Ann Oper Res
DOI 10.1007/s10479-011-0961-9

A decomposition method for approximate evaluation of continuous flow multi-stage lines with general Markovian machines

Marcello Colledani · Stanley B. Gershwin

Abstract In this paper, a decomposition method for evaluating the performance of continuous flow lines with machines characterized by general Markovian fluid models and finite capacity buffers is proposed. This study uses the exact solution of general two-stage Markovian fluid models as a building block. Decomposition equations are provided to propagate the effect of partial and complete blocking and starvation phenomena throughout the system. A decomposition algorithm that solves the new decomposition equations is proposed. Numerical results prove the good accuracy of the developed method. In particular, a comparison with existing techniques shows that our method is generally more accurate, especially in the estimation of the average buffer levels. Moreover, additional information can be collected by the application of our approach which enables a deeper analysis of the system behavior. Finally, the generality of the approach allows for modeling and studying many different system configurations within a unique framework, also including several previously uninvestigated layouts.

Keywords Asynchronous flow lines · Decomposition · Markovian fluid models · Finite capacity buffers

1 Introduction

1.1 Objective

In this paper, we consider a multi-stage fluid flow system with finite buffer capacity (Fig. 1). The dynamics of each stage is modeled by a continuous time-discrete state Markov chain

M. Colledani (✉)
Department of Mechanical Engineering, Politecnico di Milano, Via La Masa, 1, Milan, Italy
e-mail: marcello.colledani@polimi.it

S.B. Gershwin
Laboratory for Manufacturing and Productivity, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139-4307, USA
e-mail: gershwin@mit.edu

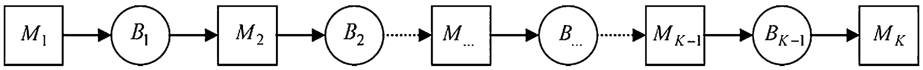


Fig. 1 Production line with K machines

of general complexity. To every state, a specific processing rate is assigned. Continuous flow models have been proposed in the literature as an alternative to discrete synchronous and asynchronous models for modeling the behavior of discrete part manufacturing systems, where processing rates are inhomogeneous, i.e. machines are characterized by different speeds (Gershwin and Dallery 1992). Systems with this feature are found in several industries, e.g. the automotive industry, the white goods industry, the electronic and mechanical component manufacturing and food industries.

1.2 Literature survey

Continuous flow models approximate the discrete flow of parts in the system to the flow of a fluid through a set of capacitated tanks (modeling buffers) and machines, that act as valves regulating the material flow rate at every production stage. Given this feature, continuous flow models can be also used to study continuous manufacturing plants such as those found in the chemical industry, in the dairy industry and in the steel-making industry. Gershwin and Dallery (1992) proposed a survey on models adopted to analyze the behavior of manufacturing lines. Specifically, the relationship between continuous models and discrete synchronous and asynchronous models was discussed. Experimental results were reported (Alvarez-Vargas et al. 1994) that show how, under the assumption of MTS (Multiple Time Scale), the continuous flow model is a suitable approximation of the discrete asynchronous model. MTS essentially means that the up times and down times of the machines in the system are much larger than the processing times, which is usually the case in production systems. Moreover, techniques to approximate synchronous models by continuous flow models have been proposed in the literature. One effective technique proposed in Gershwin and Schick (1980) and Gershwin (1994) is based on the so called δ-transformation, which consists in obtaining the parameters of the continuous model of the system by scaling the parameters of the synchronous model using a small quantity δ. If δ is small and the MTS assumption is valid, then the approximation results appear to be fairly good.

Recently a few works have presented successful applications of continuous flow models for the analysis of real life manufacturing systems. In Burman et al. (1998) a set of analytical manufacturing system models, including continuous flow models, was proposed to study ink-jet printer manufacturing lines in Hewlett-Packard. The authors showed great benefits for the company through the addition of buffer space in the manufacturing line supported by the use of the analytical models. Patchong et al. (2003) reports a study of the car body production at PSA Peugeot Citroën, where analytical models were used to improve the production rate of the system. In this application, a continuous model for the analysis of lines with stages having parallel machines was adopted. Alden et al. (2006) presents the methodologies and the results achieved by General Motors Corporation in a long term project to forecast and improve the production rate of its production lines. Continuous flow models were used to approximate the discrete flow of parts in the systems.

Given their versatility, multi-stage continuous flow models have been extensively studied in the literature, including specific Markovian structures for each process stage. Simple up-down machines were originally considered in Burman (1995), Dallery and Le Bihan (1997, 2000), Tan and Yeralan (1997), Glassey and Hong (1993). Single up-multiple down state machines (also called multiple failure modes machines) were considered in Levantesi et al. (2003). Systems featuring stages with parallel identical machines also received some attention (Patchong and Willaeys 2001; Burman 1995). More recently, a continuous model for the analysis of systems including flexible machines able to process multiple part types has been developed (Colledani and Tolio 2004). Finally, more complex machine structures involving out of control states and quality control issues have been analyzed in Gershwin and Kim (2008), Colledani (2008).

However, models for evaluating multi-stage lines with generally complex Markovian machines are not available. Usually, existing models approximate the behavior of complex machines with simpler single up-single down or single up-multiple down state machines, for which analytical evaluations of continuous two-machine line models do exist (Gershwin and Schick 1980; Levantesi et al. 1999). The idea of this aggregation, as suggested in Patchong and Willaeys (2001) for a production stage featuring identical parallel machines, is described in the following. Consider a production stage with three identical machines in parallel. Each machine has failure rate, repair rate and processing rate respectively equal to p, r and μ. The stage has three operational states, namely 1, 2 and 3, counting the machines that are simultaneously operational, and one down state 0 representing the case where all machines are failed. The equivalent aggregated machine model proposed in Patchong and Willaeys (2001) features a unique aggregated operational state and one down state. The processing, failure and repair rates are adjusted to match the original complex parallel machine model behavior. As highlighted by this example, aggregation reduces the complexity of the Markov chain modeling the stage, thus easier and faster analysis is allowed. However, in doing so, accuracy is reduced, especially in the estimation of the average buffer levels and, consequently, in the estimation of the system flow time and the system WIP. Moreover, depending on the structure of the complex Markovian model representing the machine's behavior, the aggregation may be ineffective.

1.3 Contribution

As a result, at the current state of the art, several manufacturing systems architectures and features that are common in industry cannot be analyzed by multi-stage continuous flow models. Examples of such cases on the factory floor are systems featuring: (I) stages with non-identical parallel machines; (II) machines having non-exponential up and down times; (III) machines characterized by complex failure dynamics such as correlated failures of components or progressive quality degradations and wear; (IV) machines performing manufacturing operations followed by on-line sampling inspections; and (V) stages with a limited dedicated workforce or repair capacity. The method proposed in this paper has the generality to handle all the aforementioned system configurations making it possible to estimate the system performance. The method considers systems with different processing rates at different stages and generally complex stage models having different processing rates at different states in the same stage. We will show in this paper that this modeling feature provides the required generality. A two-stage system with machines featuring the same general characteristics was recently proposed in Gershwin and Tan (2009). Gershwin and Tan (2010) showed that the proposed modeling framework enables the analysis of a wide range of system models, including multiple failure mode lines, identical parallel machine lines, split/merge systems and lines with generally distributed up and down times by avoiding state aggregation. The aim of this paper is to propose a decomposition method to extend this analysis to study long lines with such general characteristics.

1.4 Outline

The remainder of this paper is organized as follows. Section 2 details the main assumptions of the production system model studied in this paper. Section 3 deals with the performance measures to be estimated and presents the core ideas of the decomposition approach. In Sects. 4, 5 and 6 the decomposition equations that allow us to calculate the performance of multi-stage lines are presented. In Sect. 7, numerical results are compared with those obtained by existing analytical methods and by simulation, showing the high accuracy and convergence speed of the proposed method. Moreover, examples showing how complex, previously uninvestigated, systems can be modeled with our approach are presented. In Sect. 8, we show that new information on the system behavior that might be relevant for system improvement and control can be gathered by applying our approach. In Sect. 9 conclusions are drawn and the future research steps are planned.

2 System model and notation

The system is formed by K machines and $K-1$ buffers of finite capacity. (We will use the terms machine and stage interchangeably.) Machines are denoted as M_k, $k=1,\dots,K$ and buffers are denoted as B_k with $k=1,\dots,K-1$. The capacity of each buffer is N_k. We define the set $[M_q, M_k]$ as the set of machines included in the portion of line bounded by M_q and M_k with $q<k$. Similarly, we define $[B_q, B_k]$ as the set of buffers included in the portion of line bounded by B_q and B_k with $q<k$.

Each M_k is characterized by I_k states, thus the general state indicator α_k assumes values $\alpha_k = 1,\dots,I_k$. The set containing all the states of M_k is called S_k. While in the generic state i, M_k produces parts at its maximal speed $\mu_i^k \geq 0$. Machines operate at their maximum rates unless they are starved or blocked. States do not need to be classified as up or down. (A state with a maximum processing rate equal to zero can be considered a down state.) Therefore, for each machine M_k the processing rate vector μ_k with dimension I_k must be specified.

Considering the generic machine M_k, the following two mutually exclusive starvation situations are possible:

1. M_k is *completely starved* by M_q with $q=1,\dots,k-1$ if all the following conditions hold:
   - M_k is in state i such that $\mu_i^k > 0$,
   - M_q is in state j such that $\mu_j^q = 0$,
   - all the buffers in the set $[B_q, B_{k-1}]$ are empty;

   In this case M_k assumes the processing rate of $\mu_j^q = 0$, i.e. it does not produce.
2. M_k is *partially starved* or *slowed down* by M_q with $q=1,\dots,k-1$ if all the following conditions hold:
   - M_k is in state i such that $\mu_i^k > 0$,
   - M_q is in state j such that $0 < \mu_j^q < \mu_i^k$,
   - all the buffers in the set $[B_q, B_{k-1}]$ are empty;

   In this case M_k assumes the processing rate of $\mu_j^q > 0$, i.e. it produces at a reduced rate.

Similar definitions hold for blocking conditions. We assume that the first machine is never starved and the last machine is never blocked. When M_k is neither starved nor blocked

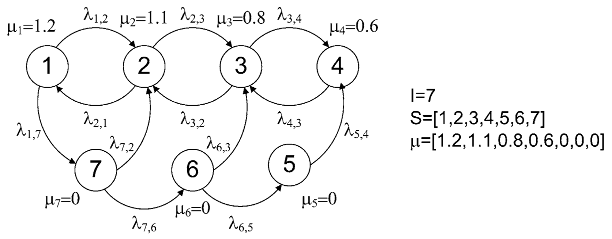


Fig. 2 State-transition diagram for the generic stage M_k

(either completely or partially) the transition time from state i to state i', with i, i' included in the set S_k, is an exponential random variable with rate $\lambda^k_{i,i'}$. When M_k is starved or blocked, the transition time from state i to state i' is also an exponential random variable with rate $\psi^k_{i,i'}$. In the case of *Operation Dependent Transitions* the rate $\psi^k_{i,i'}$ is linked to the rate $\lambda^k_{i,i'}$ through the following relation, which involves the processing rate of the machine M_q that generated the starvation (blocking) phenomenon in state j.

$$\psi^k_{i,i'} = \lambda^k_{i,i'} \frac{\mu^q_j}{\mu^k_i} \tag{1}$$

In the case of *Time Dependent Transitions* the rate $\psi^k_{i,i'}$ is equal to the rate $\lambda^k_{i,i'}$. Although the proposed decomposition method is applicable also to Time Dependent Transitions, in this paper we will consider Operation Dependent Transitions only.

Thus, the only input parameters required for each machine M_k are the transition rate matrix λ_k and the processing rate vector μ_k. A sample state transition diagram for the generic machine M_k in the system is reported in Fig. 2. For each buffer B_k, we only need to specify the buffer size N_k.

3 Outline of the decomposition method

The problem solved in this paper can be formulated in the following terms: given the system modeled in the previous section, calculate the following system performance measures:

– *Average System Production Rate, P*, defined as the average rate at which material is delivered in output by the system [parts/time].
– *Average Buffer Level, $\overline{n}_k$*, defined as the average amount of material stored in buffer B_k, $k = 1, \dots, K-1$, [parts].
– *Probability of partial and complete starvation, $Ps^{q,j}_{k,i}$* of machine M_k in state i due to machine M_q being in state j. It is defined as the probability that the machine M_k being in state i is forced to produce parts at reduced rate because its upstream buffer B_{k-1} is empty due to the fact that machine M_q with $q = 1, \dots, i-1$ is in state j, with $\mu^q_j < \mu^k_i$.
– *Probability of partial and complete blocking, $Pb^{q,j}_{k,i}$* of machine M_k in state i due to machine M_q being in state j. It is defined as the probability that the machine M_k being in state i is forced to produce parts at reduced speed because its downstream buffer B_k is full due to the fact that machine M_q with $q = i+1, \dots, K$ is in state j, with $\mu^q_j < \mu^k_i$.
– *Total probability of partial and complete starvation, Ps_k* of machine M_k, defined as the probability that the machine M_k is forced to produce at reduced rate because its upstream buffer B_k is empty.

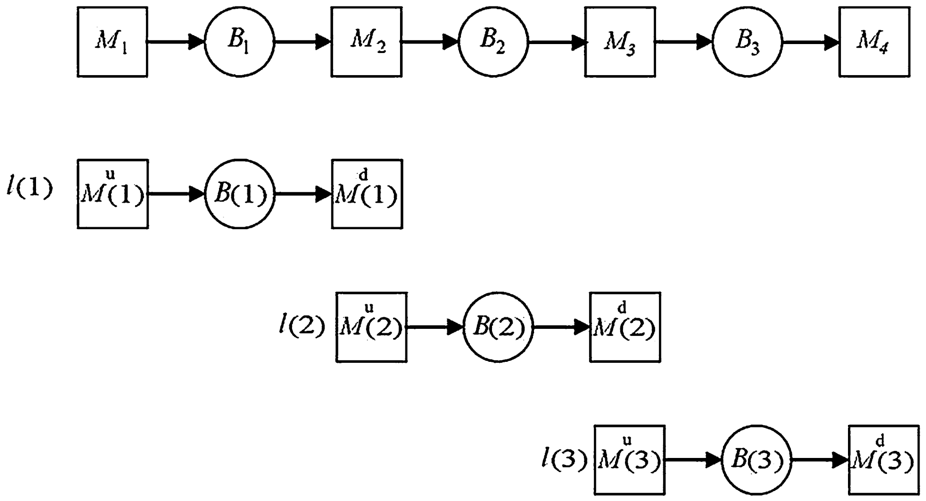


Fig. 3 Decomposition applied to a 4 machines line

– *Total probability of partial and complete blocking, Pb_k* of machine M_k, defined as the probability that the machine M_k is forced to produce at reduced rate because its downstream buffer B_k is full.

For estimating the performance of the system, a decomposition technique is developed (Gershwin 1994). The idea of the decomposition approach is to decompose the K-machine system into a set of $K-1$ two-machine one-buffer sub-systems, i.e. one for each buffer in the original system. Each building block is denoted as $l(k)$ and is characterized by one upstream pseudo-machine $M_u(k)$, one downstream pseudo-machine $M_d(k)$ and one buffer $B(k)$. Thus a line with four machines such that in Fig. 3 is decomposed into three building blocks. According to the decomposition logic, all the interruptions of the material flow entering (leaving) the buffer B_k are modeled by the pseudo-machine $M_u(k)$ $(M_d(k))$. Similarly to the decomposition method proposed in Levantesi et al. (2003) for lines with multiple failure mode machines, we assign to every pseudo-machine remote states modeling the propagation of complete starvation and blocking in the system. However, differently from the method proposed in Levantesi et al. (2003), in this paper we specifically assign to pseudo-machines one remote state for each partial starvation and blocking condition. Indeed, in Levantesi et al. (2003) partial starvation and blocking states were not considered. The effect of partial starvation and blocking was captured by calculating an average processing rate of each pseudo-machine that was derived by aggregating all relevant μ's of the machine. On the contrary, in this paper we consider in detail all possible operational states of each pseudo-machine with the corresponding processing rate μ. This modeling feature complicates the two-machine line structure but provides more accurate results also for lines with single failure and multiple failure mode machines, as it will be shown in Sect. 7.

For the problem considered in this paper, the exact solution for a two machine one buffer system was derived in Gershwin and Tan (2010). In order to model in the building blocks the behavior of the original line, decomposition equations which correctly characterize the propagation of blocking and starvation through the line are developed. For $M_u(k)$ and $M_d(k)$ of each building block $l(k)$ the following unknown parameters will be calculated in Sects. 4, 5, and 6 of this paper:

– the transition rate matrices $\lambda_u(k)$ and $\lambda_d(k)$;
– the processing rate vectors $\mu_u(k)$ and $\mu_d(k)$.

The proposed decomposition method is approximate since the transition times from local to remote states of the pseudo-machine models are considered to be exponentially distributed, while being, in fact, generally distributed. Therefore, the accuracy of the method will be investigated in Sect. 7.

4 Structure of the pseudo-machine state space

Consider the upstream pseudo machine $M_u(k)$. The dynamics of the material flow entering the buffer B_k depends directly on the processing rates of the states of M_k, if is not partially or completely starved. Otherwise, if M_k is partially or completely starved by M_q, with $q = 1, \dots, k-1$, the dynamics of the flow entering the buffer B_q depends on the processing rates of the states of M_q. Given this consideration, it is possible to assign to upstream pseudo-machines two sets of states:

– *Local States* in the set $S_u(k)$: regulate the dynamics of the material flow when the machine M_k is not partially nor completely starved.
– *Remote States* in the set $S'_u(k)$: regulate the dynamics of the material flow when the machine M_k is partially or completely starved by upstream machines M_q, $q = 1, \dots, k-1$.

The following procedure determines the number of local states, $I_u(k)$, and the number of remote states, $I'_u(k)$, for each pseudo machine $M_u(k)$. Moreover, it is used to calculate the unknown vector of processing rates for the defined state space.

Given the fact that they model the dependency of the material flow entering in B_k on the states of machine M_k, the local states in the set $S_u(k)$ are equal to the states of the corresponding machine in the original line M_k. Therefore, it is possible to write:

$$I_u(k) = I_k \tag{2}$$

$$\mu_u(k) = \mu_k \tag{3}$$

Remote states mimic the dependency of the material flow entering in B_k with the states of machine M_q in the set $[M_1, M_{k-1}]$, when M_k is partially or totally starved. Thus, they are obtained by the states of the corresponding machine in the original line M_q. However, not all the states of machines M_q, with $q = 1, \dots, k-1$ generate partial or complete starvation, due to the conditions expressed in Sect. 2. The following equation can be used to express the number of states in $S'_u(k)$:

$$I'_u(k) = \sum_{q=1}^{k-1} \sum_{j=1}^{I_q} \sum_{i=1}^{I_k} g^q_{j,i} \tag{4}$$

where the term $g^q_{j,i}$ is given by:

$$g^q_{j,i} = \begin{cases} 1, & \mu^q_j < \mu^k_i \text{ and } \forall M_t \in [M_{q+1}, M_{k-1}]\ \exists \text{ a state } z_t\text{: } \mu^q_j < \mu^t_{z_t} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

This equation accounts for the cases in which the machine M_k can be partially or completely starved by machine M_q in the set $[M_1, M_{k-1}]$. If $g^q_{j,i} = 1$ and $\mu^q_j = 0$ then a case of complete starvation of machine M_k is enumerated. Otherwise if $g^q_{j,i} = 1$ and $\mu^q_j > 0$ then a case of partial starvation of machine M_k is considered. It is worth noticing that, given equation (5), machine M_k will be subject to partial and complete starvation only in those states i in S_k for which $\mu^k_i > 0$. Consequently, the pseudo-machine $M_u(k)$ is able to enter a remote state only from those local states i in $S_u(k)$ for which $\mu^u_i(k) > 0$.

Equation (4) can be rewritten in a recursive way, considering only the local and the remote states of the upstream pseudo-machine in the previous building block $l(k-1)$:

$$I'_u(k) = \sum_{j=1}^{I_u(k-1)+I'_u(k-1)} \sum_{i=1}^{I_u(k)} g_{j,i} \tag{6}$$

where the term $g_{j,i}$ is given by:

$$g_{j,i} = \begin{cases} 1, & \mu^u_j(k-1) < \mu^u_i(k); \text{ with } j \in (S_u(k-1) \cup S'_u(k-1)) \text{ and } i \in S_u(k) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

For each $g_{j,i} = 1$ a new remote state denoted by $(j,i)_u(k)$ is assigned to the pseudo-machine $M_u(k)$ of the building block $l(k)$. The index j refers to the state of $M_u(k-1)$ that is cause of starvation for $M_d(k-1)$, i.e. cause of a remote state for the pseudo-machine $M_u(k)$ while in the local state $i \in S_u(k)$. The processing rate $\mu^u_j(k-1)$ is attributed to this new remote state. Therefore, the vector $\mu_u(k)$ is completely defined for the pseudo-machine $M_u(k)$. It is worth highlighting that, differently from what observed for the multiple failure mode decomposition proposed in Levantesi et al. (2003), the values in the vector $\mu_u(k)$ remain constant in the decomposition equations, i.e. once built the state space of pseudo-machines, they do not need to be updated.

By considering the pseudo-machine $M_d(k-1)$ the same procedure can be applied referring to complete and partial blocking conditions. For local and remote states, with $k = 2, \ldots, K-1$:

$$I_d(k-1) = I_k \tag{8}$$

$$\mu_d(k-1) = \mu_k \tag{9}$$

$$I'_d(k-1) = \sum_{j=1}^{I_d(k)+I'_d(k)} \sum_{i=1}^{I_d(k-1)} g_{i,j'} \tag{10}$$

where the term $g_{i,j'}$ is given by:

$$g_{i,j'} = \begin{cases} 1, & \mu^d_{j'}(k) < \mu^d_i(k-1); \text{ with } j' \in (S_d(k) \cup S'_d(k)) \text{ and } i \in S_d(k-1) \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

For each $g_{i,j'} = 1$ a new remote state denoted by $(i,j')_d(k-1)$ is assigned to the pseudo-machine $M_d(k-1)$ of the building block $l(k-1)$. The index j' refers to the state of $M_d(k)$ that is cause of blocking for $M_u(k)$, i.e. cause of a remote state for the pseudo-machine $M_d(k-1)$ while in the local state $i \in S_d(k-1)$. The processing rate $\mu^d_j(k)$ is attributed to this new remote state. Therefore, the vector $\mu_d(k-1)$ is completely defined for the pseudo-machine $M_d(k-1)$.

5 Coherence among building blocks $l(k)$ and $l(k-1)$

We are interested in associating steady state probabilities with the remote states of the pseudo-machine $M_u(k)$, in the case the performance of the previous two-machine line $l(k-1)$ are known. We define $\pi(0,j,i)(k)$ to be the steady state probability of having zero parts in the buffer in building block $l(k)$, when the upstream pseudo-machine is in state j and the downstream pseudo-machine is in state i. Moreover, we define $\pi(j,i)_u(k)$ to be the steady state probability associated to the remote state $(j,i)_u(k) \in S'_u(k)$ of the pseudo-machine $M_u(k)$. The following equation can be written for all the remote states $(j,i)_u(k) \in S'_u(k)$ of the pseudo-machine $M_u(k)$:

$$\pi(j,i)_u(k) = \pi(0,j,i)(k-1) + \sum_{(i,j')_d \in S'_d(k-1)} \pi(0,j,(i,j')_d)(k-1) \tag{12}$$

Indeed, by definition of remote states, the probability of finding the upstream pseudo-machine of building block $l(k)$ in a given remote state has to be equal to the probability of finding the downstream pseudo-machine of the upstream building block $l(k-1)$ in the corresponding starvation state (either complete or partial).

Similarly, for all the states $(i,j')_d(k-1) \in S'_d(k-1)$ of the pseudo-machine $M_d(k-1)$:

$$\pi(i,j')_d(k-1) = \pi(N,i,j')(k) + \sum_{(j,i)_u \in S'_u(k)} \pi(N,(j,i)_u,j')(k) \tag{13}$$

Moreover, a generalization of the *Conservation of Flow* equation, originally proposed in Gershwin (1994), needs to be derived. This equation states that the production rate must be identical in the different building blocks obtained by line decomposition. However, while dealing with general machines characterized by different operational states, this aggregated condition can be further specified. In this paper, we present for the first time a set of equations named *Generalized State-Dependent Conservation of Flow* equations. These equations guarantee that, for any operational state i of machine M_k, the output material flow rate from buffer $B(k-1)$ of the sub-system $l(k-1)$ is equal to the input material flow rate into buffer $B(k)$ of the sub-system $l(k)$. More technically, for each state i in the set S_k of the machine M_k in the original line, the upstream pseudo-machine $M_u(k)$ and the downstream pseudo-machine $M_d(k-1)$ must have the same production rate. Therefore, for the conservation of the material flow being processed by machine M_k in the generic state $i \in S_k$ the following can be written, $\forall i \in S_k$:

$$\begin{aligned}&\sum_{(j,i)_u \in S'_u(k)} \pi(j,i)_u(k)\mu^u_{j,i}(k) + \pi(i)_u(k)\mu^u_i(k)\\ &\quad= \sum_{(i,j')_d \in S'_d(k-1)} \pi(i,j')_d(k-1)\mu^d_{i,j'}(k-1) + \pi(i)_d(k-1)\mu^d_i(k-1)\end{aligned} \tag{14}$$

These Generalized State-Dependent Conservation of Flow equations will be used to derive the unknown transition rate matrix $\lambda_u(k)$ of the pseudo-machine $M_u(k)$ and $\lambda_d(k-1)$ of the pseudo-machine $M_d(k-1)$.

6 Decomposition equations for calculating $\lambda_u(k)$

Consider the transition rate matrix $\lambda_u(k)$. (The analysis of the transition rate matrix $\lambda_d(k-1)$ is reported in Appendix.) This matrix is a square matrix with size $(I_u(k)+I'_u(k)) \times$

$(I_u(k)+I'_u(k))$ which contains the transition rates governing the pseudo-machine $M_u(k)$ in its transitions among the states in the state space. It is partitioned into four matrices, depending on the type of states transition rates link. Therefore, the matrices $\lambda^u_{S,S}(k)$, $\lambda^u_{S,S'}(k)$, $\lambda^u_{S',S}(k)$ and $\lambda^u_{S',S'}(k)$ are defined. The matrix $\lambda_u(k)$ assumes the form:

$$\lambda_u(k)=\begin{vmatrix}\lambda^u_{S,S}(k) & \lambda^u_{S,S'}(k)\\ \lambda^u_{S',S}(k) & \lambda^u_{S',S'}(k)\end{vmatrix} \tag{15}$$

In the following the equations for calculating the elements of these block matrices are given.

6.1 Equations for determining the elements of $\lambda^u_{S,S}(k)$

Matrix $\lambda^u_{S,S}(k)$ is a square matrix of size $I_u(k)\times I_u(k)$. The transitions in this matrix link local states with local states. Since while in local states, the pseudo-machine $M_u(k)$ models the behavior of the original machine M_k it is possible to write:

$$\lambda^u_{i,i'}(k)=\lambda^k_{i,i'},\quad i\in S_u(k),\ i'\in S_u(k) \tag{16}$$

Therefore, $\lambda^u_{S,S}(k)$ is simply obtained by copying the elements of the matrix λ_k in the original line.

6.2 Equations for determining the elements of $\lambda^u_{S',S'}(k)$

The matrix $\lambda^u_{S',S'}(k)$ is a square matrix of size $I'_u(k)\times I'_u(k)$. The transitions in this matrix link remote states with remote states. Transitions of this type may occur for two reasons. First, when the machine which caused the partial or complete starvation, say M_q with $q<k$, makes a transition to another state and the new resulting state is again in the set $S'_u(k)$ of $M_u(k)$. This includes both the case in which, after the transition, the starvation is caused by the new state of the same machine M_q, and the case in which, after the transition, the starvation is caused by a state of a different machine which is included in the set $[M_q, M_k]$. Second, when machine M_k, working at reduced processing rate, makes a transition to another state which still is characterized by a reduced processing rate. Thus this state will be again in the set $S'_u(k)$ of $M_u(k)$. The two distinct contributions to matrix $\lambda^u_{S',S'}(k)$ are named $\lambda^{u,1}_{S',S'}(k)$ and $\lambda^{u,2}_{S',S'}(k)$.

The first contribution is obtained in the following. The transition of the machine which caused the partial or complete starvation to another state which results in $M_u(k)$ to be again in a remote state in the set $S'_u(k)$ can be expressed in terms of transitions of the upstream pseudo-machine of the building block $l(k-1)$:

$$\lambda^{u,1}_{(j,i),(j',i)}(k)=\lambda^u_{j,j'}(k-1),\quad (j,i)\in S'_u(k),(j',i)\in S'_u(k) \tag{17}$$

The second contribution can be obtained by considering transitions of the machine M_k:

$$\lambda^{u,2}_{(j,i),(j,i')}(k)=\psi^k_{i,i'},\quad (j,i)\in S'_u(k),(j,i')\in S'_u(k) \tag{18}$$

It is worth noticing that, according to (1), transition rates $\psi^k_{i,i'}$ are equal to zero in case the machine M_k is completely starved (since, in this case, μ^q_j of machine M_q which generated the starvation is zero). Therefore, the operation dependent assumption is verified (machines cannot make transitions if they are starved or blocked). Moreover, the matrix $\lambda^{u,2}_{S',S'}(k)$ will have rows of zeros in correspondence to complete starvation states. Finally, $\lambda^u_{S',S'}(k)$ is obtained by summing up both contributions:

$$\lambda^u_{S',S'}(k) = \lambda^{u,1}_{S',S'}(k) + \lambda^{u,2}_{S',S'}(k) \tag{19}$$

6.3 Equations for determining the elements of $\lambda^u_{S',S}(k)$

The matrix $\lambda^u_{S',S}(k)$ has size $I'_u(k) \times I_u(k)$. The transitions in this matrix link remote states with local states. Again, transitions of this type may occur for two reasons. They may happen when the machine which caused the partial or complete starvation makes a transition to another state and the resulting state for $M_u(k)$ is no longer in the set $S'_u(k)$. Thus the machine M_k is allowed to work at its maximum processing rate. This can be expressed in terms of transitions of the upstream pseudo-machine of the building block $l(k-1)$:

$$\lambda^{u,1}_{(j,i),i}(k) = \sum_{(j',i) not \in S'_u(k)} \lambda^u_{j,j'}(k-1), \quad (j,i) \in S'_u(k), i \in S_u(k) \tag{20}$$

The other reason for these transitions to happen is when machine M_k, working at reduced speed, makes a transition to another state which is characterized by its maximum processing rate, i.e. the machine is no longer partially starved. Thus this new state will be in the set $S_u(k)$ of $M_u(k)$:

$$\lambda^{u,2}_{(j,i),i'}(k) = \psi^k_{i,i'}, \quad (j,i) \in S'_u(k), i' \in S_u(k) \tag{21}$$

Again, it is worth noticing that, according to (1), transitions $\psi^k_{i,i'}$ are equal to zero when machine M_k is completely starved. Therefore, the matrix $\lambda^{u,2}_{S',S}(k)$ will have rows of zeros corresponding to complete starvation states. The matrix $\lambda^u_{S',S}(k)$ is obtained by summing up both contributions:

$$\lambda^u_{S',S}(k) = \lambda^{u,1}_{S',S}(k) + \lambda^{u,2}_{S',S}(k) \tag{22}$$

6.4 Equations for determining the elements of $\lambda^u_{S,S'}(k)$

Matrix $\lambda^u_{S,S'}(k)$ has size $I_u(k) \times I'_u(k)$. The transitions in this matrix link local states with remote states. They can be calculated by using the steady state equations provided in Sect. 5 and by writing balance equations to remote states of the Markov chain representing the behaviour of the pseudo-machine $M_u(k)$. In detail, for any remote state $(j,i) \in S'_u(k)$ of $M_u(k)$, at the steady-state, the entry frequency must equal the exit frequency. Moreover, by definition of the remote states, it is possible to enter remote state $(j,i) \in S'_u(k)$ only from a unique local state $i \in S_u(k)$. Therefore, the following set of balance equations can be written:

$$\lambda^u_{i,(j,i)}(k)\pi(i)_u(k) + \sum_{q \in S'_u(k)} \lambda^u_{q,(j,i)}(k)\pi(q)_u(k)$$
$$= \pi(j,i)_u(k) \left(\sum_{q \in (S_u(k) \cup S'_u(k))} \lambda^u_{(j,i),q}(k) \right) \tag{23}$$

By manipulating the previous equations this general result follows, $\forall(j,i) \in S'_u(k)$.

$$\lambda^u_{i,(j,i)}(k) = \frac{\pi(j,i)_u(k)(\sum_{q\in(S_u(k)\cup S'_u(k))}\lambda^u_{(j,i),q}(k)) - \sum_{q\in S'_u(k)}\lambda^u_{q,(j,i)}(k)\pi(q)_u(k)}{\pi(i)_u(k)} \tag{24}$$

It is worth mentioning that, under the Operation Dependent Transitions assumption, since a machine can get partially or completely starved only when producing parts, it is impossible for the pseudo-machine $M_u(k)$ to make a transition from a local state with $\mu = 0$ to a remote state. Thus the state i in (24) will refer only to local states with $\mu^u_i(k) > 0$ and the matrix $\lambda^u_{S,S'}$ will have rows of zeros corresponding to local states i with $\mu^u_i(k) = 0$.

6.5 Solution algorithm

The algorithm that makes it possible to solve decomposition equations efficiently is described in this section.

Step 1: *Building block parameter initialization.*

Decompose the original K-machine system into a set of $K-1$ two-machine lines, $l(k)$.

- For $k = 1, \dots, K-1$:
  - Use (2) and (6) for sizing the upstream pseudo-machine state space. For $k = 1$, the set $S'_u(k)$ is an empty set.
  - Use the method in Sect. 4 for populating the state spaces with the proper set of states.
  - Use equations from (16) to (22) for determining the values of the transition rates in the matrices $\lambda^u_{S,S}(k)$, $\lambda^u_{S',S'}(k)$, $\lambda^u_{S',S}(k)$. These rates are obtained from the known initial transition rate matrices.
  - Assign to the transition rates in the matrix $\lambda^u_{S,S'}(k)$ an arbitrary low starting value, i.e. 0.01.
- For $k = K-1, \dots, 1$, repeat the same procedure for $M_d(k)$:
  - Use (8) and (10) for sizing the downstream pseudo-machine state space. For $k = K-1$, the set $S'_d(k)$ is an empty set.
  - Use the method in Sect. 4 for populating the state spaces with the proper set of states.
  - Use equations from (45) to (51) for determining the values of the transition rates in the matrices $\lambda^d_{S,S}(k)$, $\lambda^d_{S',S'}(k)$, $\lambda^d_{S',S}(k)$. These rates are obtained from the known initial transition rate matrices.
  - Assign to the transition rates in the matrix $\lambda^d_{S,S'}(k)$ an arbitrary low starting value, i.e. 0.01.
- For $k = 1, \dots, K-1$: solve each building block $l(k)$ using the method proposed in Gershwin and Tan (2010) and calculate the initial values for the performance measures and state probabilities.

Step 2: *Update the unknown transition rates of pseudo-machines*

Step 2A: For $k = 2, \dots, K-1$: consider pseudo-machine $M_u(k)$.

- By using (12) update the value of the probabilities of the remote states $(j,i) \in S'_u(k)$.

- For each state $i \in S_u(k)$ calculate the probability $\pi(i)_u(k)$ by using the generalized conservation of flow (14).
- Use equations from (17) to (22) for setting the value of the transition rates in the matrices $\lambda^u_{S',S'}(k)$, $\lambda^u_{S',S}(k)$ from the building block $l(k-1)$.
- By using (24) update the values of the transition rates in the matrix $\lambda^u_{S,S'}(k)$.
- Solve building block $l(k)$ using the method proposed in Gershwin and Tan (2010) and update the values for the performance measures and state probabilities.

Step 2B: For $k = K-2, \dots, 1$: consider pseudo-machine $M_d(k)$.

- By using (13) update the value of the probabilities of the remote states $(i, j) \in S'_d(k)$.
- For each state $i \in S_d(k)$ calculate the probability $\pi(i)_d(k)$ by using the generalized conservation of flow (14).
- Use equations from (46) to (51) for setting the value of the transition rates in the matrices $\lambda^d_{S',S'}(k)$, $\lambda^d_{S',S}(k)$ from the building block $l(k+1)$.
- By using (52) update the values of the transition rates in the matrix $\lambda^d_{S,S'}(k)$.
- Solve building block $l(k)$ using the method proposed in Gershwin and Tan (2010) and update the values for the performance measures and state probabilities.

Step 3: *Convergence check*
Check the following conservation of flow equation, given a small tolerance value ϵ.

$$|P(k) = P(k+1)| \le \epsilon, \quad k = 1, \dots, K-2 \tag{25}$$

If this condition does not hold, then go back to Step 2 of this algorithm. Otherwise if this condition holds, calculate the following system performance measures:

The average system production rate

$$P = P(k) \tag{26}$$

with k arbitrarily selected in the closed interval $[1, K-1]$.

The average buffer level

$$\bar{n}_k = \bar{n}(k), \quad i = 1, \dots, K-1 \tag{27}$$

Probability of partial and complete starvation, $Ps^{q,j}_{k,i}$ of machine M_k in state i due to machine M_q being in state j

$$Ps^{q,j}_{k,i} = \pi(0, j, i)(k-1) + \sum_{(i,j')_d \in S'_d(k-1)} \pi(0, j, (i, j')_d)(k-1), \quad k = 2, \dots, K. \tag{28}$$

The total probability of partial and complete starvation

$$Ps_k = \sum_{q=1}^{k-1} \sum_{j=1}^{I_q} \sum_{i=1}^{I_k} Ps^{q,j}_{k,i}, \quad k = 2, \dots, K. \tag{29}$$

Probability of partial and complete blocking, $Pb_{k,i}^{q,j}$ of machine M_k in state i due to machine M_q being in state j'

$$Pb_{k,i}^{q,j'} = \pi(N_k, i, j')(k) + \sum_{(j,i)_u \in S_u'(k)} \pi(N_k, (j,i)_u, j')(k), \quad k = 1, \dots, K-1. \tag{30}$$

The total probability of partial and complete blocking

$$Pb_k = \sum_{q=k+1}^{K} \sum_{j'=1}^{I_q} \sum_{i=1}^{I_k} Pb_{k,i}^{q,j'}, \quad k = 1, \dots, K-1. \tag{31}$$

7 Numerical results

The accuracy of approximate analytical methods based on decomposition approaches has been traditionally tested by comparing the outputs with those of simulation. The accuracy of the proposed method is investigated in this paper by following two different approaches. First, the results provided by the method are compared, where possible, with numerical cases for which simulation results exist in literature. This approach also allows us to compare the accuracy of our method with the accuracy of existing decomposition based methods. Second, for more complex configurations of production lines for which decomposition methods do not exist in the literature, the results are compared with those of discrete material simulation models developed by using Rockwell ARENA. It must be pointed out that part of the errors with the simulation outputs may be due to the modeling differences rather than the approximation of the solution. With R simulation replicates, the percentage error of the analytical method X in the estimation of the system production rate P versus simulation is estimated by using the following equations:

$$\epsilon(X) = \frac{P^X - \widehat{P}^{sim}}{\widehat{P}^{sim}} 100, \quad \widehat{P}^{sim} = \frac{\sum_{r=1}^{R} P_r^{sim}}{R} \tag{32}$$

Moreover, for each buffer B_k the percentage error in the evaluation of the average level, $\overline{n}_k$ is calculated with the following equation:

$$\epsilon = \frac{\overline{n_k}^X - \widehat{\overline{n_k}}^{sim}}{N_k} 100, \quad \widehat{\overline{n_k}}^{sim} = \frac{\sum_{r=1}^{R} \overline{n_{k,r}}^{sim}}{R} \tag{33}$$

7.1 Accuracy testing: comparison with existing methods

Given the wide range of systems that can be modeled and analyzed with the proposed approach, in this section we concentrate on a few system architectures and machine models, making it possible to compare the results provided by our method with existing techniques and simulation.

7.1.1 Systems with single up-single down state machines

The first set of systems considered in this analysis is asynchronous production lines with one up state and one down state per machine. Several methods can be found in literature to deal with similar systems. In Levantesi et al. (2003) a set of experiments was proposed in

Table 1 Data for the test systems 1–6 evaluated in Levantesi et al. (2003)

	M_1	M_2	M_3	M_4	M_5
p	0.0125	0.005	0.02	0.01	0.01
r	0.2	0.05	0.2	0.1	0.08
N	15	20	10	15	/
μ(case 1)	1.111	1.667	1	1.428	1.25
μ(case 2)	1.25	1.111	1.667	1	1.428
μ(case 3)	1.428	1.25	1.111	1.667	1
μ(case 4)	1	1.428	1.25	1.111	1.667
μ(case 5)	1.667	1	1.428	1.25	1.111
μ(case 6)	2	10	3	1	5

which three of the most accurate methods were compared. The first method is the *ADDX* method developed in Burman (1995), which extended the continuous time decomposition developed in Gershwin (1994). The second method is the *RSA/CTA/GE* homogenization technique proposed in Dallery and Le Bihan (1997), and the third method is the decomposition approach proposed in Levantesi et al. (2003) called *LMT*. According to the results published in Levantesi et al. (2003), the latter was proved to perform best in terms of accuracy.

In this paragraph, we compare the *CG* method presented in this paper with the *LMT* method and we also report the simulation output as published in Levantesi et al. (2003). Table 2 presents the results of the analysis for cases 1–6 of that paper. For completeness, the data of these cases are also reported in Table 1. For each machine M_k, the failure rate p_k and the repair rate r_k are reported. The equivalent λ_k matrix is given by:

$$\lambda_k = \begin{vmatrix} -p_k & p_k \\ r_k & -r_k \end{vmatrix} \tag{34}$$

where the states are ordered is such a way to have the processing rate vector of M_k equal to $[\mu_k, 0]$. It is worth highlighting that the *LMT* and the *CG* methods provide different results for lines with single up-single down state machines. Indeed, in the *CG* method we specifically assign to pseudo-machines one remote state with its associated processing rate μ for each partial starvation and blocking condition, while in the *LMT* the effect of partial starvation and blocking is captured by calculating an average processing rate of each pseudo-machine, that is obtained by aggregating all relevant μ's of the machine. Therefore, the proposed *CG* method remains different from the *LMT* method also for lines where machine have a single operational state.

As a summary of the results, the largest absolute error in the estimation of the production rate of the *CG* method is 0.75% while for the *LMT* method it is 1.508%. In 25 of the 30 performance measures evaluated, the *CG* method showed better accuracy. Of particular interest is the accuracy in the buffer level estimation. In literature it is well known than usually analytical methods lose accuracy while evaluating the average buffer levels. In these six cases, the maximum error of the *CG* method in estimating the average buffer level is 4.53% but in 75% of the cases the error is lower than 1.5%. Moreover, the *CG* method provides more accurate results than the *LMT* method in the estimation of the average buffer levels in 84% of the cases.

Table 2 Results for the test systems 1–6 evaluated in Levantesi et al. (2003)

	Method	P	$\overline{n}_1$	$\overline{n}_2$	$\overline{n}_3$	$\overline{n}_4$
Case 1	Sim	0.857	12.1859	18.6721	1.1896	2.6398
	LMT	0.859	13.22	18.95	1.06	2.07
	CG	0.85809	11.946	18.618	1.1042	2.4878
	$\epsilon(LMT)$,%	0.23337	6.894	1.3895	−1.296	−3.7986
	$\epsilon(CG)$, %	0.1272	−1.599	−0.2705	−0.854	−1.019
Case 2	Sim	0.8582	13.9252	15.0433	8.8399	1.3212
	LMT	0.858	14.09	12.95	9.11	1.32
	CG	0.8603	13.954	15.1	9.1302	1.3258
	$\epsilon(LMT)$, %	−0.0233	1.098	−10.4665	2.701	−0.008
	$\epsilon(CG)$, %	0.2447	0.192	0.2835	2.903	0.036
Case 3	Sim	0.8573	14.3241	17.6127	5.3609	11.9746
	LMT	0.861	14.43	17.49	2.87	12.97
	CG	0.8606	14.345	17.721	5.1168	12.173
	$\epsilon(LMT)$, %	0.4316	0.706	−0.6135	−24.909	6.636
	$\epsilon(CG)$, %	0.3826	0.1393	0.5415	−2.441	1.3226
Case 4	Sim	0.8938	2.8562	8.0653	5.2313	1.0655
	LMT	0.899	2.16	9.65	6.5	1.05
	CG	0.89852	2.2772	7.9159	5.3552	1.0629
	$\epsilon(LMT)$, %	0.58178	−4.6413	7.9235	12.687	−0.1033
	$\epsilon(CG)$, %	0.52808	−3.86	−0.747	1.239	−0.01733
Case 5	Sim	0.8748	14.6496	4.8799	4.3628	6.9768
	LMT	0.888	14.64	3.12	3.24	7.49
	CG	0.88146	14.645	3.9722	4.241	7.3645
	$\epsilon(LMT)$, %	1.50891	−0.064	−8.7995	−11.228	3.4213
	$\epsilon(CG)$, %	0.76131	−0.0306	−4.5385	−1.218	2.5846
Case 6	Sim	0.8978	14.6339	19.8579	9.8007	0.2153
	LMT	0.898	14.75	19.88	9.8	0.215
	CG	0.89796	14.652	19.868	9.7999	0.21471
	$\epsilon(LMT)$, %	0.02227	0.774	0.1105	−0.007	−0.002
	$\epsilon(CG)$, %	0.0178	0.1206	0.0505	−0.008	−0.00393

7.1.2 Systems with single up-multiple down state machines

The second set of systems considered in this analysis is asynchronous production lines with single up state and multiple down states per machine. This model is also known in literature as *multiple failure modes model*. In Levantesi et al. (2003) a set of experiments was published to test the accuracy of the multiple failure modes method developed in that paper. In the following, the accuracy of the CG method while studying multiple failure mode lines is

Table 3 Results for the test systems $E1$–$E4$ evaluated in Levantesi et al. (2003)

	P^{sim}	P^{LMT}	P^{CG}	$\epsilon(LMT), \%$	$\epsilon(CG), \%$
Case 1	0.649	0.648	0.64916	−0.154	0.024
Case 2	0.58	0.581	0.58013	0.172	0.022
Case 3	0.783	0.78	0.7868	−0.383	0.489
Case 4	0.605	0.606	0.6058	0.1652	0.1454

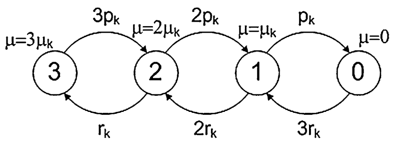


Fig. 4 State transition diagram for stage M_k with three identical parallel machines

analyzed. The λ_k matrix that models the behavior of a machine that is subject to F different causes of failure is given by:

$$\lambda_k = \left| \begin{array}{ccccc} -\sum_{i=1}^{F} p_{k,i} & p_{k,1} & p_{k,2} & \cdots & p_{k,F} \\ r_{k,1} & -r_{k,1} & 0 & \cdots & 0 \\ r_{k,2} & 0 & -r_{k,2} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{k,F} & 0 & 0 & \cdots & -r_{k,F} \end{array} \right| \tag{35}$$

where the states are ordered is such a way to have the processing rate vector of M_k equal to $[\mu_k, 0, 0, \ldots, 0]$.

Four test cases are considered, called $E1$–$E4$ in Levantesi et al. (2003). The results are shown in Table 3. As it can be noticed, also for multiple failure modes lines the CG method is very accurate (maximum error equal to 0.489% in these four cases).

7.1.3 Systems with multiple identical parallel machines

The third set of systems considered in this paper is flow lines with production stages featuring parallel machines. Such systems were studied by Burman (1995) and Patchong and Willaeys (2001). Our method allows to model multiple identical machines as a unique stage with multiple up-states (Gershwin and Tan 2010). Stage k is composed of m_k multiple identical machines in parallel. Each machine in stage k may fail with rate p_k, may be repaired with rate r_k and processes material at rate μ_k. In the framework of our model, stage k has $m_k + 1$ states. In state $j = 0, \ldots, m_k$, j machines are simultaneously operational: thus the entire stage processes material at rate $j\mu_k$. Accordingly, the possible transitions for stage M_k are:

- from state j to state $j - 1$ with rate jp_k, for $j = 1, \ldots, m_k$
- from state j to state $j + 1$ with rate $(m_k - j)r_k$, for $j = 0, \ldots, m_k - 1$.

Table 4 Data for the test systems evaluated in Burman (1995) and Patchong and Willaeys (2001)

	r	M_1	M_2	M_3		r	M_1	M_2	M_3
		0.1	0.1	0.1			0.1	0.1	0.1
Case 1	m	1	2	1	Case 2	m	1	2	1
	p	0.01	0.01	0.01		p	0.01	0.01	0.01
	μ	1	2	1		μ	1	1	1
	N	10	10			N	10	10	
Case 3	m	1	2	1	Case 4	m	1	5	1
	p	0.01	0.12	0.01		p	0.01	0.01	0.01
	μ	1	2	1		μ	1	5	1
	N	10	10			N	10	10	
Case 5	m	1	5	1	Case 6	m	1	2	1
	p	0.01	0.01	0.01		p	0.01	0.01	0.01
	μ	1	1	1		μ	1	2	1
	N	10	10			N	1	1	
Case 7	m	1	2	1	Case 8	m	1	2	1
	p	0.01	0.01	0.01		p	0.01	0.12	0.01
	μ	1	1	1		μ	1	2	1
	N	1	1			N	1	1	

Figure 4 represents the state-transition diagram of stage M_k featuring three identical machines in parallel. The transition matrix λ_k is given by:

$$\lambda_k = \begin{vmatrix} -3r_k & 3r_k & 0 & 0 \\ p_k & -p_k - 2r_k & 2r_k & 0 \\ 0 & 2p_k & -2p_k - r_k & r_k \\ 0 & 0 & 3p_k & -3p_k \end{vmatrix} \tag{36}$$

where the states are ordered as $[0, 1, 2, 3]$. The processing rate vector of M_k is equal to $[0, \mu_k, 2\mu_k, 3\mu_k]$.

To test the accuracy of our method in studying these systems we considered cases 1, 3, 5, 7, and 9 reported in Patchong and Willaeys (2001) and cases 46, 52, and 58 reported in Burman (1995). The data of the eight test cases are reported in Table 4. They involve systems with three stages and up to five parallel machines. We compare our results with simulation to estimate the error; then we compare our errors with those of the other two methods proposed in Burman (1995) and Patchong and Willaeys (2001). The results are reported in Table 5.

Summarizing, in 19 of the 24 performance measures evaluated, the *CG* method has shown the lowest absolute error. Specifically, in all eight cases our method performed best while estimating the production rate. The maximum absolute error on the production rate estimation is 1.05% while for the *Pw* method it is 10.05% and for the *Bur* method it is 12.72%. Concerning the average buffer level estimation, in 81.25% of cases our error is lower than 2%. Moreover, the maximum absolute error on the buffer level estimation is 4.342% for the *CG* method, while for the *Pw* method it is 5.51% and for the *Bur* method it is 10.2%.

Table 5 Results for test systems evaluated in Burman (1995) and Patchong and Willaeys (2001)

	Method	P	$\overline{n}_1$	$\overline{n}_2$
Case 1	Sim	0.872	3.717	6.382
	PW	0.863	3.567	6.647
	Bur	0.861	3.437	6.543
	CG	0.87398	3.7641	6.2359
	$\epsilon(PW)$, %	−1.0321	−1.5	2.65
	$\epsilon(Bur)$, %	−1.261	−2.8	1.61
	$\epsilon(CG)$, %	0.227	0.471	−1.461
Case 2	Sim	0.83	6.672	3.319
	PW	0.832	6.54	3.265
	Bur	0.83	6.603	3.397
	CG	0.83	6.7055	3.2945
	$\epsilon(PW)$, %	0.2409	−1.32	−0.54
	$\epsilon(Bur)$, %	0	−0.69	0.78
	$\epsilon(CG)$, %	0	0.335	−0.245
Case 3	Sim	0.756	5.803	4.215
	PW	0.68	5.275	4.766
	Bur	0.728	5.393	4.598
	CG	0.75279	5.8533	4.1467
	$\epsilon(PW)$, %	−10.052	−5.28	5.51
	$\epsilon(Bur)$, %	−3.703	−4.1	3.83
	$\epsilon(CG)$, %	−0.424	0.503	−0.683
Case 4	Sim	0.884	3.502	6.794
	PW	0.873	3.495	6.467
	CG	0.8758	3.7402	6.3598
	$\epsilon(PW)$, %	−1.244	−0.07	−3.27
	$\epsilon(CG)$, %	−0.921	2.382	4.342
Case 5	Sim	0.847	7.054	3.549
	PW	0.838	6.869	3.162
	CG	0.8381	7.1094	3.8906
	$\epsilon(PW)$, %	−1.062	−1.85	−3.87
	$\epsilon(CG)$, %	−1.05	0.554	3.416
Case 6	Sim	0.838	0.469	0.528
	Bur	0.811	0.381	0.618
	CG	0.83756	0.4550	0.5450
	$\epsilon(Bur)$, %	3.221	−8.8	9
	$\epsilon(CG)$, %	−0.0525	−1.404	1.704
Case 7	Sim	0.781	0.726	0.275
	Bur	0.778	0.72	0.28
	CG	0.77933	0.7355	0.2645
	$\epsilon(Bur)$, %	0.3841	−0.6	0.5
	$\epsilon(CG)$, %	−0.2138	0.951	−1.051
Case 8	Sim	0.676	0.554	0.447
	Bur	0.59	0.452	0.549
	CG	0.67098	0.5525	0.4475
	$\epsilon(Bur)$, %	12.721	−10.2	10.2
	$\epsilon(CG)$, %	−0.742	−0.154	0.054

7.2 Accuracy testing: previously uninvestigated systems

In this section, the accuracy of the proposed decomposition method is tested while estimating the performance of complex systems that were never studied with existing approaches. These include systems with generally distributed up and down times, systems with stages composed of multiple non-identical parallel machines, systems where machines are subject to aging and go through a set of degrading states characterized by increasing failure rates and decreasing yield. For each test case, five replicates of the simulation experiment are carried out. Every replicate is 1000000 time units long, with a warm up period of 100000 time units.

First, the results of an extensive experimental campaign over a set of 100 randomly generated test cases are synthetically reported to explore the accuracy of the method over a wide set of significant cases. Second, detailed results are reported and discussed for the different system architectures to show how they can be modeled under the proposed framework and to enable the interested reader to replicate experimental results.

7.2.1 Randomly generated cases

In order to limit the performance evaluation only to system configurations that are relevant in practice, the parameters of the test lines have been randomly selected, within the following ranges:

- Number of machines in the system: [3:8];
- Stage efficiency in isolation: [0.8:0.999];
- Stage production rate in isolation: [0.8:1.3];
- Stage mean down time mt^d: [1:200];
- Stage mean up time mt^u: [1:200];
- Stage scv^d of down times: [0:5];
- Stage scv^u of up times: [0:5].

Moreover, as suggested in Gershwin (2011), a set of constraints, whose generation is driven by manufacturing system engineering knowledge, has been designed that links the parameters of different stages in the line to guarantee the analysis of realistic system configurations. Specifically, the buffer capacities have been constrained to be always included between 0.5 times the smallest mean down time and 4 times the largest mean down time of the machines in the system. This is because a buffer which is sufficiently small will not have a significantly different effect on the system performance than a buffer of size zero, and a buffer which is sufficiently large will be uneconomical. Moreover, the mean repair time of each machine is constrained to be not more than 4 times that of any other machine. The random case generation algorithm we developed applies filters that reject configurations whose parameters violate these constraints. In this way, 100 random test cases have been generated and then evaluated by simulation and the developed decomposition method.

A summary of the results for the 100 test cases is shown in Table 6. The table highlights that the method has good accuracy properties in the estimation of the average system production rate over a large set of test cases. Indeed, the error in the estimation of the average production rate of the system is always lower than 3%. As can be seen, the accuracy of the method slightly decreases but is still acceptable while estimating the average buffer levels (absolute errors lower than 3% in the 93% of the cases). The maximum error on average buffer level is around 5.5%, which is still reasonable for practical purposes. In the remaining part of this section specific test cases for different system architectures are presented in detail.

Table 6 Summary of the results from 100 test cases

	Average $\|err\|\%$	Max $\|err\|\%$	% cases with $\|err\|\% < 1\%$	% cases with $\|err\|\% < 2\%$	% cases with $\|err\|\% < 3\%$
E	0.941	2.93	65	90	100
$\overline{n}$	1.37	5.35	52	85	93

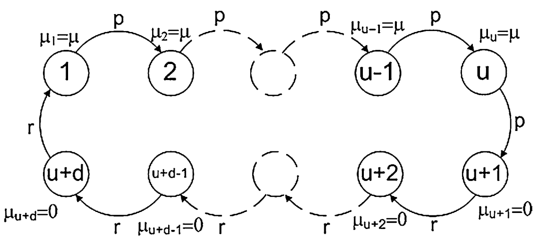
7.2.2 Phase-type distributed up and down times

Most of the models developed to study unreliable multi-stage flow lines are based on the assumptions of exponential up and down times. However, while times to machine failure can be often modeled using exponential distributions with acceptable accuracy, times to repair are very rarely observed to be exponentially distributed in actual systems (Buzacott and Hanifin 1978; Inman 1999). This feature limits the applicability of existing approximate analytical methods to real production lines. Therefore, analyzing the performance of multi-stage systems with phase-type distributed up and down times is of both practical and scientific interest. Phase type distributions (PH) have been widely used in the literature to model generally distributed events (Altiok 1985). Both *moment matching* (Osogami and Harchol-Balter 2006) and *fitting* (Horvath 2003) methods have been proposed to set the parameters characterizing the stages of a PH distribution, to model general distributions. *Moment Matching Methods* find the minimal number of phases and the characteristic parameters of a PH distribution that are needed to match the first z moments of a given distribution. The goal of *fitting* methods is in some sense broader, since they aim at approximating to a specified accuracy the shape of an input distribution using a PH distribution. For practical use, a tool for approximating general distributions by continuous PH distributions is presented in Horvath and Telek (2002).

In the literature, only two-machine line models including machines characterized by phase-type distributed failure and repair times are available (Altiok and Ozdogru 2003; Gershwin and Tan 2010). Since our method deals with machines modeled as generally complex continuous time Markov chains, PH distributed up and down times can be handled. In order to test the accuracy of our approach, in this paper we only focus on Erlang and Cox-2 distributed events. Additional results for other classes of continuous PH distributions will be reported in future works.

Erlang distributed up and down times Erlang distributions are a specific class of PH distributions, suitable to model events characterized by small square coefficient of variation ($CV^2 \leq 1$). Indeed, it is known that the minimum coefficient of variation of the continuous PH family depends only on the order n and is characteristic of the Erlang distribution (Aldous and Shepp 1987). The machines considered in this analysis have Erlang distributed up and down times. Specifically, the failure time of machine M_k is an Erlang random variable with u_k stages. The expected time to failure is $mt_k^u = u_k/p_k$ and its square coefficient of variation is $scv_k^u = 1/u_k$. Similarly, the repair time of machine M_k is an Erlang random variable with d_k stages. The expected time to repair is $mt_k^d = d_k/r_k$ and its square coefficient of variation is $scv_k^d = 1/d_k$. The state transition diagram of a machine under this failure-repair mechanism is shown in Fig. 5. The state of M_k are indexed from 1 to $u_k + d_k$ and ordered such that states $1, \dots, u_k$ are operational states and states $u_k + 1, \dots, u_k + d_k$ are down states. The processing rates for machine M_k is μ_k in every operational state and 0 in every down state. The following transitions are possible for M_k:

Fig. 5 State transition diagram for the generic machine featuring Erlang up and down times



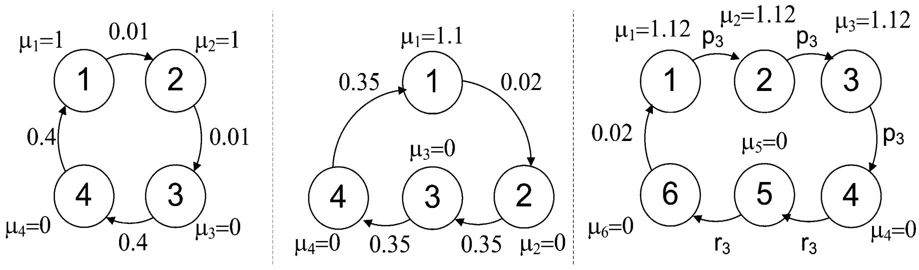


Fig. 6 State transition diagrams and data for Erlang up-down machines of Table 8

Table 7 Aggregated measures for the machines modeled in Fig. 6

k	u_k	mt_k^u	scv_k^u	d_k	mt_k^d	scv_k^d	e_k	P_k
1	2	200	0.5	2	5	0.5	0.975	0.975
2	1	50	1	3	8.57	0.333	0.853	0.939
$3-A$	3	600	0.333	3	150	0.333	0.8	0.896
$3-B$	3	300	0.333	3	150	0.333	0.667	0.746
$3-C$	3	600	0.333	3	30	0.333	0.952	1.066
$3-D$	3	300	0.333	3	30	0.333	0.909	1.018

– From state j to state $j+1$ with rate p_k, for $j = 1, \dots, u_k$.
– From state j to state $j+1$ with rate r_k, for $j = u_k + 1, \dots, u_k + d_k$.
– From state $u_k + d_k$ to state 1 with rate r_k.

In the test cases we consider three machine lines with machines having the state transition diagrams as in Fig. 6. Eight cases are obtained in the following way. We consider as fixed the parameters of machines M_1 and M_2, while we change the up time of machine M_3 by varying the rate p_3, that assumes values $0.005, 0.01$, as well as the down time of machine M_3 by varying the rate r_3, that assumes values $0.02, 0.1$. Moreover, we consider two distributions of buffers, respectively equal to $N = [2, 2]$ and $N = [10, 10]$. The combination of these features generates eight test systems. In Table 7 the performance of the machines in isolation are reported (e_k is the efficiency in isolation and P_k is the production rate in isolation of machine M_k).

Results for the eight test cases are reported in Table 8. The method provides accurate results when compared to simulation for this class of systems. The maximum error of the CG method while estimating the average production rate of the system is 1.323% and in most of the cases the error is below 1%. The maximum error in the estimation of the buffer

Table 8 Results for test systems with machines having Erlang distributed up and down times

	Method	P	$\overline{n}_1$	$\overline{n}_2$
Case 1	Sim	0.7214	0.85	0.297
$N = [2, 2]$	95% C.I	±0.013	±0.009	±0.005
$p_3 = 0.005$	CG	0.73101	0.7671	0.3473
$r_3 = 0.02$	$\epsilon(CG)$,%	1.323	−4.15	2.515
Case 2	Sim	0.764	5.577	2.46
$N = [10, 10]$	95% C.I	±0.009	±0.078	±0.061
$p_3 = 0.005$	CG	0.7681	5.8442	2.1554
$r_3 = 0.02$	$\epsilon(CG)$,%	0.523	2.628	−3.1
Case 3	Sim	0.6238	1.05	0.523
$N = [2, 2]$	95% C.I	±0.008	±0.012	±0.007
$p_3 = 0.01$	CG	0.6306	0.9517	0.6006
$r_3 = 0.02$	$\epsilon(CG)$,%	1.102	−4.915	3.88
Case 4	Sim	0.6647	6.32	3.97
$N = [10, 10]$	95% C.I	±0.01	±0.055	±0.128
$p_3 = 0.01$	CG	0.6639	6.6129	3.7369
$r_3 = 0.02$	$\epsilon(CG)$,%	−0.125	2.9	−2.4
Case 5	Sim	0.832	0.552	0.167
$N = [2, 2]$	95% C.I	±0.004	±0.006	±0.005
$p_3 = 0.005$	CG	0.8407	0.5819	0.0991
$r_3 = 0.1$	$\epsilon(CG)$,%	1.045	1.495	−3.395
Case 6	Sim	0.8836	5.02	1.16
$N = [10, 10]$	95% C.I	±0.002	±0.04	±0.035
$p_3 = 0.005$	CG	0.8891	5.1312	0.8751
$r_3 = 0.1$	$\epsilon(CG)$,%	0.621	1.1	−2.84
Case 7	Sim	0.806	0.592	0.27
$N = [2, 2]$	95% C.I	±0.003	±0.005	±0.004
$p_3 = 0.01$	CG	0.8138	0.6467	0.1936
$r_3 = 0.1$	$\epsilon(CG)$,%	0.967	2.735	−3.82
Case 8	Sim	0.8639	5.21	1.93
$N = [10, 10]$	95% C.I	±0.004	±0.044	±0.06
$p_3 = 0.01$	CG	0.8683	5.4698	1.7217
$r_3 = 0.1$	$\epsilon(CG)$,%	0.5	2.5	−2.1

level is 4.915% and in 14 over the total 16 estimated levels the error is below 4%. It can also be noticed that the error consistently decreases for increased buffer sizes, as normally observed in decomposition based methods (Gershwin and Dallery 1992), while it seems to be insensitive to the machine's mean up and down times.

Cox-2 distributed up times and down times Cox-2 distributions are a class of continuous PH distribution with two-phases, suitable to model events characterized by a large square

Fig. 7 State transition diagram for the generic machine featuring balanced mean Cox-2 distributed up and down times

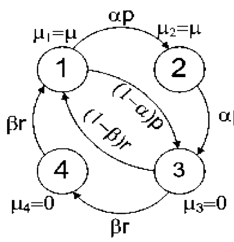


Table 9 Data for systems with Cox-2 up and down times

k	mt_k^u	scv_k^u	α_k	p_k	mt_k^d	scv_k^d	β_k	r_k	e_k	μ_k	P_k
1	200	1.5	0.333	0.01	30	0.6	0.833	0.066	0.869	1.035	0.9
2	180	1.35	0.37	0.011	26	0.75	0.667	0.077	0.873	1.03	0.9
3	160	1.2	0.416	0.0125	22	0.9	0.555	0.0909	0.879	1.024	0.9
4	140	1.05	0.476	0.014	18	1.05	0.476	0.111	0.886	1.016	0.9
5	120	0.9	0.555	0.0166	14	1.2	0.417	0.143	0.895	1.005	0.9
6	100	0.75	0.666	0.02	10	1.35	0.37	0.2	0.909	0.999	0.9
7	80	0.6	0.833	0.025	6	1.5	0.333	0.333	0.930	0.967	0.9

coefficient of variation ($CV^2 \geq 0.5$). In this set of experiments we consider machines that have Cox-2 distributed up times and down times. The Cox-2 distribution is characterized by three parameters: the probability of going through two phases, α; the rate at which the first phase is left, p_1; and the rate at which the second phase is left, p_2. The mean of the distribution is $mt = 1/p_1 + \alpha/p_2$. If $1/p_1 = \alpha/p_2 = 1/p$ the so called balanced mean Cox-2 distribution is obtained (Buzacott and Shantikumar 1993). The mean of this distribution is $mt = 2/p$ and the square coefficient of variation is $scv = 1/2\alpha$. Therefore, only two parameters are required. In the following experiments, machine's up and down times are distributed according to the balanced mean Cox-2 distribution.

Specifically, the expected time to failure of machine M_k is $mt_k^u = 2/p_k$ and its square coefficient of variation is $scv_k^u = 1/2\alpha_k$. Similarly, the expected time to repair is $mt_k^d = 2/r_k$ and its square coefficient of variation is $scv_k^d = 1/2\beta_k$. The state transition diagram of a machine under this failure-repair mechanism is shown in Fig. 7. The states of M_k are indexed from 1 to 4 and ordered such that states $1, 2$ are operational states and states $3, 4$ are down states. The processing rates for machine M_k is μ_k in every operational state and 0 in every down state.

The lines we tested have between 3 and 7 machines. Machine parameters are reported in Table 9. Moreover, we consider four distributions of buffers, respectively equal to $\overline{N1} = [2, 2, 2, 2, 2, 2]$, $\overline{N2} = [2, 4, 6, 8, 10, 12]$, $\overline{N3} = [12, 10, 8, 6, 4, 2]$, $\overline{N4} = [12, 12, 12, 12, 12, 12]$. The test system with $K + 1$ machines is obtained by adding to the system with K machines the buffer B_K and the machine M_{K+1} at the end of the line. The combination of these features generates twenty test systems.

Results for the twenty test cases are reported in Table 10. Due to space limitations, we report detailed data only for the estimate of the average production rate. The maximum error of the CG method while estimating the average production rate of the system is 2.28%, corresponding to the system with 7 machines and small buffers. In 18 over the 20 cases the error is below 2%. The error is slightly affected by the number of machines in the line, while it is confirmed that systems with smaller buffers generally involve larger errors.

Table 10 Results for test systems with machines having Cox-2 distributed up and down times

K	Method	$P(\overline{N1})$	$P(\overline{N2})$	$P(\overline{N3})$	$P(\overline{N4})$
3	Sim	0.719	0.721	0.759	0.761
	95% C.I	±0.021	±0.011	±0.028	±0.023
	CG	0.726	0.73	0.7565	0.7594
	$\epsilon(CG), \%$	0.973	1.248	−0.329	−0.21
4	Sim	0.663	0.675	0.721	0.727
	95% C.I	±0.028	±0.016	±0.04	±0.02
	CG	0.6705	0.6841	0.7154	0.7246
	$\epsilon(CG), \%$	1.131	1.348	−0.776	−0.33
5	Sim	0.617	0.649	0.68	0.71
	95% C.I	±0.018	±0.009	±0.037	±0.0189
	CG	0.6294	0.6572	0.6857	0.7037
	$\epsilon(CG), \%$	2.009	1.263	0.838	−0.887
6	Sim	0.589	0.634	0.655	0.697
	95% C.I	±0.03	±0.02	±0.018	±0.036
	CG	0.6005	0.6449	0.6636	0.6931
	$\epsilon(CG), \%$	1.952	1.719	1.312	−0.559
7	Sim	0.57	0.63	0.639	0.685
	95% C.I	±0.034	±0.026	±0.041	±0.012
	CG	0.583	0.6417	0.6481	0.6893
	$\epsilon(CG), \%$	2.28	1.857	1.4241	0.627

7.2.3 Systems with multiple non-identical parallel machines

We analyze the performance of systems having stages featuring multiple non-identical parallel machines. Two-machine systems with similar features were also studied in Tan (2001) and Helber and Jusic (2004). As in the case of multiple identical machines, our method makes it possible to model multiple non-identical machines as a unique stage with multiple up-states (Gershwin and Tan 2010). Stage k is composed of m_k machines in parallel. Each machine in stage k may fail with rate $p_{k,i}$, may be repaired from failures with rate $r_{k,i}$ and processes material at rate $\mu_{k,i}$. In the framework of our model, stage k has $S_k = 2^{m_k}$ states. Each state j is characterized by a vector of state indicators $\alpha_{k,i}$ which assume values 1 if the machine i is up and 0 if machine i is down. Considering a stage with 2 non-identical machines, the possible states of the stage are $[11, 10, 01, 00]$. The processing rate vector is $\mu_k = [\mu_{k,1} + \mu_{k,2}, \mu_{k,1}, \mu_{k,2}, 0]$.

Figure 8 represents the state transitions diagram of stage k with 2 non-identical machines. The transition rate matrix λ_k is given by:

$$\lambda_k = \begin{vmatrix} -p_{k,1} - p_{k,2} & p_{k,2} & p_{k,1} & 0 \\ r_{k,2} & -p_{k,1} - r_{k,2} & 0 & p_{k,1} \\ r_{k,1} & 0 & -p_{k,2} - r_{k,1} & p_{k,2} \\ 0 & r_{k,1} & r_{k,2} & -r_{k,1} - r_{k,2} \end{vmatrix} \qquad (37)$$

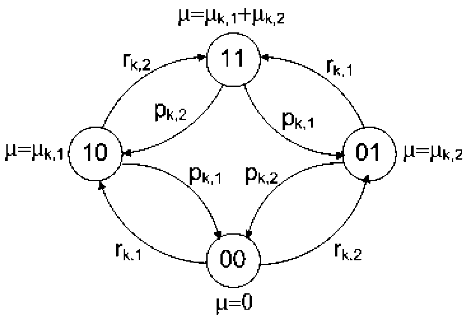


Fig. 8 State transition diagram for stage M_k with 2 non-identical parallel machines

The production rate in isolation of the stage can be calculated as:

$$P_k = \sum_{i=1}^{m_k} e_{k,i}\mu_{k,i} = \sum_{i=1}^{m_k} \frac{r_{k,i}}{p_{k,i} + r_{k,i}}\mu_{k,i} \tag{38}$$

In Table 11, the data for eight test systems, having three or four stages and one through three machines per stage, are reported. Results are shown in Table 12. The method proves to be accurate for this set of test cases. The maximum error of the CG method while estimating the average production rate of the system is 1.236%, corresponding to the system with four stages, seven machines in total, and small buffers. Concerning the average buffer level estimation, the maximum error is 3.77%, and in 14 of the 19 estimated levels, the error is below 3%. The accuracy of the method appears insensitive to the number of stages and the number of machines per stage.

7.2.4 Systems with deteriorating machines

Systems with machines that are subject to aging and may deteriorate over time are analyzed in this section. Similar machine models were previously considered in Dimitrakos and Kyriakidis (2008) and Kim (2004). Key features of deteriorating machines are the *increasing failure rate* assumption (Dimitrakos and Kyriakidis 2008), and the *decreasing yield* property (Kim 2004). The first feature makes it possible to model the decreasing reliability of the machine over time. In other words, as the machine stays operational longer it is more likely over time that the machine is subject to failure. The second feature models the decreasing fraction of conforming parts the machine produces over time. In other words, as the machine stays operational longer, it is more likely that it produces higher fraction of defective items, due to deterioration. Our method is able to deal with both aspects. It is worth highlighting that the model we propose in this section considers uncontrolled degrading machines. This means that we do not take into account any preventive maintenance policy or quality control action that may stop the machine before the down state is reached. Such extensions will be subject of future research.

Specifically, the considered machine is subject to n degrading operational states and one down state, $n+1$. While in operational state $i = 1, \ldots, n-1$, machine M_k can enter the next deterioration state $i+1$ with rate $\alpha_i^k p_i^k$ or it can fail with rate $(1-\alpha_i^k)p_i^k$. While in the operational state n the machine fails with rate p_n^k. According to the increasing failure rate assumption:

$$(1-\alpha_i^k)p_i^k < (1-\alpha_{i+1}^k)p_{i+1}^k, \quad \forall i = 1, \ldots, n-1 \tag{39}$$

Table 11 Data for systems with multiple non-identical machines per stage

		M_1	M_2	M_3	M_4
Case 1	$p_{k,i}$	0.01	$[0.01, 0.05]$	0.01	
$m_k = [1, 2, 1]$	$r_{k,i}$	0.1	$[0.1, 0.5]$	0.1	
$N_k = [2, 2]$	$\mu_{k,i}$	1	$[0.5, 0.5]$	1	
	P_k	0.909	0.909	0.909	
Case 2	$p_{k,i}$	0.01	$[0.01, 0.05, 0.001]$	0.01	
$m_k = [1, 3, 1]$	$r_{k,i}$	0.1	$[0.1, 0.5, 0.01]$	0.1	
$N_k = [2, 2]$	$\mu_{k,i}$	1	$[0.35, 0.35, 0.3]$	1	
	P_k	0.909	0.909	0.909	
Case 3	$p_{k,i}$	0.01	$[0.01, 0.02]$	$[0.01, 0.01]$	
$m_k = [1, 2, 2]$	$r_{k,i}$	0.1	$[0.1, 0.12]$	$[0.1, 0.2]$	
$N_k = [12, 6]$	$\mu_{k,i}$	1	$[0.6, 0.5]$	$[0.5, 0.5]$	
	P_k	0.909	0.974	0.93	
Case 4	$p_{k,i}$	$[0.01, 0.05]$	$[0.01, 0.02]$	0.01	
$m_k = [2, 2, 1]$	$r_{k,i}$	$[0.1, 0.2]$	$[0.1, 0.4]$	0.1	
$N_k = [6, 8]$	$\mu_{k,i}$	$[0.6, 0.5]$	$[0.72, 0.35]$	1.05	
	P_k	0.945	0.987	0.954	
Case 5	$p_{k,i}$	$[0.05, 0.05]$	$[0.01, 0.02, 0.003]$	0.02	
$m_k = [2, 3, 1]$	$r_{k,i}$	$[0.3, 0.2]$	$[0.1, 0.1, 0.008]$	0.2	
$N_k = [4, 2]$	$\mu_{k,i}$	$[0.6, 0.5]$	$[0.5, 0.3, 0.3]$	1	
	P_k	0.914	0.922	0.869	
Case 6	$p_{k,i}$	0.001	$[0.03, 0.05]$	$[0.07, 0.01]$	0.004
$m_k = [1, 2, 2, 1]$	$r_{k,i}$	0.05	$[0.1, 0.2]$	$[0.02, 0.4]$	0.12
$N_k = [2, 2, 2]$	$\mu_{k,i}$	0.95	$[0.7, 0.5]$	$[0.8, 0.8]$	0.9
	P_k	0.931	0.938	0.958	0.87
Case 7	$p_{k,i}$	$[0.03, 0.04]$	0.004	$[0.09, 0.006]$	$[0.04, 0.003]$
$m_k = [2, 1, 2, 2]$	$r_{k,i}$	$[0.09, 0.1]$	0.03	$[0.12, 0.55]$	$[0.31, 0.01]$
$N_k = [2, 6, 2]$	$\mu_{k,i}$	$[0.7, 0.6]$	1.05	$[0.6, 0.6]$	$[0.7, 0.45]$
	P_k	0.953	0.926	0.936	0.966
Case 8	$p_{k,i}$	0.04	$[0.04, 0.06, 0.009]$	$[0.01, 0.005, 0.03]$	$[0.04, 0.003]$
$m_k = [1, 3, 3, 1]$	$r_{k,i}$	0.29	$[0.12, 0.5, 0.06]$	$[0.1, 0.05, 0.4]$	0.05
$N_k = [4, 2, 2]$	$\mu_{k,i}$	1.1	$[0.7, 0.3, 0.4]$	$[0.5, 0.5, 0.3]$	0.44
	P_k	0.966	1.14	1.188	1.15

When failed, the machine is repaired with rate r_k. Processing rates are equal to μ_k for every degradation state $i = 1, \dots, n$. For every degradation state, the yield Y_i^k is assigned, i.e. the fraction of non-conforming parts the machine produces on average in the specific degradation state i. The state transition diagram representing the machine's behavior is represented in Fig. 9. The mean up time, mt_k^u, and the mean down time, mt_k^d, of the machine

Table 12 Results for test systems with multiple non-identical machines per stage

Case	Method	P	$\overline{n}_1$	$\overline{n}_2$	$\overline{n}_3$
1	Sim	0.832	1.51	0.467	
	95% C.I	±0.018	±0.041	±0.02	
	CG	0.837	1.483	0516	
	$\epsilon(CG), \%$	0.661	−1.32	2.47	
2	Sim	0.835	1.534	0.53	
	95% C.I	±0.008	±0.073	±0.06	
	CG	0.838	1.458	0.541	
	$\epsilon(CG), \%$	0.371	−3.77	0.57	
3	Sim	0.87	6.59	3.21	
	95% C.I	±0.012	±0.21	±0.14	
	CG	0.861	6.449	3.299	
	$\epsilon(CG), \%$	−0.988	−1.169	1.483	
4	Sim	0.881	2.81	2.73	
	95% C.I	±0.007	±0.06	±0.08	
	CG	0.874	2.772	2.685	
	$\epsilon(CG), \%$	−0.703	−0.623	−0.551	
5	Sim	0.768	2.42	0.993	
	95% C.I	±0.021	±0.16	±0.077	
	CG	0.774	2.557	1.011	
	$\epsilon(CG), \%$	0.846	3.425	0.915	
6	Sim	0.76	1.61	1.33	0.619
	95% C.I	±0.003	±0.093	±0.06	±0.038
	CG	0.752	1.573	1.301	0.65
	$\epsilon(CG), \%$	−0.947	−1.84	−1.445	1.59
7	Sim	0.728	1.28	2.98	0.554
	95% C.I	±0.009	±0.092	±0.19	±0.067
	CG	0.737	1.337	3.168	0.62
	$\epsilon(CG), \%$	1.236	2.89	3.148	3.315
8	Sim	0.905	1.78	0.82	0.942
	95% C.I	±0.005	±0.063	±0.04	±0.012
	CG	0.894	1.63	0.799	0.942
	$\epsilon(CG), \%$	−1.138	−3.727	−1.03	0.22

with degrading states are:

$$mt_k^u = \sum_{i=1}^{n} \frac{\prod_{q=1}^{i-1} \alpha_q^k}{p_i^k}, \quad mt_k^d = 1/r_k \tag{40}$$

In addition to the performance measures described in Sect. 3, since our method provides estimates of the probability of machine M_k being in any degradation state i, $\pi^k(i)$, it makes

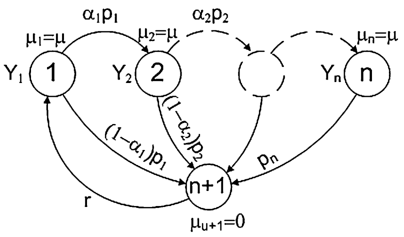


Fig. 9 State transition diagram for a degrading machine with n degradation states

it possible to evaluate the machine yield, the system yield, and the effective production rate of the system (Colledani 2008; Kim 2004). Specifically, when the algorithm converges and all the steady-state probabilities of the pseudo-machine states are estimated, $\pi^k(i)$ can be calculated by using:

$$\begin{aligned}\pi^k(i) &= \sum_{(j,i)_u \in S'_u(k)} \pi(j,i)_u(k) + \pi(i)_u(k)\\ &= \sum_{(i,j')_d \in S'_d(k-1)} \pi(i,j')_d(k-1) + \pi(i)_d(k-1)\end{aligned} \tag{41}$$

Then, the machine yield Y_k and the system yield Y^{Sys} can be calculated by using:

$$Y_k = \frac{\sum_{i=1}^n \pi^k(i) Y_i^k}{\sum_{i=1}^n \pi^k(i)}, \qquad Y^{Sys} = \prod_{k=1}^K Y_k \tag{42}$$

The effective production rate of the system, i.e., the rate at which the system delivers conforming products, is:

$$P^{Eff} = P Y^{Sys} \tag{43}$$

The accuracy of the method is tested over twelve test cases. The systems have four machines, each one subject to four deterioration states. The machines' parameters are identical, except for the processing rates, μ_k, that are equal to $\mu_k = [1, 1.02, 0.99, 1.03]$. Specifically, we consider for all machines one combination of $p_i = [0.01, 0.02, 0.03, 0.04]$, two possible combinations of α, $\alpha_i = [0.5, 0.6, 0.7]$ and $\alpha_i = [0.9, 0.9, 0.9]$, and three possible values of r, $r = 0.005$, $r = 0.05$ and $r = 0.1$. This settings generate six system configurations. It can be shown that condition (39) is satisfied by all combinations of parameters. The yield in the four degradation states is, respectively, $Y_i = [1, 0.95, 0.9, 0.8]$. Finally, two different combinations of buffer capacities generate the twelve test cases. In detail, $\overline{N1} = [2, 2, 2]$, $\overline{N2} = [8, 8, 8]$. The errors in the estimation of the effective production rates for these twelve systems are reported in Table 13. It can be noticed that our method is very accurate in the analysis of this class of systems.

In Table 13, numerical values of the estimated total production rate and system yield are also given. It is interesting to notice that, in the case of uncontrolled degrading machines considered here, the system yield is not dependent on the buffer capacity and the machine's repair rate. A possible explanation is suggested in the following. Since transitions between states are operation dependent, a change in the buffer size causes a variation of the total operational time of the machine but does not cause a variation of the dynamics of the machine up-cycle in visiting the different up states of the progressive degrading process, since no transition is possible when the machine is blocked/starved. In other words, the conditional

Table 13 Results for twelve test systems with four degrading machines

Case	$\overline{N_k}$	r	α_i	P_{CG}	Y_{CG}^{Sys}	P_{CG}^{Eff}	P_{Sim}^{Eff}	$\epsilon(CG), \%$
1	[2, 2, 2]	0.005	[0.5, 0.6, 0.7]	0.153	0.909	0.139	0.137	1.732
2	[8, 8, 8]	0.005	[0.5, 0.6, 0.7]	0.162	0.909	0.147	0.146	0.754
3	[2, 2, 2]	0.005	[0.9, 0.9, 0.9]	0.196	0.831	0.163	0.166	−1.613
4	[8, 8, 8]	0.005	[0.9, 0.9, 0.9]	0.206	0.831	0.171	0.172	−0.455
5	[2, 2, 2]	0.05	[0.5, 0.6, 0.7]	0.655	0.909	0.595	0.591	0.729
6	[8, 8, 8]	0.05	[0.5, 0.6, 0.7]	0.697	0.909	0.633	0.642	−1.281
7	[2, 2, 2]	0.05	[0.9, 0.9, 0.9]	0.719	0.831	0.598	0.596	0.337
8	[8, 8, 8]	0.05	[0.9, 0.9, 0.9]	0.759	0.831	0.631	0.632	−0.156
9	[2, 2, 2]	0.1	[0.5, 0.6, 0.7]	0.799	0.909	0.726	0.723	0.472
10	[8, 8, 8]	0.1	[0.5, 0.6, 0.7]	0.843	0.909	0.766	0.765	0.185
11	[2, 2, 2]	0.1	[0.9, 0.9, 0.9]	0.842	0.831	0.7	0.708	−1.06
12	[8, 8, 8]	0.1	[0.9, 0.9, 0.9]	0.88	0.831	0.732	0.733	−0.125

probability of being in the operational state $i = 1, \dots, n$ given that the machine is operational, independently on the state, does not change. Equivalently, the portion of operational time in which the machine is in the operational state i of the degrading process, $i = 1, \dots, n$, is not impacted by the buffer size variation. As a consequence, referring to the yield equation (42), the buffer size change proportionally impacts both the numerator and the denominator of the equation, thus the system yield remains unvaried. Similar arguments motivate the independency of the system yield upon variation of the machine's repair rate.

On the contrary, the system yield is dependent on the probabilities controlling the degradation process, i.e. α_i. Indeed, as α_i increases, the machine spends more time in the high degradation states, thus the yield is reduced. However, as α_i increases, the machine fails less frequently, thus the total production rate increases. Therefore, due to the combination of these two effects, faster degradation entailing smaller failure rates may increase the effective production rate, as in cases 1 and 3, or decrease the effective production rate, as in cases 9 and 11.

It is worth pointing out that, as already observed in Colledani (2008), Gershwin and Kim (2008), and Kim (2004) for machines featuring two degrading states, in case of controlled degrading machines with remote part inspection and monitoring the system yield is dependent on the buffer size. Indeed, in this case, a delay in the quality information feedback that depends on the buffer size is generated. This delay is due to the time processed parts spend in the portion of system included between the degrading machine and the inspection station that feeds the monitoring tool. Although this case can be handled with the proposed method, this analysis requires more elements and additional notation. Therefore, it will be subject of future research.

7.3 Decomposition algorithm speed

In this paragraph, we discuss some computation issues about the algorithm proposed in this paper. First, we compare the performance of our algorithm, in terms of number of two-machine-line evaluations, with existing algorithms developed to study the performance of systems with single up-single down state machines. Specifically, the results reported in Glassey and Hong (1993) concerning their decomposition method (GH method) and, later,

Table 14 Accuracy and number of two-machine-line evaluations ($2M$)—comparison between the GH, ADDX and CG methods

Case	K	*GH*	*ADDX*	*CG*	*SIM*	*GH*	*ADDX*	*CG*
		P	P	P	P	$\#2M$	$\#2M$	$\#2M$
1	3	0.8356	0.8341	0.8337	0.8312	28	9	6
2	3	0.8588	0.8567	0.8567	0.8613	46	7	4
3	3	0.728	0.7278	0.7251	0.7231	46	9	4
4	3	0.817	0.817	0.8167	0.816	50	7	6
5	3	0.8754	0.8748	0.8716	0.873	28	19	14
6	4	0.8352	0.8257	0.8244	0.8304	402	26	15
7	4	0.8056	0.8	0.7996	0.7962	495	18	11
8	4	0.7476	0.7473	0.7459	0.7426	102	26	19
9	5	0.8256	0.8321	0.8304	0.8206	116	45	28
10	5	0.8323	0.8326	0.8322	0.8259	112	57	34

adopted as a benchmark in Burman (1995) (ADDX method) are considered. In Table 14 the results over ten cases referred to flow lines with inhomogeneous machines are reported. Column 2 reports the number of stations in the test systems. Columns 3 through 6 report the production rate estimated by three decomposition methods and by simulation. Columns 7 through 10 report the number of two-machine-line evaluations required. The amount of time the algorithm takes to converge is indeed roughly related to this measure. In all cases, our method estimates the system performance with the lowest number of two-machine-line evaluations between the considered methods. Also, in 7 of the 10 cases our method provides the best performance estimate, even if the difference is negligible in most of the cases. It should be remarked that the two-machine line model used in our approach is more complex then the two-machine line models used by the GH and ADDX methods. Indeed, our method considers in detail all the possible states of each pseudo-machine, thanks to the flexibility of the general Markovian modeling framework.

For testing the computation speed of our algorithm in more complex cases, we consider the test systems with Cox-2 distributed up and down times analyzed in the previous section. Figure 10 reports the number of two-machine line evaluations and the time the algorithm requires to converge, when running on a 2.2 GHz Intel Core2 Duo CPU, with a convergence tolerance ϵ on the throughput of 10^{-3}. The results show that the number of two-machine-line evaluations increases almost linearly with the number of machines in the system, for all four buffer capacity distributions. In terms of complete algorithm iterations, over the analyzed twenty cases the algorithm always requires less then 4 complete iterations before convergence. Concerning the computation time, the algorithm requires less then 3 seconds for lines with 5 machines or less, while it takes about 7 minutes as a maximum for a seven-machine line with large buffers. The computation time increases almost exponentially with the number of machines. However, it should be pointed out that for the seven-machine line of our test cases, the number of states of the pseudo machine $M_d(1)$ amounts to 24, since every state of the machines M_3 through M_7 can cause partial or complete blocking of machine M_2. This suggest that possible improvements in the algorithm speed could be achieved by designing a suitable state aggregation procedure making it possible to reduce the number of pseudo-machine states while keeping the advantage of our approach in terms of flexibility and accuracy. This issue will be subject of future research.

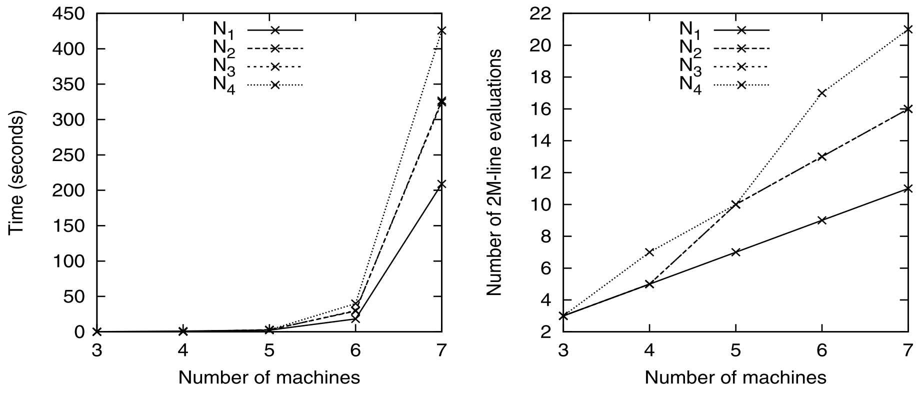


Fig. 10 Algorithm convergence speed in Cox-2 up and down machine lines

8 Complete and partial starvation and blocking probabilities

Our method assigns to pseudo-machines of every building block one remote state for each cause of partial and complete blocking and starvation. Therefore, once the algorithm converges, in addition to the common system performance measures previously discussed, our method provides for the first time the probabilities of finding each stage in every possible state, including all the existing causes for partial and complete blocking and starvation. For example, consider a serial line with 5 machines, each one having a single up state and a single down state. Failure rates are equal to 0.01 for each machine in the line and processing rates are increasing and respectively equal to $[1.13, 1.14, 1.15, 1.2, 1.25]$. Buffer sizes are all equal to 4. With two set of repair rates we obtain two different line configurations, namely line 1 and line 2. In line 1, repair rates are decreasing and respectively equal to $[0.5, 0.4, 0.3, 0.2, 0.1]$; in line 2 repair rates are reversed with respect to line 1 and equal to $[0.1, 0.2, 0.3, 0.4, 0.5]$.

We apply the proposed decomposition method for both line configurations. For line 1, the estimated production rate is 1.014 and the average buffer levels are respectively $[2.8688, 2.4573, 1.3352, 0.7765]$; for line 2 the estimated production rate is 0.985 and the average buffer levels are respectively $[1.8937, 1.0998, 0.4183, 0.2051]$. Moreover, we focus on the last stage of the line, M_5. The probability of finding machine M_5 in every state, including all possible causes of partial and complete starvation, is reported in Table 15 and represented in the histograms of Fig. 11, for both line configurations.

As it can be noticed, while for line 1 both the partial and the complete starvation of M_5 is mainly due to the closest machine M_4, in line 2 stage M_3 has the most relevant impact on M_5 in terms of partial starvation while M_1 is the principal cause for complete starvation, due to its large repair time. This information may be useful for bottleneck identification and root causes analysis, for assigning priorities to alternative improvement actions according to their impact on the overall line performance and for supporting sensitivity analysis. Further investigation is needed to gather insights from this additional information.

9 Conclusions

This paper presents a decomposition method for the approximate analysis of multi-stage continuous flow lines with general Markovian machines. This model and this method make

Table 15 Probability of finding machine M_5 in every state, including all possible causes of partial and complete starvation, for lines 1 and 2

State ID	State type	Root cause	Processing rate	Probability line 1	Probability line 2
1	Partially starved	M_1	1.13	0.02001	0.14596
2	Completely starved	M_1	0	0.00192	0.06296
3	Partially starved	M_2	1.14	0.07289	0.18409
4	Completely starved	M_2	0	0.00724	0.03272
5	Partially starved	M_3	1.15	0.25008	0.24621
6	Completely starved	M_3	0	0.02018	0.02490
7	Partially starved	M_4	1.2	0.28996	0.16227
8	Completely starved	M_4	0	0.03838	0.01941
9	Up at maximum speed	M_5	1.25	0.21824	0.10573
10	Down	M_5	0	0.08112	0.01576

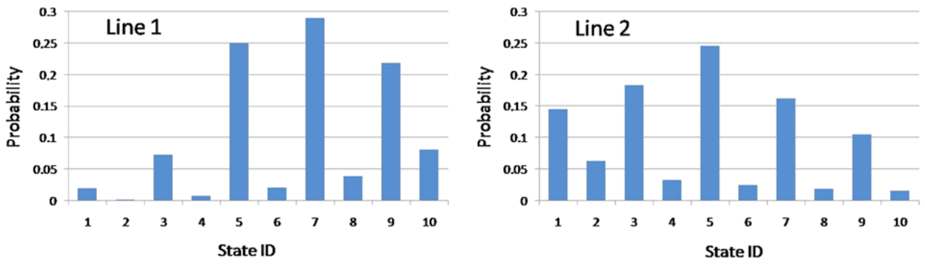


Fig. 11 Histograms of the impact of partial and complete starvation causes on M_5 for lines 1 and 2

it possible to analyze a wide set of different system architectures and machine models within a unique framework. Already analyzed systems, such as unreliable flow lines with single and multiple failure mode machines, lines with identical parallel machines per stage and integrated quality/quantity machine models can be analyzed. Moreover, the method enables the analysis of previously uninvestigated classes of manufacturing systems such as systems with machines subject to aging and to generally distributed up and down times, and systems with multiple non-identical machines per stage. Numerical results show good accuracy of our method and highlight the possibility of managing additional information about the system behavior, having an overall view on the propagation of complete and partial starvation and blocking throughout the line. Future research will be devoted to the application of the proposed method to modeling and studying the properties of these and other previously uninvestigated classes of systems. Priority will be given to the integrated quality/quantity analysis of systems with sampling inspection stations and on-line rework operations, systems with limited repair and workforce capacity, systems where machines have multiple speed levels, systems with machines producing in batches and systems where machines are characterized by degraded yield states.

Acknowledgements The research is partially funded by the Roberto Rocca Project "Interactions Among Quality and Productivity Performance Measures in Production Systems". It is also partly funded by the Singapore-MIT Alliance (SMA) and the MIT-Portugal Program (MPP). The authors would like to thank Prof. Baris Tan for his comments and suggestions on this topic.

Appendix: Decomposition equations for calculating $\lambda_d(k-1)$

Consider the transition rate matrix $\lambda_d(k-1)$. This matrix is a square matrix with size $(I_d(k-1)+I'_d(k-1)) \times (I_d(k-1)+I'_d(k-1))$ which contains the transition rates governing the pseudo-machine $M_d(k-1)$ in its transitions among the states in the state space. The matrix $\lambda_d(k-1)$ is partitioned into four matrices and assumes the form:

$$\lambda_d(k-1) = \begin{vmatrix} \lambda^d_{S,S}(k-1) & \lambda^d_{S,S'}(k-1) \\ \lambda^d_{S',S}(k-1) & \lambda^d_{S',S'}(k-1) \end{vmatrix} \tag{44}$$

In the following the equations for calculating the elements of these block matrices are reported. The rationale behind the derivation of these equations is explained in detail in Sect. 6 for the pseudo-machine $\lambda_u(k)$.

A.1 Equations for determining the elements of $\lambda^d_{S,S}(k-1)$

$$\lambda^d_{i,i'}(k-1) = \lambda^k_{i,i'} \quad i \in S_d(k-1),\ i' \in S_d(k-1) \tag{45}$$

A.2 Equations for determining the elements of $\lambda^d_{S',S'}(k-1)$

$$\lambda^{d,1}_{(i,j),(i,j')}(k-1) = \lambda^d_{j,j'}(k), \quad (i,j) \in S'_d(k-1), (i,j') \in S'_d(k-1) \tag{46}$$

$$\lambda^{d,2}_{(i,j),(i',j)}(k-1) = \psi^k_{i,i'}, \quad (i,j) \in S'_d(k-1), (i',j) \in S'_d(k-1) \tag{47}$$

$$\lambda^d_{S',S'}(k-1) = \lambda^{d,1}_{S',S'}(k-1) + \lambda^{d,2}_{S',S'}(k-1) \tag{48}$$

A.3 Equations for determining the elements of $\lambda^d_{S',S}(k-1)$

$$\lambda^{d,1}_{(i,j),i}(k-1) = \sum_{(i,j') not \in S'_d(k-1)} \lambda^d_{j,j'}(k), \quad (i,j) \in S'_d(k-1), i \in S_d(k-1) \tag{49}$$

$$\lambda^{d,2}_{(i,j),i'}(k-1) = \psi^k_{i,i'}, \quad (i,j) \in S'_d(k-1),\ i' \in S_d(k-1) \tag{50}$$

$$\lambda^d_{S',S}(k-1) = \lambda^{d,1}_{S',S}(k-1) + \lambda^{d,2}_{S',S}(k-1) \tag{51}$$

A.4 Equations for determining the elements of $\lambda^d_{S,S'}(k-1)$

$$\forall (i,j) \in S'_d(k-1):$$

$$\lambda^d_{i,(i,j)}(k-1) = \frac{\pi(i,j)_d(k-1)(\sum_{q \in (S_d(k-1) \cup S'_d(k-1))} \lambda^d_{(i,j),q}(k-1))}{\pi(i)_d(k-1)} - \frac{\sum_{q \in S'_d(k-1)} \lambda^d_{q,(i,j)}(k-1)\pi(q)_d(k-1)}{\pi(i)_d(k-1)} \tag{52}$$

References

Alden, J. M., Burns, L. D., Costy, T., Hutton, R. D., Jackson, C. A., Kim, D. S., Kohls, K. A., Owen, J. H., Turnquist, M. A., & Vander Veen, D. J. (2006). General Motors increases its production throughput. *Interfaces*, *36*(1), 6–25.

Aldous, D., & Shepp, L. (1987). The least variable phase-type distribution is Erlang. *Stochastic Models*, *3*, 467–473.

Altiok, T. (1985). On the phase-type approximations of general distributions. *IIE Transactions*, *17*(2), 110–116.

Altiok, T., & Ozdogru, U. (2003). *Analysis of two-valve fluid flow systems with general repair times*. *Analysis and modeling of manufacturing systems* (pp. 255–288). Dordrecht: Kluwer Academic.

Alvarez-Vargas, R., Dallery, Y., & David, R. (1994). A study of the continuous flow model of production lines with unreliable machines and finite buffers. *Journal of Manufacturing Systems*, *13*(3), 221–234.

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

Burman, M. H., Gershwin, S. B., & Suyematsu, C. (1998). Hewlett-Packard uses operations research to improve the design of a printer production line. *Interfaces*, *28*(1), 24–36.

Buzacott, J. A., & Hanifin, L. E. (1978). Models of automatic transfer lines with inventory banks: a review and comparison. *AHE Transactions*, *10*, 197–207.

Buzacott, J. A., & Shantikumar, J. G. (1993). *Stochastic models of manufacturing systems*. Englewood Cliffs: Prentice Hall.

Colledani, M., & Tolio, T. (2004). Performance evaluation of continuous production lines with deterministic processing times, multiple failure modes and multiple part types. In *Proceedings of 4th CIRP conference on intelligent computation in manufacturing engineering, ICME04*, June 30th–July 2nd 2004, Sorrento, Italy (pp. 29–34).

Colledani, M. (2008). Integrated analysis of quality and production logistics in asynchronous manufacturing lines. In *17th IFAC world congress*, July 6–11, Seoul, Korea.

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

Dallery, Y., & Le Bihan, H. (2000). A robust decomposition method for the analysis of production lines with unreliable machines and finite buffers. *Annals of Operations Research*, *93*, 265–297.

Dimitrakos, T. D., & Kyriakidis, E. G. (2008). A Semi-Markov decision algorithm for the maintenance of a production system with buffer capacity and continuous repair times. *International Journal of Production Economics*, *111*, 752–762.

Gershwin, S. B., & Schick, I. C. (1980). Continuous model of an unreliable two-stage material flow system with a finite interstage buffer. M.I.T. Laboratory for Information and Decision Systems Report LIDS-R-1039.

Gershwin, S. B., & Dallery, Y. (1992). Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems Theory and Applications* (Vol. 12, pp. 3–94). Special Issue on Queueing Models of Manufacturing Systems

Gershwin, S. B. (1994). *Manufacturing systems engineering*. Englewood Cliffs: Prentice-Hall.

Gershwin, S. B., & Kim, J. (2008). Analysis of long flow lines with quality and operational failures. *IIE Transactions*, *40*, 284–296.

Gershwin, S. B., & Tan, B. (2009). Analysis of a general Markovian two-stage continuous-flow production system with a finite buffer. *International Journal of Production Economics*, *120*(2), 327–339.

Gershwin, S. B., & Tan, B. (2010). Modelling and analysis of Markovian continuous flow systems with a finite buffer. *Annals of Operations Research*, *182*(1), 5–30.

Gershwin, S. B. (2011). Case generation for algorithm testing. In *Proceedings of the SMMSO 2011, 8th conference on stochastic models of manufacturing and service operations*, May 28–June 2 2011, Turkey (pp. 193–200).

Glassey, C. R., & Hong, Y. (1993). Analysis of behaviour of an unreliable n-stage transfer line with $(n-1)$ inter-stage storage buffers. *International Journal of Production Research*, *31*(3), 519–530.

Helber, S., & Jusic, H. (2004). A new decomposition approach for non-cyclic continuous material flow lines with a merging flow of material. *Annals of Operations Research*, *125*, 117–139.

Horvath, A. (2003). *Approximating non-Markovian behaviour by Markovian models*. Ph.D. thesis, Budapest University of Technology and Economics.

Horvath, A., & Telek, M. (2002). Phfit: a general phase-type fitting tool. In *Proceedings of the TOOLS 2002 conference* (pp. 82–91).

Inman, R. R. (1999). Empirical evaluation of exponential and independence assumptions in queueing models of manufacturing systems. *Production Operations Management*, *8*, 409–432.

Kim, J. (2004). *Designing production systems for quality and quantity*. Ph.D. thesis, Massachusetts Institute of Technology. Available at http://cell1.mit.edu/theses/kim-jongyoon-phd.pdf.

Levantesi, R., Matta, A., & Tolio, T. (1999). Continuous two-machine line with multiple failure modes and finite buffer capacity. In *Proceedings of IV AITEM conference*, Brescia, Italy, September 13–15.

Levantesi, R., Matta, A., & Tolio, T. (2003). Performance evaluation of continuous production lines with machines having different processing rates and multiple failure modes. *Performance Evaluation*, *51*, 247–268.

Patchong, A., & Willaeys, D. (2001). Modeling and analysis of an unreliable flow line composed of parallel-machine stages. *IIE Transactions*, *33*, 559–568.

Patchong, A., Lemoine, T., & Kern, G. (2003). Improving car body production at PSA Peugeot Citroën. *Interfaces*, *33*(1), 36–49.

Tan, B., & Yeralan, S. (1997). A decomposition model for continuous materials flow production systems. *International Journal of Production Research*, *35*(10), 2759–2772.

Osogami, T., & Harchol-Balter, M. (2006). Closed form solutions for mapping general distributions to quasi-minimal PH distributions. *Performance Evaluation*, *62*, 524–552.

Tan, B. (2001). A three station continuous materials flow merge system with unreliable stations and a shared buffer. *Mathematical and Computer Modelling*, *33*(8–9), 1011–1026.