begin{aligned} \theta_D^{(i)} &= \frac{f_{u,d}^{(i+1)}(b) + f_{u,bl}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i-1)} - p_{u,d}^{(i-1)}(b) + \pi_{u,bl}^{(i+1)} - p_{u,bl}^{(i+1)}(b)} v_D^{(i+1)} \\ &\quad + \frac{p_{u,u}^{(i+1)}(b)}{\pi_{u,u}^{(i+1)} + \pi_{u,d}^{(i+1)} - p_{u,d}^{(i+1)}(b) + \pi_{u,bl}^{(i+1)} - p_{u,bl}^{(i+1)}(b)} (Q_D^{(i+1)}(u,d) + Q_D^{(i+1)}(u,bl)), \end{aligned}$$

$$\psi_D^{(i)} = \frac{p_{u,d}^{(i+1)}}{p_{u,d}^{(i+1)} + p_{u,bl}^{(i+1)}} Q_D^{(i+1)}(d,u) + \frac{p_{u,d}^{(i+1)}}{p_{u,d}^{(i+1)} + p_{u,bl}^{(i+1)}} Q_D^{(i+1)}(bl,u).$$

This concludes the analysis of this subsection.

3.3 Iterative algorithm

We analyze an N-stage production line L decomposed into subsystems $L_1, ..., L_{N-1}$. The following iterative procedure can be applied to obtain the throughput and average total buffer content, where either the method in Subsection 3.1 or 3.2 is used.

Step 0: Initialize
For each subsystem L_i, $i = 1, ..., N-1$, set $\lambda_A^{(i)} = \lambda_i$, $\lambda_D^{(i)} = \lambda_{i+1}$, $\mu_A^{(i)} = \mu_i$, and $\mu_D^{(i)} = \mu_{i+1}$. When using the method in Subsection 3.2, set $\theta_A^{(i)} = \theta_D^{(i)} = \psi_A^{(i)} = \psi_D^{(i)} = 0$. Set the iteration counter $k = 0$.

Step 1: Calculate
For each subsystem L_i, $i = 1, ..., N-1$:

- Increase the iteration counter by one, $k = k + 1$.
- Using the matrix-analytic methods as in Appendix A, calculate the steady state probabilities $\pi^{(i)}$, boundary probabilities $p^{(i)}(0)$ and $p^{(i)}(b)$, buffer content probability density function $f^{(i)}$, and throughput estimate $T_k^{(i)}$ (the subscript k refers to the estimate obtained in the kth iteration).
- If $i < N$-1, adjust the arrival server transition matrix $Q_A^{(i+1)}$ and speed vector $r_A^{(i+1)}$ of the downstream subsystem using the expressions from either Subsection 3.1 or 3.2.
- If $i > 1$, adjust the departure server transition matrix $Q_D^{(i-1)}$ and speed vector $r_D^{(i-1)}$ of the upstream subsystem using the expressions from either Subsection 3.1 or 3.2.

Step 2: Repeat
Repeat Step 1 until all throughput estimates have converged. In other words, if

$$\min_i \left(\frac{|T_k^{(i)} - T_{k-1}^{(i)}|}{T_{k-1}^{(i)}} \right) > \epsilon,$$

for some small ϵ, we do another iteration; otherwise we stop.

Note that the throughput values for the different subsystems are not necessarily the same when using this algorithm. In the literature, one often forces equal throughput values by applying a conservation of flow equation to determine estimates for (one of) the parameters. In our algorithm, it appeared to be not necessary, as in all experiments that we carried out, we found that the throughput values of all subsystems converge to the same value.

4 Results

In this section, we test the approximation method as proposed in the previous section. In the first subsection we test the algorithm on a large test set; the second subsection treats a practical case from brewery Heineken Den Bosch. The results of both are compared to those of a discrete-event simulation, where the 95% confidence intervals for the performance characteristics of interest have a width of at most 0.5%.

4.1 Test set

We test our method on a large test set, where five parameters are varied:

- Number of servers in the production line,
- Mean uptimes,
- Mean downtimes,
- Machine speeds,
- Buffer sizes.

Our "standard" case has mean uptimes of 10, mean downtimes of 1, machine speeds of 10, and buffer sizes of 10. In this case, each machine is theoretically up for 91% of the time, assuming it is not affected by other machines. The buffer sizes are chosen such that the buffers are exactly large enough to store fluid produced during an average downtime.

From the standard case we build a test set consisting of 1728 cases. The number of machines in the production line is varied between 4, 8, 12, and 16. The mean uptimes in the test set are 5, 10, and 20. Imbalance in mean uptimes is included by dividing the mean uptime of each even server by 2. The mean downtimes are varied between 0.5, 1, and 2. Also for the downtimes we include imbalance by dividing the mean downtimes of even servers by 2. For the machine speeds we chose three possible setups. In the first case, the speeds of all servers are 10. Secondly, the speed of each odd server is 10 and the speed of each even server is 15. The third setup is called a V-shape; in this case the first and last servers have a speed of 15 and the middle server(s) have a speed of 10. The speeds of the servers before the middle server(s) decrease linearly and the speeds of servers behind the middle server(s) increase linearly. For instance, in a six-machine production line the machine speeds in a V-shape production line would be 15-12.5-10-10-12.5-15. Such machine speeds are motivated by practice. For instance, the Heineken case

as described in the next subsection has a V-shape setup. The buffer sizes are chosen to be 1, 10, 25, and 50.

In Tables 1-5 the results for this test set are summarized. The left part of the tables shows the relative errors in throughput of our approximative methods compared to the simulation results; the right part shows the relative errors in average total buffer content (i.e. average amount of fluid in the system). For each table row, the average error is taken over all the cases specified in the first column. For example, the percentages in the first row of Table 2 give the average errors for all 288 cases with all mean uptimes equal to 5. The columns "Two-states" corresponds to the two-state approach as described in Subsection 3.1; "Three-state" refers to the three-state approach as described in Subsection 3.2.

To focus on the difference between the two approaches, Table 6 presents results for "extreme" cases with increasing imbalance in mean downtimes. In these cases the line length is 9, the mean down times are specified in the first column (e.g., in the last case $10, 1, 0.1, 10, 1, 0.1, 10, \ldots$); the other parameters are the same as for the standard case.

Table 1: Results for production lines with different lengths

Line length	**Error (%) in the throughput**		**Error (%) in avg buffer content**	
	Two-state	**Three-state**	**Two-state**	**Three-state**
4	0.50	0.44	0.63	0.60
8	1.39	1.24	0.55	0.50
12	2.13	1.93	0.57	0.52
16	2.74	2.52	0.63	0.58

Table 2: Results for production lines with different mean uptimes

Mean uptimes	**Error (%) in the throughput**		**Error (%) in avg buffer content**	
	Two-state	**Three-state**	**Two-state**	**Three-state**
5,5,5,5,...	2.22	2.01	0.72	0.66
5,2.5,5,2.5,...	2.32	2.08	0.72	0.67
10,10,10,10,...	1.59	1.45	0.60	0.55
10,5,10,5,...	1.78	1.62	0.58	0.54
20,20,20,20,...	1.01	0.93	0.50	0.46
20,10,20,10,...	1.21	1.11	0.47	0.43

Table 3: Results for production lines with different mean downtimes

Mean downtimes	**Error (%) in the throughput**		**Error (%) in avg buffer content**	
	Two-state	**Three-state**	**Two-state**	**Three-state**
0.5,0.5,0.5,0.5,...	0.88	0.88	0.28	0.28
0.5,0.25,0.5,0.25,...	0.95	0.83	0.27	0.26
1,1,1,1,...	1.54	1.54	0.31	0.31
1,0.5,1,0.5,...	1.81	1.53	0.42	0.36
2,2,2,2,...	2.08	2.08	0.79	0.79
2,1,2,1,...	2.87	2.34	1.51	1.31

4.2 Heineken production line

We analyze a production line at Heineken Den Bosch, where retour bottles are processed by eleven machines in series. Between each of the machines is a conveyor belt, which is used for transportation and for buffering. The number of bottles processed per hour is very large, typically in the order of 30,000, which justifies considering the flow of bottles through the machines as fluid.

The production line consists of the following eleven machines:

Table 4: Results for production lines with different machine speeds

Machine speeds	Error (%) in the throughput		Error (%) in avg buffer content	
	Two-state	Three-state	Two-state	Three-state
10,10,10,10,...	1.52	1.37	0.24	0.24
10,15,10,15,...	2.88	2.68	1.46	1.34
15,...,10,...,15	0.66	0.55	0.09	0.07

Table 5: Results for production lines with different buffer sizes

Buffer sizes	Error (%) in the throughput		Error (%) in avg buffer content	
	Two-state	Three-state	Two-state	Three-state
1	0.86	0.78	0.18	0.19
10	2.42	2.16	0.32	0.31
25	2.02	1.83	0.54	0.47
50	1.46	1.36	1.35	1.23

- Retour bottles enter the production line in crates on pallets; these pallets are put on the production line by the **depallatizer**.
- The **logo detection** checks the logo on the crates.
- Bottles are taken out of the crates by the **depacker**. From here on we follow the bottles.
- The bottles are washed by the **bottle-washer**.
- The **Empty Bottle Inspector (EBI)** checks the bottles for bursts. Broken bottles are rejected and taken out of the production line. The number of rejected bottles is typically very small, so they do not have a significant impact on the throughput.
- The **filler** grabs the bottles, fills them with beer, closes the bottles, and puts them on a conveyor belt.
- Next, the bottles move to the **pasteurizer**.
- The **labeler** provides the bottles with the correct labels.
- In the **packer**, the bottles are either packed into six-packs or into crates.
- Before leaving the production line, the filled crates have to be checked by the **cratemanco**.
- Lastly, the **palletizer** puts the crates on pallets, after which they can be transported to customers.

Each machine has an up- and down behavior. Typically, machines cannot go down whenever they are not producing because of blocking and starvation. From a large data set, we obtained the mean uptimes and downtimes in hours, as can be seen in table 7. From the same data set we were able to obtain the maximum machine speeds, since the number of bottles processed per time unit of each machine was recorded. By measuring the length and width of the conveyor belts we were able to obtain the maximum number of bottles fitting into each buffer. We also investigated whether there exist dependencies between

Table 6: Results for production lines with increasing imbalance in mean downtimes

Mean downtimes	Error (%) in the throughput	
	Two-state	Three-state
1,1,1,...	1.71	1.71
1.33,1,0.67,1.33,1,0.67...	2.22	1.93
2,1,0.5,2,1,0.5,...	3.82	2.58
4,1,0.25,4,1,0.25,...	7.95	3.62
10,1,0.1,10,1,0.1,...	11.86	3.99

Table 7: Data for the Heineken production line

Machine name	Mean uptime (hrs)	Mean downtime (hrs)	Machine speed	Buffer size
Depalletizer	1.3712	0.0595	48349	3647
Logo-detection	0.5821	0.0256	43284	1823
Depacker	0.1389	0.0283	43284	6895
Bottle-washer	0.3229	0.0473	40389	5300
EBI	0.5828	0.0336	37407	270
Filler	0.4244	0.0361	37407	4874
Pasteurizer	3.9386	0.0806	40170	7014
Labeler	0.2930	0.0246	37094	6622
Packer	0.2698	0.0349	40988	4630
Cratemanco	2.8161	0.1517	41500	6945
Palletizer	1.9550	0.0685	42559	-

breakdown rates and machine speeds; there appeared to be no evidence for such dependencies, justifying our assumption of constant breakdown rates.

We compared the estimated throughput of a discrete event simulation to the results obtained by using our approximation method as described in Section 3. Simulation gives an average throughput of 31,523 bottles per hour. The estimated throughput of our two-state approach as described in Section 3.1 is 32,046 bottles per hour, which is a difference of 1.66 $\%$ compared to simulation. The three-state approach of Section 3.2 estimates a throughput of 31,976 bottles per hour, which is a difference of 1.44 $\%$ with simulation.

5 Discussion

In this section, we discuss the results of the testset as presented in the previous section. We are interested in differences in throughput and mean buffer content between our approximation and a simulation model. Both the two-state model and the three-state model perform well on this testset. The average error in throughput is 1.69% and 1.53% respectively; the average error in mean buffer content is 0.60% and 0.55% respectively.

Table 1 shows that the error in throughput is larger for longer production lines, although the results are still satisfying for production lines with 16 servers. More surprisingly, there seems to be no dependency between the length of the production line and the error in mean buffer content. In Table 2, it can be seen that results are better for lines with larger mean uptimes. Table 3 shows that for mean downtimes the opposite holds: for smaller mean downtimes, the results are better. A possible explanation for this effect is that we estimate time until a starvation or blocking by an exponential distribution, while these transitions are not exponentially distributed in reality. For cases with small mean uptimes or large mean downtimes, these transitions occur more frequently. Tables 2 and 3 also show that the results are fairly insensitive to imbalance in mean uptimes; while imbalance in mean downtimes does seem to give slightly worse results.

In Table 4, we see the average results depending on the machine speed configuration. When adding imbalance, we see that results become worse for lines with jumping speeds and better for lines with a V-shape speed configuration, which seems a surprising results at first sight. This effect can be explained by the following argument. As said, we approximate the non-exponentially distributed time until starvation or blocking by an exponential distribution. For lines with jumping speeds, we deal with these type of transitions a lot, which is why results deteriorate for these cases. For the V-shape production lines there is another effect playing a role. In the first part of these lines, speeds are decreasing and buffers are typically full when all servers are producing for some time. When all buffers in the first part are full, the time until blocking is exponentially distributed (as we estimated it) and starvation will not occur. In the second part of the production line, speeds increase and the buffers are frequently empty. This implies that the time until starvation is mostly exponentially distributed and blocking is not likely to happen very

often.

Table 5 shows that the error in throughput is the smallest for production lines with very small buffers or large buffers. This again has to do with the frequency of occurrence of jumps to starvation and blocking. The average error in mean buffer content is the largest for lines with large buffers.

To show the difference between the two-state and the three-state model, we focus on imbalance in mean downtimes. As can be seen in Table 2, both approximations give exactly the same results for production lines with equal mean downtimes. For these cases, the mean time spent in the down-state is the same as the mean time spent in the starved or blocked-state in the three-state model (see Figure 7). Therefore combining these states, as we do in the two-state model, will give exactly the same approximations. For cases with imbalance in mean downtimes, we see that the three-state model performs slightly better than the two-state model. However as we increase the imbalance in mean downtimes and take it to the extreme, we see in Table 6 that the three-state model performs significantly better.

As can be seen in Table 7, the Heineken production line has imbalance in all parameters. The results for this line are very satisfying, with an average error of 1.66% and 1.44% in throughput compared to simulation for the two-state and three-state approach respectively.

6 Conclusions

In this paper, we develop an approximative method to analyze production lines with fluid flows and exponentially distributed breakdown and repair times. We assume operationally dependent failures, meaning that a server cannot break down when it is not producing because of starvation or blocking. The method decomposes the line into subsystems, each subsystem consisting of an arrival server, a departure server, and a buffer in between. In the description of the arrival server, we include the fact that this server can be starved or has to slow down because of the upstream part of the production line. Similarly, in the description of the departure server we include possible blocking or speed adaption because of the downstream part of the production line.

We propose two approaches to analyze a subsystem. In the first approach, we model the behavior of both the arrival and departure servers as two-state Markov chains, with a state where the server is producing and a state where the server is not producing, either because of a breakdown or because of starvation or blocking. The second approach is to model both servers as three-state Markov chains, where blocking and starvation are treated as separate states. The transition rates and average speed of both servers are determined iteratively.

For both two-state and three-state models, the iterative algorithm to determine the parameter values converge rapidly. By using a large test set, we compare the throughput and mean buffer content of both approximations to those obtained from a discrete-event simulation. The average error in both quantities is small. For cases with lots of imbalance in mean downtimes, the three-state model performs significantly better than the two-state model. For the real-life production line at Heineken Den Bosch, both models produces accurate estimates of the throughput.

Since in reality breakdown and repair distributions are typically not exponentially distributed, further research will include the extension towards general breakdown and repair distributions.

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

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] 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.

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

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

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

[12] 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.

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

[14] 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.

[15] 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.

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

[17] 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.

[18] 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.

[19] 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.

A Matrix-analytic methods for two-stage production lines with ODFs

This appendix is devoted to obtaining the steady state distribution and performance measures of the two-stage system as defined in Section 2, using matrix-analytic methods. The method is based on the analysis of Soares and Latouche [17]. The system is characterized by generator Q, speed vector r, full-buffer process generator Q^F and empty-buffer process generator Q^E. In Section A1, we analyze a system where the net speed r_i in each state $i \in S$ is either 1 or -1. Section A2 describes how to obtain the density function for a general system by transforming it into a system with two possible net speeds. Finally, Section A3 shows how to derive performance measures from the steady state distribution.

A.1 Analysis of two-stage systems with two possible net speeds

We assume that $r_i = 1$ or $r_i = -1$ for each $i \in S$. First, we divide the state space in two sets: S_+ with phases i for which $r_i = 1$, and S_- with phases i for which $r_i = -1$. We partition the transition matrix Q and the speed vector r in a similar way,

$$Q = \begin{pmatrix} Q_{++} & Q_{+-} \\ Q_{-+} & Q_{--} \end{pmatrix}, \qquad r = \begin{pmatrix} r_+ \\ r_- \end{pmatrix}.$$

We introduce a matrix Ψ, where the element Ψ_{ij} for $i \in S_+$ and $j \in S_-$ is the probability that when starting in state i with an empty buffer, the system returns to an empty buffer situation and does so in state j. To obtain Ψ, we use matrix-analytic methods. The following procedure is proposed in [16] (Appendix B). Define $c \leq \max_{i \in S} |Q_{ii}|$, $P = I + \frac{1}{c}Q$, and $V = I + \frac{1}{c}U$. A Quasi-Birth-and-Death process can be defined with transition matrices

$$A_0 = \begin{pmatrix} \frac{1}{2}I & 0 \\ 0 & 0 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} \frac{1}{2}P_{++} & 0 \\ P_{-+} & 0 \end{pmatrix}, \qquad A_2 = \begin{pmatrix} 0 & \frac{1}{2}P_{+-} \\ 0 & P_{--} \end{pmatrix}.$$

We can now define

$$G = \begin{pmatrix} 0 & \Psi \\ 0 & V \end{pmatrix},$$

where G is the minimal nonnegative solution of the equation $G = A_2 + A_1 G + A_0 G^2$. The logarithm reduction algorithm of [11] can be used to efficiently calculate the minimal nonnegative solution of this equation.

We define K and U as the generator of an irreducible Markov process on S_+ and S_- respectively, where

$$K = Q_{++} + \Psi Q_{-+},$$

$$U = Q_{--} + Q_{-+}\Psi.$$

We can also consider the level-reversed process, which has the same generator Q and (reversed) net speed rates $\hat{r}_i = -r_i$. The matrices $\hat{\Psi}$, $\hat{K}$ and $\hat{U}$ for the level-reversed process have the same interpretation as the matrices Ψ, K and U for the original queue, so we obtain

$$\hat{K} = Q_{--} + \hat{\Psi} Q_{+-},$$

$$\hat{U} = Q_{++} + Q_{+-}\hat{\Psi}.$$

Soares and Latouche [17] show that the stationary density function of the buffer content is given by

$$(f_+(x), f_-(x)) = (p_+^{(b)}, p_-^{(0)}) \begin{pmatrix} 0 & Q_{+-} \\ Q_{-+} & 0 \end{pmatrix} \begin{pmatrix} I & e^{Kb}\Psi \\ e^{\hat{K}b}\hat{\Psi} & I \end{pmatrix}^{-1} \begin{pmatrix} e^{Kx} & e^{Kx}\Psi \\ e^{\hat{K}(b-x)}\hat{\Psi} & e^{\hat{K}(b-x)} \end{pmatrix}, \quad (9)$$

where x is the buffer content. We define $p_+^{(b)}$ as the upper bound probability vector, where the ith element denotes the probability that the system is in state $i \in S_+$ and the buffer is full. Similarly, $p_-^{(0)}$ is the vector of lower bound probabilities, where the ith element represents the probability of the system being in state $i \in S_-$ and the buffer being empty. To obtain the boundary probability vectors $(p_+^{(b)}, p_-^{(0)})$, we have to solve the following system:

$$(p_+^{(b)}, p_-^{(0)})W = 0, \tag{10}$$

$$\sum_{i=1}^{N} \int_0^b (f_+(x), f_-(x))dx = 1, \tag{11}$$

where

$$W = \begin{pmatrix} Q_{++}^F + Q_{+-}^F \hat{\Psi}_{-+} & Q_{+-}^F \hat{\Lambda}_{--} \\ Q_{-+}^E \Lambda_{++} & Q_{--}^E + Q_{-+}^E \Psi_{+-} \end{pmatrix}.$$

The matrices Ψ_{+-}, $\hat{\Psi}_{-+}$, Λ_{++} and $\hat{\Lambda}_{--}$ are obtained in [17] as

$$\Psi_{+-} = (\Psi - e^{\hat{U}b}\Psi e^{Ub})(I - \hat{\Psi}e^{\hat{U}b}\Psi e^{Ub})^{-1},$$
$$\hat{\Psi}_{-+} = (\hat{\Psi} - e^{Ub}\hat{\Psi}e^{\hat{U}b})(I - \Psi e^{Ub}\hat{\Psi}e^{\hat{U}b})^{-1},$$
$$\Lambda_{++} = (I - \Psi\hat{\Psi})e^{\hat{U}b}(I - \Psi e^{Ub}\hat{\Psi}e^{\hat{U}b})^{-1},$$
$$\hat{\Lambda}_{--} = (I - \hat{\Psi}\Psi)e^{Ub}(I - \hat{\Psi}e^{\hat{U}b}\Psi e^{Ub})^{-1}.$$

The normalizing equation (11) can be written as

$$\sum_{i=1}^{N} \int_0^b (f_+(x), f_-(x))dx =$$

$$(p_+^{(b)}, p_-^{(0)}) \begin{pmatrix} 0 & Q_{+-} \\ Q_{-+} & 0 \end{pmatrix} \begin{pmatrix} I & e^{Kb}\Psi \\ e^{\hat{K}b}\hat{\Psi} & I \end{pmatrix}^{-1} \begin{pmatrix} \int_0^b e^{Kx}dx & \int_0^b e^{Kx}dx\Psi \\ \int_0^b e^{\hat{K}(b-x)}dx\hat{\Psi} & \int_0^b e^{\hat{K}(b-x)}dx \end{pmatrix} \mathbf{1}_N. \tag{12}$$

Since either K or $\hat{K}$ in this equation is singular, we make use of the group inverse concept. The group inverse $P^\#$ of a matrix P is a unique matrix satisfying $PP^\#P = P$, $P^\#PP^\# = P^\#$, and $PP^\# = P^\#P$. With the following property we are able to compute the group inverse:

$$\begin{cases} P^\#P = I - vu, \\ P^\#v = 0, \end{cases}$$

where u and v are respectively the left and right eigenvectors of P for the eigenvalue 0, normalized by $uv = 1$ and $u1 = 1$.

The integrals in (12) can be computed by using either the inverse or the group inverse of K and $\hat{K}$ to obtain (see [17] for the proof):

$$\int_0^b e^{Kx}dx = \begin{cases} (e^{Kb} - I)K^\# + bvu & \text{if } K \text{ is singular,} \\ (e^{Kb} - I)K^{-1} & \text{otherwise,} \end{cases}$$

$$\int_0^b e^{\hat{K}(b-x)}dx = \begin{cases} (e^{\hat{K}b} - I)\hat{K}^\# + b\hat{v}\hat{u} & \text{if } \hat{K} \text{ is singular,} \\ (e^{\hat{K}b} - I)\hat{K}^{-1} & \text{otherwise.} \end{cases}$$

The values for $(p_+^{(b)}, p_-^{(0)})$ obtained from (10)-(11) can be plugged into (9) to complete the buffer content density function. By integrating (9) and adding the probabilities at the boundaries, we can retrieve the steady state probability vector π:

$$\pi = (\pi_+, \pi_-) = (p_+^{(b)}, p_-^{(0)}) + \int_0^b (f_+(x), f_-(x))dx.$$

A.2 Analysis of general two-stage systems

In this subsection, we analyze a general two-stage fluid system with a finite set of states $\tilde{S}$, transition matrix $\tilde{Q}$ and net speed vector $\tilde{r}$, where $\tilde{r}_i$ can have any real value for all $i \in S$. We analyze this system by transforming it into a system with a set of states S, rate matrix Q and speed vector r, where $r_i = 1$ or $r_i = -1$ for each $i \in S$. From this system, we can obtain the steady state distribution as described in the previous subsection. Secondly, we show how to transform this distribution into the steady state distribution of the original system, $\tilde{f}$.

First, we divide the state space $\tilde{S}$ into three subsets; S_+, S_- and S_0 consist of all states $i \in \tilde{S}$ for which $\tilde{r}_i > 0$, $\tilde{r}_i < 0$ and $\tilde{r}_i = 0$ respectively. We divide the rate matrices $\tilde{Q}$, $\tilde{Q}^E$ and $\tilde{Q}^F$ accordingly:

$$\tilde{Q} = \begin{pmatrix} \tilde{Q}_{00} & \tilde{Q}_{0+} & \tilde{Q}_{0-} \\ \tilde{Q}_{+0} & \tilde{Q}_{++} & \tilde{Q}_{+-} \\ \tilde{Q}_{-0} & \tilde{Q}_{-+} & \tilde{Q}_{--} \end{pmatrix}, \qquad \tilde{Q}^E = \begin{pmatrix} \tilde{Q}^E_{00} & \tilde{Q}^E_{0+} & \tilde{Q}^E_{0-} \\ \tilde{Q}^E_{+0} & \tilde{Q}^E_{++} & \tilde{Q}^E_{+-} \\ \tilde{Q}^E_{-0} & \tilde{Q}^E_{-+} & \tilde{Q}^E_{--} \end{pmatrix}, \qquad \tilde{Q}^F = \begin{pmatrix} \tilde{Q}^F_{00} & \tilde{Q}^F_{0+} & \tilde{Q}^F_{0-} \\ \tilde{Q}^F_{+0} & \tilde{Q}^F_{++} & \tilde{Q}^F_{+-} \\ \tilde{Q}^F_{-0} & \tilde{Q}^F_{-+} & \tilde{Q}^F_{--} \end{pmatrix}.$$

From this system we determine the following quantities. The buffer content pdf is determined as $\tilde{f} = (\tilde{f}_0, \tilde{f}_+, \tilde{f}_-)$. The (row) probability vector of being in either of the states in $i \in \tilde{S} = \{S_0, S_+, S_-\}$ is given by $\tilde{\pi} = (\tilde{\pi}_0, \tilde{\pi}_+, \tilde{\pi}_-)$. The row vector $\tilde{p}^{(0)} = (\tilde{p}^{(0)}_0, \tilde{p}^{(0)}_+, \tilde{p}^{(0)}_-)$ contains the steady state probabilities of the system being in state $i \in \tilde{S}$ and the buffer being empty. The ith element of row vector $\tilde{p}^{(b)} = (\tilde{p}^{(b)}_0, \tilde{p}^{(b)}_+, \tilde{p}^{(b)}_-)$ denotes the probability of being in state $i \in \tilde{S}$ and the buffer being full. Note that $\tilde{p}^{(0)}_+(i) = \tilde{p}^{(b)}_-(i) = 0$, since the buffer cannot be empty if the net speed is positive and it cannot be full if the net speed is negative. For ease of notation, we write the (i,j)th element of a matrix G as $G(i,j)$ and the ith element of a vector g as $g(i)$ from here on.

From the original process, we delete the states in S_0 to obtain a new process with state space $\bar{S} = S_+ \cup S_-$, net speed vector $\bar{r}$ and transition matrix

$$\bar{Q} = \begin{pmatrix} \tilde{Q}_{++} & \tilde{Q}_{+-} \\ \tilde{Q}_{-+} & \tilde{Q}_{--} \end{pmatrix} + \begin{pmatrix} \tilde{Q}_{+0} \\ \tilde{Q}_{-0} \end{pmatrix} (-\tilde{Q}_{00})^{-1} (\tilde{Q}_{0+}, \tilde{Q}_{0-}).$$

By changing the time scale, we can change this process with nonzero speeds into a process with two possible speeds. We define $\bar{C} = \text{diag}(\tilde{r}_i : i \in S_+ \cup S_-)$. The transition matrix of the simplified process with state space $S = S_+ \cup S_-$ is given by

$$Q = |\bar{C}|^{-1} \bar{Q}.$$

By using the method as described in the previous subsection, we can determine the probability density function f, boundary probability vectors $p^{(0)}$ and $p^{(b)}$, and probability vector π of the simplified process. By rescaling these results we obtain the function $\bar{f}$ and vectors $\bar{p}^{(0)}$, $\bar{p}^{(b)}$, and $\bar{\pi}$ corresponding to the process described by $\bar{Q}$ and $\bar{r}$:

$$\bar{f} = (\pi|\bar{C}|\mathbf{1})^{-1} f |\bar{C}|^{-1}, \qquad \bar{\pi} = (\pi|\bar{C}|\mathbf{1})^{-1} \pi |\bar{C}|^{-1},$$

$$\bar{p}^{(0)} = (\pi|\bar{C}|\mathbf{1})^{-1} p^{(0)} |\bar{C}|^{-1}, \qquad \bar{p}^{(b)} = (\pi|\bar{C}|\mathbf{1})^{-1} p^{(b)} |\bar{C}|^{-1},$$

where $\mathbf{1}$ is the column vector of ones. To determine $\tilde{f}(i)$, $\tilde{\pi}(i)$, $\tilde{p}^{(0)}(i)$, $\tilde{p}^{(b)}(i)$ of the original process for all states $i \in S_+ \cup S_-$, we have to scale the original expressions $\bar{f}(i)$, $\bar{\pi}(i)$, $\bar{p}^{(0)}(i)$, $\bar{p}^{(b)}(i)$:

$$\tilde{f}(i) = (1 - \sum_{j \in S_0} \tilde{\pi}(j)) \bar{f}(i), \qquad \tilde{\pi}(i) = (1 - \sum_{j \in S_0} \tilde{\pi}(j)) \bar{\pi}(i), \tag{13}$$

$$\tilde{p}^{(0)}(i) = (1 - \sum_{j \in S_0} \tilde{\pi}(j)) \bar{p}^{(0)}(i), \qquad \tilde{p}^{(b)}(i) = (1 - \sum_{j \in S_0} \tilde{\pi}(j)) \bar{p}^{(b)}(i). \tag{14}$$

Since we are dealing with an irreducible Markov chain, we obtain $\tilde{\pi}(i)$, $\tilde{f}(i)$, $\tilde{p}^{(0)}(i)$, and $\tilde{p}^{(b)}(i)$ for $i \in S_0$ by using balance arguments. These arguments imply that the total number of jumps into a certain state should be equal to the number of jumps out of that state. By applying balance arguments, we derive sets of equations from which we can solve the unknown quantities.

- To determine the steady state probabilities $\tilde{\pi}(i)$ for $i \in S_0$, we argue that the number of jumps into a state $i \in S_0$ should be equal to the number of jumps out of that state. The corresponding balance equations are given by

$$-\tilde{\pi}(i)\tilde{Q}(i,i) = \sum_{j \in S_+ \cup S_-} \Big[(\tilde{\pi}(j) - \tilde{p}^{(b)}(j) - \tilde{p}^{(0)}(j))\tilde{Q}(j,i) + \tilde{p}^{(b)}(j)\tilde{Q}^F(j,i) + \tilde{p}^{(0)}(j)\tilde{Q}^E(j,i) \Big] +$$

$$\sum_{j \in S_0} \tilde{\pi}(j)\tilde{Q}(j,i).$$

  By substituting (13)-(14) into these equations and writing in matrix-form, we obtain the following set of equations, from which we can solve $\tilde{\pi}_0$ explicitly:

$$\left[(\hat{\pi}_+ - \hat{p}_+^{(b)}, \hat{\pi}_- - \hat{p}^{(0)}) \begin{pmatrix} \tilde{Q}_{+0} \\ \tilde{Q}_{-0} \end{pmatrix} + \tilde{p}_+^{(b)}\tilde{Q}_{+0}^F + \tilde{p}_-^{(0)}\tilde{Q}_{-0}^E \right] (1 - \tilde{\pi}_0 \mathbf{1}) + \tilde{\pi}_0 \tilde{Q}_{00} = 0.$$

  The solution can be substituted into (13) and (14) to obtain all the quantities for states in S_+ and S_-.

- The balance arguments for the functions $\tilde{f}(i)$ for $i \in S_0$ imply that the total number of jumps into a state with zero-speed and buffer level x, $0 < x < b$, should be equal to the number of jumps out of that state. Note that the buffer level does not change when the system is in a state in S_0, so we only need to include phase transitions in the balance equations.

$$-\tilde{Q}(i,i)\tilde{f}(i) = \sum_{j \in S, j \neq i} \tilde{Q}(j,i)\tilde{f}(j).$$

  Solving this set of equations for $\tilde{f}_0$ gives (in matrix-notation)

$$\tilde{f}_0 = -(\tilde{f}_+\tilde{Q}_{+0} - \tilde{f}_-\tilde{Q}_{-0})\tilde{Q}_{00}^{-1}. \tag{15}$$

- Finally, we obtain $\tilde{p}_0^{(0)}(i)$ and $\tilde{p}_0^{(b)}(i)$ for $i \in S_0$ by using balance equations at the boundaries:

$$-\tilde{Q}^E(i,i)\tilde{p}_0^{(0)}(i) = \sum_{j \in S_-, j \neq i} \tilde{Q}^E(j,i)\tilde{p}_-^{(0)}(j),$$

$$-\tilde{Q}^F(i,i)\tilde{p}_0^{(b)}(i) = \sum_{j \in S_+, j \neq i} \tilde{Q}^F(j,i)\tilde{p}_+^{(b)}(j).$$

  Solving and writing in matrix-notation gives the following set of equations, which we can solve for $\tilde{p}_0^{(b)}$ and $\tilde{p}_0^{(0)}$ by plugging in (14) for $\tilde{p}_-^{(0)}$ and $\tilde{p}_+^{(b)}$:

$$\tilde{p}_-^{(0)}\tilde{Q}_{-0}^E + \tilde{p}_0^{(0)}\tilde{Q}_{00}^E = 0.$$

$$\tilde{p}_+^{(b)}\tilde{Q}_{+0}^F + \tilde{p}_0^{(b)}\tilde{Q}_{00}^F = 0,$$

A.3 Performance measures

We are interested in the mean buffer content and throughput of the system as described in the beginning of this section. As in the previous subsection, we split the state space into three parts, with S_0, S_+, and S_- containing the states with zero, positive and negative speeds respectively.

Using the analysis of the previous subsections, we can obtain the mean buffer content $E(BC)$ by taking the expectation of the buffer content between the lower and upper bound and adding a term for the maximum buffer content at level b:

$$E(BC) = \left[\int_0^b (f_0(x), f_+(x), f_-(x)) x dx \; + \; b(p_+^{(b)}, 0) \right] \mathbf{1}_N,$$

where $\mathbf{1}_N$ is the column vector of ones of size N. For this expression, we show how to obtain the integral $\int_0^b (f_+(x), f_-(x)) x dx \mathbf{1}_N$. From this, the integral $\int_0^b x f_0(x) dx$ follows by applying a similar analysis using $f_0(x)$ from (15). We use (9) to write

$$\sum_{i=1}^{N} \int_0^b (f_+(x), f_-(x)) x dx =$$

$$(p_+^{(b)}, p_-^{(0)}) \begin{pmatrix} 0 & Q_{+-} \\ Q_{-+} & 0 \end{pmatrix} \begin{pmatrix} I & e^{Kb}\Psi \\ e^{\hat{K}b}\hat{\Psi} & I \end{pmatrix}^{-1} \begin{pmatrix} \int_0^b e^{Kx} x dx & \int_0^b e^{Kx} x dx \Psi \\ \int_0^b e^{\hat{K}(b-x)} x dx \hat{\Psi} & \int_0^b e^{\hat{K}(b-x)} x dx \end{pmatrix} \mathbf{1}_N. \quad (16)$$

The integrals $\int_0^b e^{Kx} x dx$ and $\int_0^b e^{\hat{K}(b-x)} x dx$ can be obtained using integration by parts, where either the inverse or the group inverse of K and $\hat{K}$ are used (see [17] for the proof):

$$\int_0^b e^{Kx} x dx = \begin{cases} K^{\#}(be^{Kb} - K^{\#}(e^{Kb} - I)) + \frac{1}{2} b^2 vu & \text{if } K \text{ is singular,} \\ K^{-1}(be^{Kb} - K^{-1}(e^{Kb} - I)) & \text{otherwise,} \end{cases} \quad (17)$$

$$\int_0^b e^{\hat{K}(b-x)} x dx = \begin{cases} -\hat{K}^{\#}(bI - \hat{K}^{\#}(e^{\hat{K}b} - I)) + \frac{1}{2} b^2 \hat{v}\hat{u} & \text{if } \hat{K} \text{ is singular,} \\ -\hat{K}^{-1}(bI - \hat{K}^{-1}(e^{\hat{K}b} - I)) & \text{otherwise.} \end{cases} \quad (18)$$

To determine the throughput T, recall that each state $i = (i_A, i_D) \in S$ has corresponding arrival machine speed r_{A,i_A} and departure machine speed r_{D,i_D}. By using the steady state probabilities π and boundary probabilities $p^{(0)}$ and $p^{(b)}$ we obtain the throughput as

$$T = \sum_{i=(i_A,i_D)} (\pi_i - p_i^{(b)}) r_{A,i_A} + p_i^{(b)} r_{D,i_D}, \quad (19)$$

or, equivalently,

$$T = \sum_{i=(i_A,i_D)} (\pi_i - p_i^{(0)}) r_{D,i_D} + p_i^{(0)} r_{A,i_A}. \quad (20)$$