Approximate Analysis of Unreliable Transfer Lines with Rework or Scrapping of Parts

by

S. Helber

OR 320-97 June 1997

Approximate Analysis of Unreliable Transfer Lines with Rework or Scrapping of Parts

Stefan Helber

Institut für Produktionswirtschaft und Controlling
Fakultät für Betriebswirtschaftslehre
Ludwig-Maximilians-Universität München
Ludwigstr. 28 RG
D-80539 München

Helber@Bwl.Uni-Muenchen.de

June 1997

Abstract

This paper presents a Markov process model and an approximate decomposition technique for a discrete material transfer line with limited buffer capacity. A fraction of the parts processed at some station in the line may be scrapped or reworked to meet product quality requirements. Feeding back the reworked parts leads to cycles in the flow of material. Processing times are deterministic and identical for all machines and are taken as the time unit. Machine specific times to failure and to repair are geometrically distributed. The model is analyzed through a decomposition into two-machine systems. We develop new decomposition equations for machines performing split and merge operations. Production rates and inventory levels are computed and compared to simulation results. The results indicate that the method produces useful results for a variety of systems.

Contents

List of Figures

List of Tables

1 Introduction

1.1 The Problem

Many stochastic models of production lines assume a purely linear flow of material through a serial arrangement of machines interconnected by buffers of limited capacity. The analysis of these models shows the impact of random processing times and/or machine failures on system performance measures such as production rates and inventory levels. It is often assumed that parts processed at a machine are all perfect and therefore the throughput of all machines in this linear arrangement is the same. However, in reality parts may not always be of perfect quality. Defective parts may have to be reworked and sent back into the main line or they may have to be scrapped. In both cases, the flow of material is no longer purely linear, the throughput of the machines is no longer identical, and the rework and scrapping processes may have a major impact on the total system performance. We develop a simple model of a transfer line extended by rework and scrapping machines to study the impact of both product quality and the processing of defective parts on the system behavior. This model allows for loops in the flow of material as depicted in Figure 1.

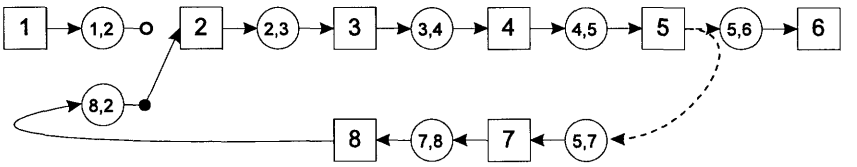

Figure 1: System with a Loop in the Flow of Material

In Figure 1, the squares indicate unreliable machines M_i and the circles represent buffers $B_{i,q}$ of limited capacity $C_{i,q}$ between adjacent machines M_i and M_q. In this system, bad parts may be detected at Machine M_5. At Machines M_7 and M_8, all bad parts receive some treatment such that previous processing steps can be repeated. These parts are fed back into the line at Machine M_2. The approximation technique developed in this paper makes it possible to determine production rate and inventory level estimates for systems like the one in Figure 1. It is fast and reliable and can be used to quickly narrow down the range of possible solutions when designing a transfer line. Slower but more detailed simulation models can then be used for the fine-tuning of the remaining candidate solutions to the design problem.

1.2 The Model

To analyze production lines with scrapping and rework, we extend an existing model of a transfer line [Gershwin, 1987, Gershwin, 1994] by allowing for two additional phenomena concerning the flow of material. These two phenomena are split and merge operations. The previously existing model as well as the numerical technique to determine performance measures assumed a purely linear flow of material. This situation is depicted in Figure 2.

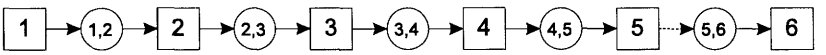

Figure 2: Production Line with Linear Flow of Material

The system produces discrete parts. Processing times are assumed to be deterministic and identical for all machines and are taken as the time unit. A machine processes a part during a time unit if it is not starved (at least one input buffer is non-empty), not blocked (none of its output buffers is full), and it does not fail. We assume geometrically distributed operation dependent failures (ODFs) at the machines, i.e. a machine M_i which is is neither starved nor blocked and could thus process a part fails at the beginning of a period with probability p_i. If it is either starved or blocked, it cannot fail. Machine M_i is repaired at the beginning of a period with probability r_i if it was down during the previous period, i.e. times to repair are also geometrically distributed.

We also assume blocking after service (BAS): Machine M_i is blocked if it has processed a part and finds its output buffer full. In this case, the processed part remains at the workspace of Machine M_i until a space in the downstream buffer becomes available. We further assume that machine states change (due to failures and repairs) at the beginning of periods whereas buffer levels change (due to completion of processing) at the end of periods. Travel times within buffers are zero. A more detailed and formal description of the model is given in [Gershwin, 1994, pages 71–74].

An example of the first new phenomenon which we model is the *split operation* depicted in Figure 3. In this system, there is one machine, M_2, which has *multiple alternative* immediate successors. After processing a part, Machine M_2 sends this part to *one* of its immediate successors. Thus, if at time t a part is sent from Machine M_2 to M_3, then nothing is sent to M_4 and M_5. We use this concept of split operations to model a probabilistic routing of parts due to random quality properties such as being "good" or "bad".

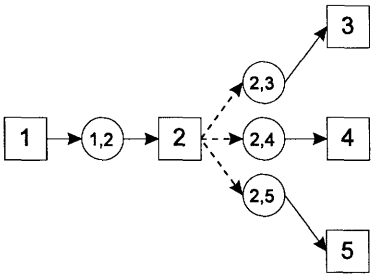

Figure 3: System with a Split Operation

We assume that the routing decision is made *after* the part has been processed. The processing step at the split machine may therefore represent a quality inspection. Thus, if Machine M_2 does not process a part because it is down, starved, or blocked, no routing decision is made. We model the routing decision as the flip of a multi-sided coin. That is, the choice of the succeeding machine for any part is random and independent of the state of the surrounding machines and buffers and the choices for previous parts. (This may be a simplification since in reality, machine failures and the production of bad parts may be correlated.) Formally, if $D(i)$ is the set of buffers immediately downstream of Machine M_i, a part processed by Machine M_i is sent to M_q with probability $d_{i,q}$ where $\sum_{(i,i)\in D(i)} d_{i,q} = 1$. The directed arcs in Figure 3 between Machine M_2 and its immediately succeeding buffers are broken to indicate that they represent *alternative* routings. We will use these alternative routings to model phenomena such as scrapping or rework of bad parts.

The split in the flow of material after Machine M_2 in Figure 3 should not be confused with a disassembly operation. In a disassembly operation, there is no choice between different downstream buffers as *each* downstream buffer receives one part whenever an operation has been performed by the disassembly machine. The split operation in this paper, however, is due to a *random choice* between alternative routings and only one of the alternative downstream buffers receives the part. Note that the routing depends solely on the random quality properties of the respective part: If we have produced a good part and the downstream buffer holding good parts is full while the buffer holding bad parts is not full, the good part remains at the workspace of its current machine. This machine is now blocked until a buffer space for good parts becomes available.

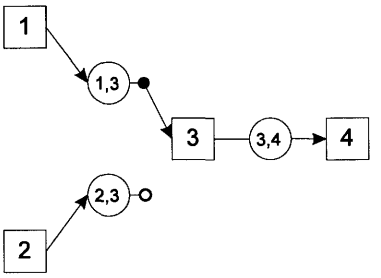

Figure 4: System with a Merge Operation

The second phenomena which we introduce into the analysis of production lines is the *merge operation* shown in Figure 4. Here, Machine M_3 has two immediate predecessors, M_1 and M_2. Whenever Machine M_3 performs an operation, it takes a part out of either Buffer $B_{1,3}$ or $B_{2,3}$. Machine M_3 is starved if both buffers are empty.

We assume that from the point of view of Machine M_3, parts coming from M_1 and M_2 are identical. This allows us to model the re-entrance of reworked parts as in Figure 1. We have to specify how a merging machine selects between its two input buffers. For the sake of simplicity, we assume that a merging machine uses a *priority ranking*. In Figure 4, the priority one buffer $B_{1,3}$ will always be chosen unless it is empty. It is only in this case that a part is taken from the priority two buffer $B_{2,3}$. In the graphical representation in Figure 4, the priority one buffer $B_{1,3}$ is depicted with a closed connection to its downstream machine whereas the connection between the priority two buffer $B_{2,3}$ and the downstream machine is open.

Note that Machine M_2 does not perform an assembly operation as there is a *choice* between two input buffers which hold the same part type (from the perspective of Machine M_2). In an assembly operation, a part is taken from each input buffer as different part types are matched in the assembly process.

If we allow for machines where the flow of material splits or merges, we can model systems like the one shown in Figure 1. In Figure 1, Buffer $B_{8,2}$ containing the reworked parts has priority over Buffer $B_{1,2}$. Giving the higher priority to reworked parts leads to a lower total inventory and decreases the probability of deadlock situations.

We finally state the assumptions that each splitting machine has exactly one immediate predecessor, that each merging machine has exactly one suc-

cessor, that all *input machines* without preceding machines are never starved, that all *output machines* without succeeding machines are never blocked, and that there is at least one input and one output machine in the system. If each machine has no more than one predecessor and successor, the transfer line model in [Gershwin, 1987] results.

1.3 Related Research

Several researchers have studied unreliable transfer lines and/or assembly/-disassembly (A/D) systems with limited buffer capacity. A recent and comprehensive survey is given by [Dallery and Gershwin, 1992]. Their review includes the literature on reliable two-machine transfer lines, on transfer lines without buffers as well as longer lines with more than two machines and A/D systems. Earlier reviews are [Koenigsberg, 1959], [Buxey et al., 1973] and [Buzacott and Hanifin, 1978].

Transfer lines and A/D systems are often modeled as Markov processes to allow for an analytic solution or an accurate approximation. Many of these approximations are based on a decomposition of the complete system into a set of single server queues [Hillier and Boling, 1967] or two-machine transfer lines [Gershwin, 1987, Sevast'yanov, 1962, Zimmern, 1956] which can be evaluated analytically. The main advantage of analytical approaches as opposed to simulation is that the analytical techniques are much faster. This is crucial if a larger number of different systems has to be evaluated in order to find a configuration which is optimal with respect to some objective.

When analyzing the related work with respect to two-machine models and decomposition approaches, we can distinguish ([Dallery and Gershwin, 1992])

- Markov processes with discrete state and discrete time,
- Markov processes with discrete state and continuous time, and
- Markov processes with mixed state and continuous time.

In the first two cases, the state is discrete since discrete parts are produced and machines can be either operational (up) or under repair (down). Time is divided into discrete periods in the first case or treated as continuous in the second. The third group of Markov processes assumes that continuous material is produced in continuous time (which leads to a continuous buffer level), but machine states are discrete.

Type of Process	Analysis of Two-Machine Models	Approximate Decomposition Approaches
Discrete State/ Discrete Time	[Artamonov, 1977] [Buzacott, 1967] [Buzacott and Hanifin, 1978] [Gershwin and Schick, 1983] [Helber, 1995] [Okamura and Yamashina, 1977] [Yeralan and Muth, 1987]	[Dallery et al., 1988] [Gershwin, 1987] [Gershwin, 1991] [Helber, 1997]
Discrete State/ Continuous Time	[Buzacott, 1972] [Gershwin and Berman, 1981] [Sastry and Awate, 1988]	[Choong and Gershwin, 1987] [Gershwin, 1989] [Helber, 1997]
Continuous State/ Continuous Time	[Gershwin, 1994] [Gershwin and Schick, 1980] [Sevast'yanov, 1962] [Wijngaard, 1979] [Zimmern, 1956]	[Burman, 1995] [Dallery and Xie, 1989] [Di Mascolo et al., 1991]

Table 1: Two-Machine Models and Approximation Approaches

Table 1 gives an overview of two-machine models and decomposition approaches for the case of unreliable machines and limited buffer capacity. Current textbooks covering these and similar techniques in detail are [Altiok, 1996, Buzacott and Shanthikumar, 1993, Papadopoulus et al., 1993] as well as [Gershwin, 1994] which gives a thorough introduction into how to derive these models. In this paper, a two-machine transfer line decomposition of the discrete state-discrete time type will be developed.

The few papers which address scrapping of bad parts in the context of performance analysis of production lines differ with respect to the assumptions about scrapping of bad parts. [Okamura and Yamashina, 1977] assume in a two-machine model that whenever a stage breaks down, its current part is scrapped. [Shanthikumar and Tien, 1983] develop a two-machine model where parts are scrapped with some probability when their current ma-

chine fails. [Jafari and Shanthikumar, 1987] extend this two-machine model to longer transfer lines and present an approximation technique to determine production rates and buffer levels. The scrapped parts leave the line immediately and can never be reworked. Some papers consider scrapping and/or rework for two- and multistage [Yu and Bricker, 1993] systems with unlimited buffer capacity.

In a postdoctoral thesis, [Schmidbauer, 1995] introduces a decomposition of a transfer line with stochastic demand into a set of two-buffer, one-machine subsystems with a stochastic demand indicator behind the second machine (see also [Schmidbauer and Rösch, 1994]). Introducing this demand indicator allows to determine service levels. He also considers a random routing due to a split operation, but he only models the case of two instead of n output buffers and he does not analyze merge operations.

[Bürger, 1997] develops a decomposition approach for reliable and unreliable linear transfer lines which allow for scrapping of parts at each machine. Scrapped parts leave the system and cannot be reworked.

[Gopalan and Kannan, 1994] present a two-machine zero-buffer model in which bad parts can be reworked or scrapped. Rework takes place at the machine where the bad part is produced and starts immediately.

[Pourbabai, 1990] describes a model with more than two machines and non-zero buffers, but assumes that if a blockage occurs, the blocked workpieces are permanently lost.

We are not aware of papers which explicitly consider a random routing of parts and loops in the flow of material due to rejects, rework, or scrapping of bad parts in the presence of limited buffers and unreliable machines.

The work most closely related to the research reported below is the transfer line model in [Gershwin, 1987] and a Ph.D. thesis on flow line analysis [Burman, 1995]. Burman shows how to reformulate decomposition equations in a way that leads to a dramatically improved convergence behavior.

The remainder of the paper is organized as follows: In Section 2, we derive decomposition equations for machines which perform split and merge operations. Section 3 describes an iterative algorithm based on these equations and on results by [Dallery et al., 1988] and [Burman, 1995]. Production rate and buffer level estimates obtained by this algorithm are compared to results from a simulation model in a large-scale numerical study in Section 4. We show a wide range of systems where the method works well and some few cases where it should not or cannot be used. Section 5 contains concluding remarks and suggestions for future research.

2 Derivation of Decomposition Equations

2.1 Principle of the Decomposition Technique

The state space of the Markov process model developed in Section 1 is so large that we cannot compute its steady-state probabilities. The reason is that there can be an extremely large number of *combinations* of different machine states and buffer levels if there are more than two machines and one buffer.

The state space for a two-machine, one-buffer system with N buffer spaces, however, is relatively small: Each of the two machines in a two-machine system can be either up or down, i.e. there are $2 \cdot 2 = 4$ different machine states. The buffer can be empty or hold up to N parts, that is there are $N + 1$ buffer levels and the total size of the state space is therefore $4(N + 1)$. Some of these states may be transient, i.e. their steady-state probability is zero. An example is the situation that no workpiece is at the second machine or in the buffer and that both machines are down: In the case of operation dependent failures, the second machine can only fail while it is processing a workpiece, and therefore this is a transient state.

The computational effort to solve a two-machine model of the type given above is negligible. Solving the model in the context of performance evaluation means to determine all steady state probabilities and compute performance measures such as production rate and inventory levels.

To determine performance measures for the system modeled in Section 1, we decompose it into a set of two-machine transfer lines of the same type, i.e. with unreliable machines and limited buffer capacity. The reason for this approach is that the virtual two-machine systems arising in the decomposition can be solved easily. The decomposition to be derived below is a generalization of the one given in [Gershwin, 1987].

For each buffer $B_{j,i}$ between two machines M_j and M_i, we introduce a virtual upstream machine $M_u(j,i)$ which represents to "an observer in the buffer of the original line" [Dallery and Gershwin, 1992] the flow of material into this buffer. A virtual downstream machine $M_d(j,i)$ represents the flow out of this buffer in the original line. The split system depicted in the upper part of Figure 5, for example, is decomposed in three two-machine systems corresponding to the buffers in the original system as shown in the lower part of Figure 5.

We have to determine the failure and repair probabilities $p_u(j,i)$, $r_u(j,i)$,

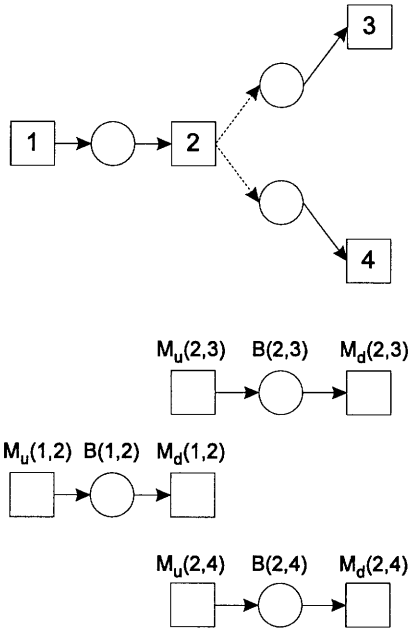

Figure 5: Example of a Decomposition

$p_d(j,i)$, and $r_d(j,i)$ of these virtual machines for each line $L(j,i)$. Given these parameters, we can efficiently compute the steady-state probabilities of all the virtual two-machine lines using the procedure by Gershwin and Schick reported in [Gershwin and Schick, 1980, Gershwin, 1994]. Performance measures of the underlying model can then be derived from the steady-state probabilities of these two-machine models.

The goal of the derivation in the remainder of Section 2 is therefore to determine the failure and repair probabilities for the virtual machines in the decomposition. This leads to a system of equations that has to be solved simultaneously as the perspectives of different observers in different buffers of the original system are interrelated. To solve these equations, a version of the iterative DDX-algorithm ([Dallery et al., 1988]) is used.

Several of the decomposition equations are approximations. For this reason, the performance measures determined by the decomposition are also approximations and their accuracy has to be checked against a simulation model.

2.2 Some Notation

Let $\alpha_i(t)$ denote the state of the real machine M_i where $\{\alpha_i(t) = 1\}$ is the event that Machine M_i is up and $\{\alpha_i(t) = 0\}$ the event that Machine M_i is down in period t. Similarly, $\alpha_u[(i,m),t]$ and $\alpha_d[(i,m),t]$ denote the state of the virtual up- and downstream machines in the two-machine line $L(i,m)$ related to Buffer $B_{i,m}$.

The conditional probability that a part moves from Machine M_i to Buffer $B_{i,m}$ at time t, given that at time t the part is processed by Machine M_i, is $d_{i,m}$. Let prob$[\{\beta_i(t) = m\}]$ denote the unconditional probability that a part is processed by Machine M_i at time t and then it goes to Buffer $B_{i,m}$. Since Machine M_i is occasionally starved, blocked or down, prob$[\{\beta_i(t) = m\}] \neq d_{i,m}$.

The physical buffer between Machines M_i and M_m can hold up to $C_{i,m}$ parts. The extended storage $N(i,m)$ related to Line $L(i,m)$ in the decomposition includes the workspaces at the virtual machines $M_u(i,m)$ and $M_d(i,m)$, i.e. $N(i,m) = C_{i,m} + 2$. A part is assumed to be *stored* in the work area of an upstream machine if it has been processed and cannot be removed due to a full buffer immediately downstream, that is, when the upstream machine is blocked.

Let $\mathbf{p}[(i,m); n\alpha_u\alpha_d]$ denote the steady-state probability of finding the virtual upstream machine $M_u(i,m)$ in Line $L(i,m)$ in state α_u, the downstream machine $M_d(i,m)$ in state α_d and the buffer at level n.

2.3 Conservation of Flow

The conservation of flow (COF) property for systems with split and merge operations differs from those for transfer lines and A/D networks. In A/D systems, the rate of flow of material through each buffer is the same. In a split or merge system, however, the flow to each machine M_i over all immediate upstream buffers $U(i)$ equals the flow from each machine over all immediate downstream buffers $D(i)$. Define E_i as the production rate of Machine M_i in the real system and $E_{j,i}$ as the production rate through Buffer $B_{j,i}$.

The conservation of flow equation for the real system

$$E_i = \sum_{(j,i)\in U(i)} E_{j,i} = \sum_{(i,q)\in D(i)} E_{i,q}, \ \forall i \tag{1}$$

states that the flow into a machine equals the flow out of the machine. The

decomposition must be performed in a way that a similar condition is met by the production rates in all the virtual two-machine lines.

Define $E(j,i)$ and $E(i,q)$ as the production rate in the decomposed two-machine lines $L(j,i)$ and $L(i,q)$, respectively. The decomposition must satisfy the following conservation of flow equation

$$\sum_{(j,i)\in U(i)} E(j,i) = \sum_{(i,q)\in D(i)} E(i,q),\ \forall i \tag{2}$$

which couples the solutions for the different two-machine models.

$E(i)$ is the total production rate related to Machine M_i in the two-machine lines $L(i,q)$ or $L(j,i)$, i.e.

$$E(i) = \sum_{(i,q)\in D(i)} E(i,q) = \sum_{(j,i)\in U(i)} E(j,i),\ \forall i. \tag{3}$$

The following condition must hold *in addition* if we assume that a part which has been processed by Machine M_i is routed to Machine M_m with a routing probability $d_{i,m}$, irrespective of buffer levels or machine states:

$$d_{i,m} = \frac{E(i,m)}{\sum_{(i,q)\in D(i)} E(i,q)},\ (i,m) \in D(i), \forall i \tag{4}$$

It says that the ratio of the flow rates through the different output buffers of Machine M_i is determined by the routing probabilities. This is because the routing decision is made *after* the part has been processed. If the selected buffer, for example for good parts, is full, the part just processed remains at the workspace of the machine which is now blocked until a space in the buffer becomes available. The COF equation for the transfer line model in [Gershwin, 1987] is a special case of (2) where each machine has no more than one input buffer and output buffer, respectively.

The COF equation serves a stopping criterion in the iterative algorithm that is based on the decomposition. The decomposition must lead to a set of failure and repair parameters for the virtual machines such that production rates in the two-machine lines satisfy the conservation of flow equations for all the machines to a prespecified accuracy.

2.4 Flow Rate-Idle Time Relationship

In this section, a set of equations is derived that can be used to determine the failure probabilities $p_u(j,i)$ and $p_d(j,i)$ of the virtual machines $M_u(j,i)$

and $M_d(j,i)$ in each line $L(j,i)$ of the decomposition. (An additional set of equations is needed to compute the repair probabilities.) The equations are needed in a form that depend on parameters of real and/or virtual machines and performance measures of two-machine lines as these are the only available quantities in a two-machine decomposition.

The flow rate through a machine is determined by the failures and repairs of the machine and by the probability that a machine is starved or blocked due to events that happen up- or downstream in the system. The *flow rate-idle time* equation relates the production rate of a machine to these two factors.

The effect of failures of Machine M_i can be analyzed under the assumption that M_i operates in isolation. In isolation, a Machine M_i is always either up and working (and waiting for the next failure), or it is down (and waiting for the next repair).

The mean time to failure $MTTF_i$ of a Machine M_i which is never starved nor blocked is the inverse of the failure probability, i.e.

$$MTTF_i = \frac{1}{p_i}, \tag{5}$$

and a similar equation holds for the mean time to repair $MTTR_i$ with

$$MTTR_i = \frac{1}{r_i}. \tag{6}$$

Define e_i as the isolated production rate of the real machine M_i if it is never starved nor blocked [Gershwin, 1994, p. 75], i.e. the fraction of time Machine M_i is up. Since a machine operating in isolation can only be up or down, the following holds:

$$e_i = \frac{MTTF_i}{MTTF_i + MTTR_i} = \frac{r_i}{r_i + p_i} \tag{7}$$

Machine M_i can only fail if it is neither starved nor blocked. All input buffers must be empty for Machine M_i to be starved whereas one full output buffer is sufficient to block it. The flow rate-idle time (FRIT) relationship is therefore

$$E_i = e_i \text{ prob} \left[\begin{array}{ll} \{n(l,i) > 0, & \text{some } (l,i) \in U(i)\} \text{ and} \\ \{n(i,q) < N(i,q), & \forall (i,q) \in D(i)\} \end{array} \right] \tag{8}$$

where $n(l,i)$ denotes the buffer level in Buffer $B_{l,i}$. It says that the production rate E_i is the probability that Machine M_i is up and neither blocked nor starved.

We assume that the probability of a machine being blocked and starved simultaneously is negligible. This is a common assumption in the analysis of *linear transfer lines* which helps to approximate the probability that M_i is neither starved nor blocked. A similar assumption appears to be reasonable in the analysis of *split and merge systems.*

The reason is that, compared to a purely linear transfer line, having two instead of one input buffer makes starvation of Machine M_i *ceteris paribus* less likely. Similarly, having several output buffers makes blocking less likely. Thus, for a machine with multiple predecessors or successors, the probability of it being starved and blocked simultaneously is smaller than for a machine in a linear transfer line.

If a machine has multiple output buffers due to a split operation, only one of these buffers can be full at any time. However, the two input buffers of a merge machine can be empty simultaneously. The probability of having an empty priority two buffer depends on the level of the corresponding priority one buffer: The priority two buffer is more likely to be empty if the priority one buffer is empty as well. As an approximation, however, we assume that input buffer levels are independent to find

$$E(i) \approx e_i \left[\left(1 - \prod_{(l,i)\in U(i)} \text{prob}[\{n(l,i) = 0\}] \right) \left(1 - \sum_{(i,q)\in D(i)} \text{prob}[\{n(i,q) = N(i,q)\}] \right) \right]. \quad (9)$$

Given that the probability of a machine being blocked and starved simultaneously is negligible, we can further approximate:

$$E(i) \approx e_i \left[1 - \prod_{(l,i)\in U(i)} \text{prob}[\{n(l,i) = 0\}] - \sum_{(i,q)\in D(i)} \text{prob}[\{n(i,q) = N(i,q)\}] \right] \quad (10)$$

Define $e_u(i,q) = r_u(i,q)/(r_u(i,q) + p_u(i,q))$ as the isolated production rate of the virtual upstream machine in the two-machine line $L(i,q)$ and

$e_d(l,i) = r_d(l,i)/(p_d(l,i) + r_d(l,i))$ as the isolated production rate of the virtual downstream machine in Line $L(l,i)$. The two-machine flow rate-idle time equations [Gershwin, 1994, p. 81-82] can be expressed as

$$\text{prob}[\{n(i,q) = N(i,q)\}] = 1 - \frac{E(i,q)}{e_u(i,q)} \tag{11}$$

and

$$\text{prob}[\{n(l,i) = 0\}] = 1 - \frac{E(l,i)}{e_d(l,i)} \tag{12}$$

yielding

$$E(i) = e_i \left[1 - \prod_{(l,i)\in U(i)} \left(1 - \frac{E(l,i)}{e_d(l,i)} \right) - \sum_{(i,q)\in D(i)} \left(1 - \frac{E(i,q)}{e_u(i,q)} \right) \right]. \tag{13}$$

This leads to two sets of equations used to determine the parameters of the two-machine lines:

$$\frac{r_u(i,m) + p_u(i,m)}{r_u(i,m)} = \frac{1}{e_u(i,m)} = K_1 \tag{14}$$

$$\frac{r_d(j,i) + p_d(j,i)}{r_d(j,i)} = \frac{1}{e_d(j,i)} = K_2 \tag{15}$$

where

$$K_1 = \frac{\frac{E(i)}{e_i} + \prod_{(l,i)\in U(i)}(1 - \frac{E(l,i)}{e_d(l,i)}) + \sum_{(i,q)\in D(i), q\neq m}(1 - \frac{E(i,q)}{e_u(i,q)})}{E(i,m)} \tag{16}$$

$$K_2 = [\frac{\frac{E(i)}{e_i} + \sum_{(i,q)\in D(i)}(1 - \frac{E(i,q)}{e_u(i,q)}) - 1}{\prod_{(l,i)\in U(i), l\neq j}(1 - \frac{E(l,i)}{e_d(l,i)})} + 1]\frac{1}{E(j,i)} \tag{17}$$

This is a useful result as it relates, for example for Line $L(i,m)$ in (14), failure and repair probabilities $r_u(i,m)$ and $p_u(i,m)$ to the given isolated efficiency e_i of Machine M_i, to the isolated efficiencies $e_d(l,i)$ and $e_u(i,q)$ of other virtual machines, and to production rates $E(l,i)$ and $E(i,q)$ in other two-machine models of the decomposition. Whenever the parameters $r_u(i,m)$ and

$p_u(i,m)$ for the upstream machine $M_u(i,m)$ of Line $L(i,m)$ are updated, all these other quantities are given or can be easily computed.

Note that the term K_1 in (16) related to Line $L(i,m)$ contains parameters and performance measures of adjacent two-machine systems *other* than $L(i,m)$. Separating these parameters from $p_u(i,m)$ and $r_u(i,m)$ in (14) will later allow us to solve the complete set of decomposition equations simultaneously in a way proposed in [Burman, 1995] which leads to an improved convergence behavior of the algorithm. The same holds for K_2 and the downstream parameters $p_d(j,i)$ and $r_d(j,i)$. The FRIT equation for the transfer line model in [Gershwin, 1987, Gershwin, 1994] is again a special case of (14) and (15).

2.5 Resumption of Flow Equations I: Split Operations

A second set of equations is required to determine the repair probabilities $r_u(i,m)$ and $r_d(i,m)$ for each line $L(i,m)$ in the decomposition. Since the flow resumes after each of these repairs, the equations are frequently called *resumption of flow* equations [Gershwin, 1994]. They are required in a form similar to the flow rate-idle time equations, i.e. one that requires quantities that are either given system parameters or that can be computed from other two-machine models in the decomposition.

These resumption of flow equations reflect the perspective of an observer in a buffer. His perspective depends on whether he is up- or downstream of a split or merge system. For this reason, two different sets of resumption of flow equations have to be derived. However, the approach is the same in all cases and the results show a common structure. In this subsection, split operations are analyzed. Merge operations are studied in the next subsection.

The resumption of flow equations for a purely linear transfer line as depicted in Figure 2 are given in [Gershwin, 1987, Gershwin, 1994]. The first new component which we model in Figure 6 consists of a machine M_i which has exactly one upstream machine denoted as M_j and multiple downstream machines.

Each part processed at Machine M_i in Figure 6 is randomly routed to one of Machine M_i's downstream machines. To produce a part at time t and send it to Machine M_m with $(i,m) \in D(i)$, several conditions must hold simultaneously. First, Machine M_i must be up at time t. Second, it must not be starved, i.e. the level $n[(j,i),t-1]$ of its one and only upstream buffer $B_{j,i}$ must be positive. Third, it must not be blocked. Since we assume

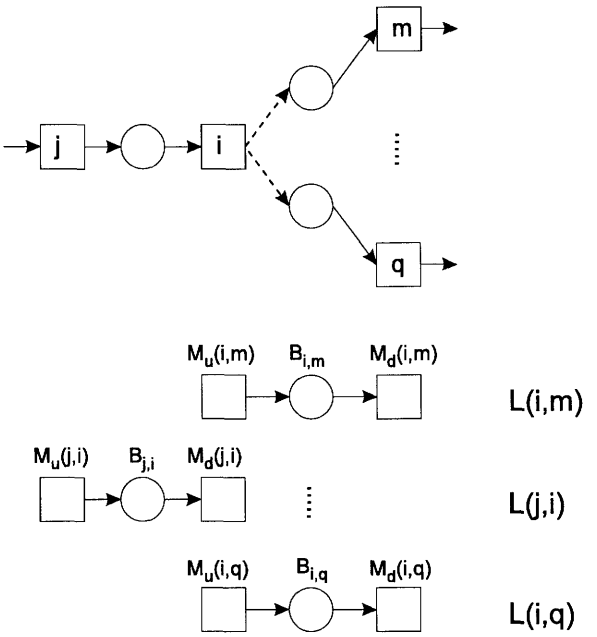

Figure 6: Split System and its Decomposition

blocking after service this means that there must not be an already processed part waiting at the workspace of Machine M_i for its selected output buffer to become non-full. In the two-machine model used for the decomposition, the workspaces at the two machines are included in the extended buffer size: $N(i,m) = C_{i,m} + 2$. Thus, if Machine M_i is blocked due to an already processed part which is waiting for a space in Buffer $B_{i,m}$, we have $n(i,m) = N(i,m)$ for Line $L(i,m)$. For this reason, Machine M_i is not blocked if we have $n[(i,q), t-1] < N(i,q), \forall (i,q) \in D(i)$. Finally, the part produced at time t must be sent to Machine M_m instead of being sent to one of the other machines M_q with $(i,q) \in D(i), q \neq m$.

2.5.1 Upstream Machine

To an observer in Buffer $B_{i,m}$, the virtual upstream machine $M_u(i,m)$ is up at time t when a part enters Buffer $B_{i,m}$ at time t or when Buffer $B_{i,m}$ is full and $M_u(i,m)$ is blocked (since a blocked machine cannot fail). For this to happen, Machine M_i must be up, it must not be starved or blocked due to a full buffer $B_{i,q}, q \neq m$, and the processed part must be sent to Machine M_m.

Define $\{\alpha_u[(i,m),t]=1\}$ as the event that the virtual upstream machine $M_u(i,m)$ is up at time t:

$$\begin{aligned}\{\alpha_u[(i,m),t]=1\} \quad \text{iff} \quad & \{\alpha_i(t)=1\} \text{ and} \\ & \{n[(j,i),t-1]>0\} \text{ and} \\ & \{n[(i,q),t-1]<N(i,q),\ \forall(i,q)\in D(i), q\neq m\} \text{ and} \\ & \{\beta_i(t)=m\}.\end{aligned} \tag{18}$$

The first condition on the right hand side of (18) says that Machine M_i must be up. Since we assume in our model that a split machine has exactly one input buffer denoted as $B_{j,i}$, the second condition demands that this input buffer is not empty. The third condition says that Machine M_i must not be blocked due to a part which is waiting for a space in Buffer $B_{i,q}$ $(q \neq m)$, i.e. one of the other buffers downstream of Machine M_i. (If Machine M_i is blocked due to a part which is waiting for a space in Buffer $B_{i,m}$, then Machine $M_u(i,m)$ as seen from an observer in Buffer $B_{i,m}$ is *up* as it is trying to deliver a part.) Finally, the part produced at time t must be randomly routed to Machine M_m. This is denoted as the event $\{\beta_i(t)=m\}$.

Machine $M_u(i,m)$ is down if it is not up, i.e:

$$\begin{aligned}\{\alpha_u[(i,m),t]=0\} \quad \text{iff} \quad & \{\alpha_i(t)=0\} \text{ or} \\ & \{n[(j,i),t-1]=0\} \text{ or} \\ & \{n[(i,q),t-1]=N(i,q), \\ & \text{ for some } (i,q)\in D(i), q\neq m\} \text{ or} \\ & \{\beta_i(t)=q, \text{ for some } (i,q)\in D(i), q\neq m\}\end{aligned} \tag{19}$$

In (19), the different events which can force Machine $M_u(i,m)$ down are approximately mutually disjoint: If Machine M_i is either starved or blocked it can neither fail nor send a part to a M_q with $(i,q)\in D(i), q\neq m$. If a part is sent to Machine M_q, M_i cannot be down. Finally, we have already assumed in the derivation of the flow rate-idle time equation on Page 19 that the probability of a machine being blocked and starved simultaneously is very small and can be neglected in an approximation.

The repair probability $r_u(i,m)$ of the virtual upstream machine $M_u(i,m)$ is the probability of seeing a part being sent into Buffer $B_{i,m}$ at time $t+1$

given that no part was sent into Buffer $B_{i,m}$ at time t and that Machine M_i was not blocked due to a part for Buffer $B_{i,m}$, i.e. $n[(i,m),t-1] < N(i,m)$. This can be written as

$$\begin{aligned} r_u(i,m) &= \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \{\alpha_u[(i,m),t]=0\} \text{ and} \\ &\quad \{n[(i,m),t-1] < N(i,m)\}\Big] \qquad (20) \\ &= \text{prob}\Bigg[\{\alpha_u[(i,m),t+1]=1\} \mid \Big\{\{\alpha_i(t)=0\} \text{ or} \\ &\quad \{n[(j,i),t-1]=0\} \text{ or} \\ &\quad \{n[(i,q),t-1]=N(i,q), \text{ for some } (i,q)\in D(i), q\neq m\} \text{ or} \\ &\quad \{\beta_i(t)=q, \text{ for some } (i,q)\in D(i), q\neq m\}\Big\} \text{ and} \\ &\quad \{n[(i,m),t-1] < N(i,m)\}\Bigg] \qquad (21) \end{aligned}$$

if we use the definition of an upstream machine being down (19). Since the different events which force Machine $M_u(i,m)$ down are approximately mutually disjoint, we can break equation (21) down by decomposing the conditioning event to find:

$$\begin{aligned} r_u(i,m) &\approx A(i,m)W(i,m) + B(i,m)X(i,m) \\ &\quad + \sum_{(i,q)\in D(i), q\neq m} \Big[C_{i,q}(i,m)Y_{i,q}(i,m) + D_{i,q}(i,m)Z_{i,q}(i,m)\Big] \qquad (22) \end{aligned}$$

where we define

$$\begin{aligned} A(i,m) &= \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \\ &\quad \{\alpha_i(t)=0\} \text{ and } \{n[(i,m),t-1] < N(i,m)\}\Big] \qquad (23) \end{aligned}$$

$$\begin{aligned} W(i,m) &= \text{prob}\Big[\{\alpha_i(t)=0\} \text{ and } \{n[(i,m),t-1] < N(i,m)\} \mid \\ &\quad \{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1] < N(i,m)\}\Big] \qquad (24) \end{aligned}$$

$$B(i,m) = \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \{n[(j,i),t-1]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (25)$$

$$X(i,m) = \text{prob}\Big[\{n[(j,i),t-1]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\} \mid \{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (26)$$

$$C_{i,q}(i,m) = \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (27)$$

$$Y_{i,q}(i,m) = \text{prob}\Big[\{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(i,m),t-1]<N(i,m)\} \mid \{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (28)(29)$$

$$D_{i,q}(i,m) = \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \{\beta_i(t)=q\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (30)$$

$$Z_{i,q}(i,m) = \text{prob}\Big[\{\beta_i(t)=q\} \text{ and } \{n[(i,m),t-1]<N(i,m)\} \mid \{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \quad (31)$$

We now determine the conditional probabilities. Probabilities $A(i,m)$ and $W(i,m)$ deal with a possible failure of Machine M_i itself. In (23), $A(i,m)$ is the probability that flow resumes into Buffer $B_{i,m}$ at time $t+1$ given that Machine M_i was down at time t. For this to happen, Machine M_i must be repaired (with probability r_i) and the part which is then produced must be sent into Buffer $B_{i,m}$ (with probability $d_{i,m}$), i.e.

$$A(i,m) = r_i d_{i,m} \quad (32)$$

In (24), $W(i,m)$ is the probability that a failure of the virtual machine $M_u(i,m)$ is due to a failure of Machine M_i. It can be expressed in terms of the conditional probabilities of all the other events which may lead to a failure of Machine $M_u(i,m)$

$$W(i,m) = 1 - X(i,m) - \sum_{(i,q)\in D(i), q\neq m} (Y_{i,q}(i,m) + Z_{i,q}(i,m)) \qquad (33)$$

since all these conditional probabilities add up to 1.

Probabilities $B(i,m)$ and $X(i,m)$ account for starvation of Machine M_i. In (25), $B(i,m)$ is the probability that flow resumes into Buffer $B_{i,m}$ at time $t+1$, given that Machine M_i was starved at time t. The reason for an empty upstream buffer is an upstream machine failure. Therefore, the virtual upstream machine $M_u(j,i)$ must be repaired and Buffer $B_{i,m}$ must be selected, so

$$B(i,m) = r_u(j,i)d_{i,m} \qquad (34)$$

In (26), $X(i,m)$ is the probability that Machine M_i is starved given that Machine $M_u(i,m)$ is down. Since event $\{n[(j,i),t-1]=0\}$ implies event $\{\alpha_u[(i,m),t]=0\}$, we can write

$$\begin{aligned} X(i,m) \;=\; & \text{prob}\Big[\{n[(j,i),t-1]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\} \\ & \text{and } \{\alpha_u[(i,m),t]=0\} \mid \\ & \{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \end{aligned} \qquad (35)$$

Using the definition of conditional probability, this can be written as a quotient, i.e.

$$\begin{aligned} X(i,m) \;=\; & \text{prob}\left[\begin{array}{l}\{n[(j,i),t-1]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\} \\ \text{and } \{\alpha_u[(i,m),t]=0\}\end{array}\right] : \\ & \text{prob}\Big[\{\alpha_u[(i,m),t]=0\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \end{aligned} \qquad (36)$$

Since we assume that the probability of Machine M_i being starved and blocked simultaneously is negligible, the event $\{n[(j,i),t-1]=0\}$ implies $\{n[(i,m),t-1]<N(i,m)\}$. It furthermore implies that Machine $M_u(j,i)$ is down as the only reason for Machine M_i to be starved is a failure of $M_u(j,i)$. For this reason, the numerator in (36) is approximately $\mathbf{p}[(j,i);001]$ where $\mathbf{p}[(j,i);n\alpha_u\alpha_d]$ denotes the probability of finding the virtual upstream machine $M_u(j,i)$ in Line $L(j,i)$ in state α_u, the downstream machine $M_d(j,i)$ in state α_d and the buffer at level n.

As there must be exactly one repair for each failure, the following equation [Gershwin, 1994, p. 81-82] for the two-machine model by Gershwin and Schick holds exactly as in the transfer line decomposition [Gershwin, 1987]

$$\begin{aligned} & r_u(i,m)\ \text{prob}\Big[\{\alpha_u[(i,m),t]=0\}\ \text{and}\ \{n[(i,m),t-1]<N(i,m)\}\Big] \\ &= p_u(i,m)\ \text{prob}\Big[\{\alpha_u[(i,m),t]=1\}\ \text{and}\ \{n[(i,m),t-1]<N(i,m)\}\Big] \\ &= p_u(i,m)E(i,m) \end{aligned} \tag{37}$$

where $E(i,m)$ is the production rate of Line $L(i,m)$ in the decomposition. The denominator in (36) can hence be written as:

$$\begin{aligned} & \text{prob}\Big[\{\alpha_u[(i,m),t]=0\}\ \text{and}\ \{n[(i,m),t-1]<N(i,m)\}\Big] \\ & \qquad = \frac{p_u(i,m)E(i,m)}{r_u(i,m)} \end{aligned} \tag{38}$$

Using these expressions for the numerator and denominator of (35), we find:

$$X(i,m) = \frac{\mathbf{p}[(j,i);001]r_u(i,m)}{p_u(i,m)E(i,m)} \tag{39}$$

In (28), $C_{i,q}(i,m)$ is the probability that flow resumes into Buffer $B_{i,m}$ after a blockage of Machine M_i due to a failure of Machine $M_d(i,q)$. It is the repair probability of $M_d(i,q)$ times the probability of sending the part then processed at Machine M_i to M_m:

$$C_{i,q}(i,m) = r_d(i,q)d_{i,m} \tag{40}$$

In (29), $Y_{i,q}(i,m)$ is the probability that Machine M_i is blocked due to a full buffer $B_{i,q}$, given that $M_u(i,m)$ is down. The expression for $Y_{i,q}(i,m)$ is derived like the one for $X(i,m)$ to find:

$$Y_{i,q}(i,m) = \frac{\mathbf{p}[(i,q);N(i,q)10]r_u(i,m)}{p_u(i,m)E(i,m)} \tag{41}$$

We now derive an expression for $Z_{i,q}(i,m)$, the probability that a failure of Machine $M_u(i,m)$ is due to sending a part from Machine M_i to $M_q, q \neq m$.

Since event $\{\beta_i(t) = q\}$ implies events $\{\alpha_u[(i,m),t] = 0\}$ and $\{n[(i,m),t-1] < N(i,m)\}$, we can write

$$Z_{i,q}(i,m) = \text{prob}\left[\begin{array}{c} \{\beta_i(t) = q\} \text{ and } \{n[(i,m),t-1] < N(i,m)\} \\ \text{and } \{\alpha_u[(i,m),t] = 0\}| \\ \{\alpha_u[(i,m),t] = 0\} \text{ and } \{n[(i,m),t-1] < N(i,m)\} \end{array}\right]. \quad (42)$$

Using the definition of conditional probability and equation (37), this can be written as

$$\begin{aligned} Z_{i,q}(i,m) &= \text{prob}\left[\begin{array}{l} \{\beta_i(t) = q\} \text{ and } \{n[(i,m),t-1] < N(i,m)\} \\ \text{and } \{\alpha_u[(i,m),t] = 0\} \end{array}\right] : \\ &\quad \text{prob}\Big[\{\alpha_u[(i,m),t] = 0\} \text{ and } \{n[(i,m),t-1] < N(i,m)\}\Big] \\ &= \text{prob}[\{\beta_i(t) = q\}]\frac{r_u(i,m)}{p_u(i,m)E(i,m)}. \end{aligned} \quad (43)$$

To determine prob$[\{\beta_i(t) = q\}]$, we first analyze in Table 2 the implications of sending a part from Machine M_i to M_q at time t, i.e. the event $\{\beta_i(t) = q\}$: If at time t a part is processed by Machine M_i and sent to M_q, the virtual upstream machine $M_u(i,q)$ seen by an observer in the corresponding buffer $B_{i,q}$ is up. Since Machine M_i is not blocked at time t, this implies that the buffer level $n[(i,q),t-1]$ is in the interval $[0, N(i,q)-1]$. However, there is no implication concerning the state of the virtual downstream machine $M_d(i,q)$. These implications are summarized in the first row of Table 2.

Line	Buffer Level	Machine State
$L(i,q)$	$0 \le n[(i,q),t-1] \le N(i,q)-1$	$\alpha_u[(i,q),t] = 1$, $\alpha_d[(i,q),t] \in \{0,1\}$
$L(i,k)$, $k \neq q$, $k \neq m$	$0 \le n[(i,k),t-1] \le N(i,k)-1$	$\alpha_u[(i,k),t] = 0$, $\alpha_d[(i,k),t] \in \{0,1\}$
$L(j,i)$	$1 \le n[(j,i),t-1] \le N(j,i)$	$\alpha_u[(j,i),t] \in \{0,1\}$, $\alpha_d[(j,i),t] = 1$

Table 2: Implications of Event $\{\beta_i(t) = q\}$

If the part processed at Machine M_i at time t is sent to Machine M_q with $q \neq m$, then no part is sent into any other buffer $B_{i,k}$ downstream of M_i. For this reason, the virtual upstream machine $M_u(i,k)$ corresponding to any line $L(i,k)$ is down. The buffer corresponding to Line $L(i,k)$ cannot be full since Machine M_i is not blocked. However, nothing is implied with respect to Machine $M_d(i,k)$. It can be up or down. See the second row of Table 2.

The last set of implications we have to study are those on the single upstream line $L(j,i)$. If Machine M_i is processing a part, then the virtual downstream machine $M_d(j,i)$ corresponding to Line $L(j,i)$ is up and not starved, i.e. the buffer level $n[(j,i),t-1]$ is in the interval $[1, N(j,i)]$. However, nothing is implied with respect to the state of Machine $M_u(j,i)$.

We use the implications listed in Table 2 to write:

$$\text{prob}[\{\beta_i(t) = q\}] = \text{prob}\left[\begin{array}{l} \{0 \leq n[(i,q),t-1] \leq N(i,q)-1\} \text{ and} \\ \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] \in \{0,1\}\} \\ \qquad \text{and} \\ \left\{\begin{array}{l} \{0 \leq n[(i,k),t-1] \leq N(i,k)-1\} \text{ and} \\ \{\alpha_u[(i,k),t] = 0\} \text{ and } \{\alpha_d[(i,k),t] \in \{0,1\}\}, \\ \forall (i,k) \in D(i), k \neq m, k \neq q\} \end{array}\right. \\ \qquad \text{and} \\ \{1 \leq n[(j,i),t-1] \leq N(j,i)\} \text{ and} \\ \{\alpha_u[(j,i),t] \in \{0,1\}\} \text{ and } \{\alpha_d[(j,i),t] = 1\} \end{array}\right]$$

We approximate the probability on the right hand side of this equation by treating the events for Line $L(i,q)$, Lines $L(i,k), k \neq m, k \neq q$, and Line $L(j,i)$ as if they were independent:

$$\begin{array}{l} \text{prob}[\{\beta_i(t) = q\}] \\ \quad \approx \quad \text{prob}\left[\begin{array}{l} \{0 \leq n[(i,q),t-1] \leq N(i,q)-1\} \text{ and} \\ \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] \in \{0,1\}\} \end{array}\right] \cdot \\ \qquad \text{prob}\left[\begin{array}{l} \left\{\begin{array}{l} \{0 \leq n[(i,k),t-1] \leq N(i,k)-1\} \text{ and} \\ \{\alpha_u[(i,k),t] = 0\} \text{ and } \{\alpha_d[(i,k),t] \in \{0,1\}\}, \\ \forall (i,k) \in D(i), k \neq m, k \neq q\} \end{array}\right. \end{array}\right] \cdot \end{array}$$

$$\text{prob}\left[\begin{array}{l}\{1 \le n[(j,i),t-1] \le N(j,i)\} \text{ and} \\ \{\alpha_u[(j,i),t] \in \{0,1\}\} \text{ and } \{\alpha_d[(j,i),t] = 1\}\end{array}\right] \tag{44}$$

These three probabilities can be obtained from the solutions of the two-machine lines $L(i,q)$, $L(i,k)$, and $L(j,i)$. We have to add over the probabilities of the respective states in each of the two-machine lines. In the case of Line $L(i,q)$ which receives the part, these are all states where the upstream machine is up and not blocked. We find

$$\begin{aligned}&\text{prob}\left[\begin{array}{l}\{0 \le n[(i,q),t-1] \le N(i,q)-1\} \text{ and} \\ \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] \in \{0,1\}\}\end{array}\right] \\ &= \left[\sum_{n=0}^{N(i,q)-1} \mathbf{p}[(i,q);n11] + \mathbf{p}[(i,q);n10]\right]\end{aligned} \tag{45}$$

for Line $L(i,q)$. As an approximation, we treat Lines $L(i,k), k \neq m, k \neq q$, as if they were independent and add over all states where the upstream machine is down and not blocked:

$$\begin{aligned}&\text{prob}\left[\begin{array}{l}\Big\{\{0 \le n[(i,k),t-1] \le N(i,k)-1\} \text{ and} \\ \quad \{\alpha_u[(i,k),t] = 0\} \text{ and } \{\alpha_d[(i,k),t] \in \{0,1\}\}, \\ \quad \forall (i,k) \in D(i), k \neq m, k \neq q\Big\}\end{array}\right] \\ &\approx \prod_{\substack{(i,k)\in D(i) \\ k\neq m \\ k\neq q}} \text{prob}\left[\begin{array}{l}\{0 \le n[(i,k),t-1] \le N(i,k)-1\} \text{ and} \\ \{\alpha_u[(i,k),t] = 0\} \text{ and } \{\alpha_d[(i,k),t] \in \{0,1\}\}\end{array}\right] \\ &= \prod_{\substack{(i,k)\in D(i) \\ k\neq m \\ k\neq q}} \sum_{n=0}^{N(i,k)-1} [\mathbf{p}[(i,k);n00] + \mathbf{p}[(i,k);n01]]\end{aligned} \tag{46}$$

For the single Line $L(j,i)$ immediately upstream of Machine M_i, we add over all states where the downstream machine is up and the buffer not empty, i.e.

$$\begin{aligned}
&\text{prob}\left[\begin{array}{l}\{1 \le n[(j,i),t-1] \le N(j,i)\} \text{ and} \\ \{\alpha_u[(j,i),t] \in \{0,1\}\} \text{ and } \{\alpha_d[(j,i),t]=1\}\end{array}\right] \\
&= \left[\sum_{n=1}^{N(j,i)} \mathbf{p}[(j,i);n11] + \mathbf{p}[(j,i);n01]\right] \qquad (47)
\end{aligned}$$

and therefore

$$\begin{aligned}
&\text{prob}[\{\beta_i(t)=q\}] \\
&\approx \left[\sum_{n=0}^{N(i,q)-1} \mathbf{p}[(i,q);n11] + \mathbf{p}[(i,q);n10]\right] \cdot \\
&\left[\prod_{\substack{(i,k)\in D(i)\\ k\neq m\\ k\neq q}} \sum_{n=0}^{N(i,k)-1} \mathbf{p}[(i,k);n00] + \mathbf{p}[(i,k);n01]\right] \cdot \\
&\left[\sum_{n=1}^{N(j,i)} \mathbf{p}[(j,i);n11] + \mathbf{p}[(j,i);n01]\right] \qquad (48)
\end{aligned}$$

which eventually leads to

$$\begin{aligned}
&Z_{i,q}(i,m) \\
&\approx \left[\sum_{n=0}^{N(i,q)-1} \mathbf{p}[(i,q);n11] + \mathbf{p}[(i,q);n10]\right] \cdot \\
&\left[\prod_{\substack{(i,k)\in D(i)\\ k\neq m\\ k\neq q}} \sum_{n=0}^{N(i,k)-1} \mathbf{p}[(i,k);n00] + \mathbf{p}[(i,k);n01]\right] \cdot \\
&\left[\sum_{n=1}^{N(j,i)} \mathbf{p}[(j,i);n11] + \mathbf{p}[(j,i);n01]\right] \cdot \\
&\frac{r_u(i,m)}{p_u(i,m)E(i,m)} \qquad (49)
\end{aligned}$$

$$
\begin{aligned}
a_1 &= \{0 \le n[(i,q),t-1] \le N(i,q)-2\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t]=1\} \text{ and } \{\alpha_d[(i,q),t] \in \{0,1\}\} \\
a_2 &= \{n[(i,q),t-1]=N(i,q)-1\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t]=1\} \text{ and } \{\alpha_d[(i,q),t]=1\} \\
a_3 &= \{n[(i,q),t-1]=N(i,q)-1\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t]=1\} \text{ and } \{\alpha_d[(i,q),t]=0\}
\end{aligned}
$$

$$
\begin{aligned}
b &= \{0 \le n[(i,k),t-1]=N(i,k)-1\} \text{ and} \\
&\quad \{\alpha_u[(i,k),t]=0\} \text{ and } \{\alpha_d[(i,k),t] \in \{0,1\}\}, \\
&\quad (i,k) \in D(i), k \ne m, k \ne q
\end{aligned}
$$

$$
\begin{aligned}
c_1 &= \{2 \le n[(j,i),t-1] \le N(j,i)\} \text{ and} \\
&\quad \{\alpha_u[(j,i),t] \in \{0,1\}\} \text{ and } \{\alpha_d[(j,i),t]=1\} \\
c_2 &= \{n[(j,i),t-1]=1\} \text{ and} \\
&\quad \{\alpha_u[(j,i),t]=1\} \text{ and } \{\alpha_d[(j,i),t]=1\} \\
c_3 &= \{n[(j,i),t-1]=1\} \text{ and} \\
&\quad \{\alpha_u[(j,i),t]=0\} \text{ and } \{\alpha_d[(j,i),t]=1\}
\end{aligned}
$$

$$
\begin{aligned}
e &= \{\alpha_u[(i,m),t+1]=1\} \\
f &= \{n[(i,m),t-1]<N(i,m)\}
\end{aligned}
$$

Table 3: Definition of Auxiliary Events

$Z_{i,q}(i,m)$ is now expressed in terms of quantities of the two-machine models in the decomposition that are related to Machine M_i and that are available in the course of the decomposition.

We now have to find an expression for the conditional probability $D_{i,q}(i,m)$

$$
\begin{aligned}
D_{i,q}(i,m) &= \text{prob}\Big[\{\alpha_u[(i,m),t+1]=1\} \mid \\
&\quad \{\beta_i(t)=q\} \text{ and } \{n[(i,m),t-1]<N(i,m)\}\Big] \qquad (50)
\end{aligned}
$$

of processing a part at Machine M_i at time $t+1$ and sending it to M_m, given that at time t a part was sent to Machine $M_q, q \neq m$. The implications of the conditioning event $\{\beta_i(t) = q\}$ are given in Table 2 on Page 28. To reduce the notational effort when decomposing the conditioning event, auxiliary events $a_1, a_2, a_3, b, c_1, c_2$, and c_3 are defined in Table 3.

The mutually exclusive and collectively exhaustive events a_1, a_2, and a_3 describe the possible states of Line $L(i,q)$ that are implied by event $\{\beta_i(t) = q\}$. In a similar manner, the auxiliary event b describes the possible states of Lines $L(i,k)$ and c_1, c_2, and c_3 those of Line $L(j,i)$. Given the definition of the auxiliary variables, we find

$$\text{prob}[\{\beta_i(t) = q\}] = \text{prob}[\{a_1 \text{ or } a_2 \text{ or } a_3\} \text{ and } b \text{ and } \{c_1 \text{ or } c_2 \text{ or } c_3\}] \quad (51)$$

and therefore

$$\begin{aligned} D_{i,q}(i,m) &= \text{prob}\Big[\, e \,\Big|\{a_1 \text{ or } a_2 \text{ or } a_3\} \text{ and } b \text{ and } \{c_1 \text{ or } c_2 \text{ or } c_3\} \text{ and } f\Big] \\ &= \text{prob}\,\Big[e \,\Big|\, a_1\, b\, c_1\, f \;\text{ or }\; a_1\, b\, c_2\, f \;\text{ or }\; a_1\, b\, c_3\, f \;\text{ or} \\ &\qquad\qquad a_2\, b\, c_1\, f \;\text{ or }\; a_2\, b\, c_2\, f \;\text{ or }\; a_2\, b\, c_3\, f \;\text{ or} \\ &\qquad\qquad a_3\, b\, c_1\, f \;\text{ or }\; a_3\, b\, c_2\, f \;\text{ or }\; a_3\, b\, c_3\, f\,\Big] \end{aligned}$$

Each of the events $a_k\, b\, c_l\, f$ with $k, l \in \{1,2,3\}$ implies event $\{\beta_i(t) = q\}$ and each of the events $a_k, k \in \{1,2,3\}$ implies event f, i.e. $\{n[(i,m), t-1] < N(i,m)\}$. As an approximation, we assume that the events related to different lines are independent. In this case, $D_{i,q}(i,m)$ can be decomposed to find

$$\begin{aligned} D_{i,q}(i,m) = \Big[&\text{prob}[\, e \mid a_1\, b\, c_1\,]\ \text{prob}[\, a_1\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_1\,] + \\ &\text{prob}[\, e \mid a_1\, b\, c_2\,]\ \text{prob}[\, a_1\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_2\,] + \\ &\text{prob}[\, e \mid a_1\, b\, c_3\,]\ \text{prob}[\, a_1\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_3\,] + \\ &\text{prob}[\, e \mid a_2\, b\, c_1\,]\ \text{prob}[\, a_2\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_1\,] + \\ &\text{prob}[\, e \mid a_2\, b\, c_2\,]\ \text{prob}[\, a_2\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_2\,] + \\ &\text{prob}[\, e \mid a_2\, b\, c_3\,]\ \text{prob}[\, a_2\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_3\,] + \\ &\text{prob}[\, e \mid a_3\, b\, c_1\,]\ \text{prob}[\, a_3\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_1\,] + \\ &\text{prob}[\, e \mid a_3\, b\, c_2\,]\ \text{prob}[\, a_3\,]\ \text{prob}[\, b\,]\ \text{prob}[\, c_2\,] + \end{aligned}$$

$$\text{prob}[\ e\ |\ a_3\, b\, c_3\ ]\ \text{prob}[\ a_3\ ]\ \text{prob}[\ b\ ]\ \text{prob}[\ c_3\ ]\Big] \cdot \frac{1}{\text{prob}[\{\beta_i(t) = q\}]}. \tag{52}$$

The conditional probabilities in (52) can be determined using the information in Table 3 on Page 32. The first conditional probability, prob[e | $a_1\, b\, c_1$], is the probability that Machine $M_u(i,m)$ is up at time $t+1$, given that the buffer level in Line $L(i,q)$ which receives the part at time t, is below $N(i,q)-1$ at time $t-1$ (event a_1), that none of the buffers in any of the lines $L(i,k)$ is full (event b), and that $n[(j,i),t-1] > 1$ holds for the input buffer level (event c_1).

The following has to happen in order for Machine $M_u(i,m)$ to be up at time $t+1$:

- Machine M_i in the real system must not fail, with probability $(1-p_i)$.
- The part must be sent to Machine M_m, with probability $d_{i,m}$.

Given the buffer levels in Lines $L(i,q)$, $L(i,k)$, and $L(j,i)$, Machine M_i cannot be blocked or starved at time $t+1$. Since routing decisions and machine failures are assumed to be independent, we find

$$\text{prob}[\ e\ |\ a_1\, b\, c_1\ ] = (1-p_i)\, d_{i,m}. \tag{53}$$

In the second conditional probability in (52), prob[e | $a_1\, b\, c_2$], event c_2 says that Buffer $B_{j,i}$ which looses a part at time t is almost empty at time $t-1$, i.e. $n[(j,i),t-1] = 1$, and that Machine $M_u(j,i)$ is up at time t. In this situation, an additional condition must be met in order for Machine $M_u(i,m)$ to be up at time $t+1$:

- Machine $M_u(j,i)$ must not fail (because otherwise Machine M_i would be starved), with probability $(1-p_u(j,i))$.

The second conditional probability is therefore

$$\text{prob}[\ e\ |\ a_1\, b\, c_2\ ] = (1-p_i)\, d_{i,m}\, (1-p_u(j,i)). \tag{54}$$

In the third conditional probability in (52), prob[e | $a_1\, b\, c_3$], event c_3 says that Buffer $B_{j,i}$ is almost empty at time $t-1$, and that Machine $M_u(j,i)$

is down at time t. If $M_u(j,i)$ is down at time t, it has to be repaired in order for $M_u(i,m)$ to be up at time $t+1$, with probability $r_u(j,i)$, so

$$\text{prob}[\, e \mid a_1\, b\, c_1 \,] = (1-p_i)\, d_{i,m}\, r_u(j,i)). \tag{55}$$

The derivation for the six remaining conditional probabilities is completely analogous and we find:

$$\begin{aligned}
\text{prob}[\, e \mid a_2\, b\, c_1 \,] &= (1-p_i)\, d_{i,m}\, (1-p_d(i,q)) \\
\text{prob}[\, e \mid a_2\, b\, c_2 \,] &= (1-p_i)\, d_{i,m}\, (1-p_d(i,q))\, (1-p_u(j,i)) \\
\text{prob}[\, e \mid a_2\, b\, c_3 \,] &= (1-p_i)\, d_{i,m}\, (1-p_d(i,q))\, r_u(j,i)) \\
\text{prob}[\, e \mid a_3\, b\, c_1 \,] &= (1-p_i)\, d_{i,m}\, r_d(i,q) \\
\text{prob}[\, e \mid a_3\, b\, c_2 \,] &= (1-p_i)\, d_{i,m}\, r_d(i,q)\, (1-p_u(j,i)) \\
\text{prob}[\, e \mid a_3\, b\, c_3 \,] &= (1-p_i)\, d_{i,m}\, r_d(i,q)\, r_u(j,i))
\end{aligned}$$

The special structure of these nine conditional probabilities allows to factor (52). This yields

$$\begin{aligned}
&D_{i,q}(i,m) \\
&= (1-p_i) d_{i,m} \\
&\quad \Big[\, \text{prob}[\, a_1 \,] + (1-p_d(i,q))\ \text{prob}[\, a_2 \,] + r_d(i,q))\ \text{prob}[\, a_3 \,] \Big] \cdot \\
&\quad \text{prob}[\, b \,] \cdot \\
&\quad \Big[\, \text{prob}[\, c_1 \,] + (1-p_u(j,i))\ \text{prob}[\, c_2 \,] + r_u(j,i))\ \text{prob}[\, c_3 \,] \Big] \cdot \\
&\quad \frac{1}{\text{prob}[\{\beta_i(t) = q\}]}.
\end{aligned} \tag{56}$$

Using the information in Table 3 on Page 32, the unconditional probabilities can be expressed as

$$\begin{aligned}
\text{prob}[\, a_1 \,] &= \sum_{n=0}^{N(i,q)-2} \Big(\mathbf{p}[(i,q); n11] + \mathbf{p}[(i,q); n10] \Big) \\
\text{prob}[\, a_2 \,] &= \mathbf{p}[(i,q); N(i,q)-1, 11] \\
\text{prob}[\, a_3 \,] &= \mathbf{p}[(i,q); N(i,q)-1, 10]
\end{aligned}$$

$$
\begin{aligned}
\text{prob}[\, b\,] &= \Big[\prod_{\substack{(i,k)\in D(i)\\ k\neq m\\ k\neq q}} \sum_{n=0}^{N(i,k)-1} \Big(\mathbf{p}[(i,k);n00] + \mathbf{p}[(i,k);n01]\Big)\Big] \\
\text{prob}[\, c_1\,] &= \sum_{n=2}^{N(j,i)} \Big(\mathbf{p}[(j,i);n11] + \mathbf{p}[(j,i);n01]\Big) \\
\text{prob}[\, c_2\,] &= \mathbf{p}[(j,i);111] \\
\text{prob}[\, c_3\,] &= \mathbf{p}[(j,i);101]
\end{aligned}
$$

This leads to the following result

$$
D_{i,q}(i,m) \approx (1-p_i)d_{i,m}\frac{F(i,q)}{\text{prob}[\{\beta_i(t)=q\}]} \tag{57}
$$

where we define

$$
\begin{aligned}
F(i,q) &= \Big[\sum_{n=0}^{N(i,q)-2} \Big(\mathbf{p}[(i,q);n11] + \mathbf{p}[(i,q);n10]\Big) \\
&\quad + (1-p_d(i,q))\mathbf{p}[(i,q);N(i,q)-1,11] \\
&\quad + r_d(i,q)\mathbf{p}[(i,q);N(i,q)-1,10]\Big] \cdot \\
&\quad \Big[\prod_{\substack{(i,k)\in D(i)\\ k\neq m\\ k\neq q}} \sum_{n=0}^{N(i,k)-1} \Big(\mathbf{p}[(i,k);n00] + \mathbf{p}[(i,k);n01]\Big)\Big] \cdot \\
&\quad \Big[\sum_{n=2}^{N(j,i)} \Big(\mathbf{p}[(j,i);n11] + \mathbf{p}[(j,i);n01]\Big) \\
&\quad + (1-p_u(j,i))\mathbf{p}[(j,i);111] \\
&\quad + r_u(j,i)\mathbf{p}[(j,i);101]\Big]
\end{aligned} \tag{58}
$$

The fraction $\frac{F(i,q)}{\text{prob}[\{\beta_i(t)=q\}]}$ in (57) has a value between 0 and 1, as a comparison of (58) and (48) reveals. The probability $D_{i,q}(i,m)$ that Machine M_i receives a part from Machine M_i at time $t+1$, given that at time t a part was routed to Machine M_q, is therefore smaller than $(1-p_i)d_{i,m}$, as we have to take the possibility of blockage and starvation into account.

Now all components of the resumption of flow equation for the virtual upstream machine $M_u(i,m)$ are expressed in terms of parameters and performance measures of the real system or the two-machine lines in the decomposition, i.e. of quantities that are available in the course of the iterative solution of the decomposition equations. The equation can be written as

$$r_u(i,m) = \left[r_i + K_3 \frac{r_u(i,m)}{p_u(i,m)} \right] d_{i,m} \tag{59}$$

where we define

$$\begin{aligned} K_3 = & \Big[(r_u(j,i) - r_i)\mathbf{p}[(j,i);001] \\ & + \sum_{(i,q)\in D(i), q\neq m} (r_d(i,q) - r_i)\mathbf{p}[(i,q);N(i,q)10] \\ & + \sum_{(i,q)\in D(i), q\neq m} (1-p_i)F(i,q) - r_i \text{ prob}[\{\beta_i(t) = q\}] \Big] \frac{1}{E(i,m)} \end{aligned} \tag{60}$$

and $d_{i,m}$ is the fraction of parts sent from Machine M_i to M_m. The term $F(i,q)$ is given in (58) and prob$[\{\beta_i(t) = q\}]$ in (48). The factor K_3 in (60) contains parameters of two-machine lines other than Line $L(i,m)$. This is again a generalization of the corresponding equation for the transfer line model in [Gershwin, 1987].

2.5.2 Downstream Machine

In this section, the resumption of flow probability $r_d(j,i)$ for Line $L(j,i)$ upstream of Machine M_i in Figure 6 on Page 22 is derived.

To an observer in Buffer $B_{j,i}$, the virtual downstream machine $M_d(j,i)$ is up at time t when a part leaves the buffer at time t or when Buffer $B_{j,i}$ is empty and Machine $M_d(j,i)$ is starved (since a starved machine cannot fail).

Define $\{\alpha_d[(j,i),t] = 1\}$ as the event of the virtual downstream machine $M_d(j,i)$ being up at time t. Machine $M_d(j,i)$ is up if Machine M_i is up and not blocked, i.e.

$$\begin{aligned} \{\alpha_d[(j,i),t] = 1\} \quad \text{iff} \quad & \{\alpha_i(t) = 1\} \text{ and} \\ & \{n[(i,q),t-1] < N(i,q),\ \forall (i,q) \in D(i)\} \end{aligned} \tag{61}$$

Machine $M_d(j,i)$ is down if it is not up, i.e:

$$\{\alpha_d[(j,i),t]=0\} \quad \text{iff} \quad \{\alpha_i(t)=0\} \text{ or } \{n[(i,q),t-1]=N(i,q), \text{ for some } (i,q)\in D(i)\} \tag{62}$$

Note that this definition *also* describes the perspective of an observer upstream of a *disassembly machine* ([Gershwin, 1991]). In both cases, i.e. split as well as disassembly operations, Machine M_i must be up and not blocked in order for $M_d(j,i)$ to be up and one full downstream buffer $B_{i,q}$ is sufficient to block Machine M_i.

The resumption of flow probability $r_d(j,i)$ is defined as

$$r_d(j,i) = \text{prob}\left[\{\alpha_d[(j,i),t+1]=1\} \mid \{\alpha_d[(j,i),t]=0\} \text{ and } \{n[(j,i),t-1]>0\}\right] \tag{63}$$

and can again be evaluated by decomposing the conditioning event

$$r_d(j,i) = A(j,i)X(j,i) + \sum_{(i,q)\in D(i)} B_{i,q}(j,i)Y_{i,q}(j,i) \tag{64}$$

where we define

$$A(j,i) = \text{prob}\left[\{\alpha_d[(j,i),t+1]=1\} \mid \{\alpha_i(t)=0\} \text{ and } \{n[(j,i),t-1]>0\}\right] \tag{65}$$

$$X(j,i) = \text{prob}\left[\{\alpha_i(t)=0\} \text{ and } \{n[(j,i),t-1]>0\} \mid \{\alpha_d[(j,i),t]=0\} \text{ and } \{n[(j,i),t-1]>0\}\right] \tag{66}$$

$$B_{i,q}(j,i) = \text{prob}\left[\{\alpha_d[(j,i),t+1]=1\} \mid \{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(j,i),t-1]>0\}\right] \tag{67}$$

$$Y_{i,q}(j,i) = \text{prob}\left[\{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(j,i),t-1]>0\} \mid \{\alpha_d[(j,i),t]=0\} \text{ and } \{n[(j,i),t-1]>0\}\right] \tag{68}$$

We find that $A(j,i)$ in (65) is the repair probability of Machine M_i, i.e.

$$A(j,i) = r_i \tag{69}$$

and $B_{i,q}(j,i)$ in (67) is the repair probability of the blocking machine $M_d(i,q)$, i.e.

$$B_{i,q}(j,i) = r_d(i,q) \tag{70}$$

After a derivation similar to that leading to (41), we find

$$Y_{i,q}(j,i) = \frac{\mathbf{p}[(i,q);N(i,q)10]r_d(j,i)}{p_d(j,i)E(j,i)} \tag{71}$$

and

$$X(j,i) = 1 - \sum_{(i,q)\in D(i)} Y_{i,q}(j,i) \tag{72}$$

We can therefore write the resumption of flow equation in a general form using an auxiliary parameter F as

$$r_d(j,i) = r_i F + K_4 \frac{r_d(j,i)}{p_d(j,i)} \tag{73}$$

with

$$F = 1 \tag{74}$$

$$K_4 = \sum_{(i,q)\in D(i)} (r_d(i,q) - r_i)\mathbf{p}[(i,q);N(i,q)10]\frac{1}{E(j,i)} \tag{75}$$

This general form involving the auxiliary parameter F allows to formulate the resumption of flow equation for merge systems (to be derived below) in a very similar way. As in (60), we have in (75) a factor (K_4) containing parameters of two-machine lines other than Line $L(j,i)$. This is exactly the same equation as for a disassembly machine in [Gershwin, 1991, Gershwin, 1994].

2.6 Resumption of Flow Equations II: Merge Operations

The second type of subsystem depicted in Figure 7 for which we derive resumption of flow equations consists of Machine M_i which has two immediately preceding machines denoted as Machines M_{j_1} and M_{j_2}, and one immediately succeeding machine denoted as Machine M_q.

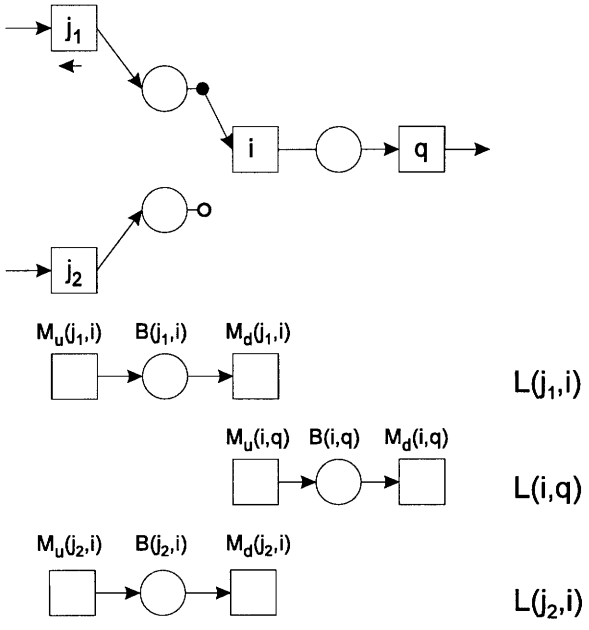

Figure 7: Merge System and its Decomposition

Buffer $B_{j_1,i}$ has priority one and $B_{j_2,i}$ has priority two. From the perspective of Machine M_i, both Machine M_{j_1} and Machine M_{j_2} produce the same type of parts. Thus, Machine M_i is starved if both of its upstream buffers are empty.

Note that this differs from the situation in an assembly system in two ways: First, in an assembly system, different types of parts are arriving from the different machines upstream of the assembly machine. In a merge system, however, the parts arriving from different upstream machines are identical. Second, an assembly machine is starved if at least one upstream buffer is empty as all the different components have to be assembled together. In a merge system, a machine is starved if all upstream buffers are empty. If one

of them is non-empty, the machine can serve this buffer and is therefore not starved.

There are three types of two-machine lines that arise in the decomposition of the system depicted in Figure 7. The perspective of an observer in Buffer $B_{i,q}$ is analyzed first.

2.6.1 Upstream Machine

In this section, the repair probability of the virtual machine $M_u(i,q)$ as seen by an observer downstream of Machine M_i is determined. To an observer in the buffer corresponding to Line $L(i,q)$ in Figure 7, the virtual machine $M_u(i,q)$ is up if Machine M_i is up and not starved, i.e.

$$\begin{aligned}\{\alpha_u[(i,q),t]=1\} \quad \text{iff} \quad & \{\alpha_i(t)=1\} \text{ and} \\ & \{n[(j,i),t-1]>0, \text{ for some } j\in\{j_1,j_2\}\}. \quad (76)\end{aligned}$$

Machine $M_u(i,q)$ is down if it is not up, i.e

$$\begin{aligned}\{\alpha_u[(i,q),t]=0\} \quad \text{iff} \quad & \{\alpha_i(t)=0\} \text{ or} \\ & \{n[(j,i),t-1]=0, \forall j\in\{j_1,j_2\}\}. \quad (77)\end{aligned}$$

The different events which can force Machine $M_u(i,q)$ down are mutually disjoint since Machine M_i cannot fail if it is starved.

We derive the resumption of flow equation in the usual way using the definition of virtual machine states:

$$\begin{aligned} r_u(i,q) &= \text{prob}\Big[\{\alpha_u[(i,q),t+1]=1\}\Big| \quad (78)\\ &\qquad \{\alpha_u[(i,q),t]=0\} \text{ and } \{n[(i,q),t-1]<N(i,q)\}\Big] \\ &= \text{prob}\Big[\{\alpha_u[(i,q),t+1]=1\}\Big| \quad (79)\\ &\qquad \Big\{\{\alpha_i(t)=0\} \text{ or} \\ &\qquad \{n[(j,i),t-1]=0, \forall j\in\{j_1,j_2\}\}\Big\} \text{ and} \\ &\qquad \{n[(i,q),t-1]<N(i,q)\}\Big] \end{aligned}$$

Since the different events which force Machine $M_u(i,q)$ down are mutually disjoint, we can break equation (79) down by decomposing the conditioning event to find

$$r_u(i,q) = A(i,q)X(i,q) + B(i,q)Y(i,q) \tag{80}$$

where we define

$$\begin{aligned} A(i,q) &= \text{prob}\Big[\{\alpha_u[(i,q),t+1]=1\}\Big| \\ &\qquad \{\alpha_i(t)=0\} \text{ and } \{n[(i,q),t-1] < N(i,q)\}\Big] \end{aligned} \tag{81}$$

$$\begin{aligned} X(i,q) &= \text{prob}\Big[\{\alpha_i(t)=0\} \text{ and } \{n[(i,q),t-1] < N(i,q)\}\Big| \\ &\qquad \{\alpha_u[(i,q),t]=0\} \text{ and } \{n[(i,q),t-1] < N(i,q)\}\Big] \end{aligned} \tag{82}$$

$$\begin{aligned} B(i,q) &= \text{prob}\Big[\{\alpha_u[(i,q),t+1]=1\}\Big| \\ &\qquad \{n[(j,i),t-1]=0, \forall j \in \{j_1,j_2\}\} \text{ and} \\ &\qquad \{n[(i,q),t-1] < N(i,q)\}\Big] \end{aligned} \tag{83}$$

$$\begin{aligned} Y(i,q) &= \text{prob}\Big[\{n[(j,i),t-1]=0, \forall j \in \{j_1,j_2\}\} \text{ and} \\ &\qquad \{n[(i,q),t-1] < N(i,q)\}\Big| \\ &\qquad \{\alpha_u[(i,q),t]=0\} \text{ and } \{n[(i,q),t-1] < N(i,q)\}\Big] \end{aligned} \tag{84}$$

In (81), $A(i,q)$ is the probability that Machine M_i is repaired, i.e.,

$$A(i,q) = r_i, \tag{85}$$

whereas $B(i,q)$ in (83) is the probability that at least one virtual upstream machine is repaired. It is expressed in terms of the probability that none of the upstream machines is repaired, i.e.,

$$B(i,q) = 1 - \prod_{\forall j \in \{j_1,j_2\}} (1 - r_u(j,i)), \tag{86}$$

since repairs are independent as they do not depend on buffer levels or the states of other machines.

The probability that Machine M_i is starved given that Machine $M_u(i,q)$ is down is

$$
\begin{aligned}
Y(i,q) \;=\; & \text{prob}\,\Big[\{n[(j,i),t-1]=0, \forall j \in \{j_1,j_2\}\} \text{ and} \\
& \{n[(i,q),t-1] < N(i,q)\}\Big] : \\
& \text{prob}\Big[\{\alpha_u[(i,q),t]=0\} \text{ and } \{n[(i,q),t-1] < N(i,q)\}\Big]
\end{aligned} \tag{87}
$$

We approximate the numerator by assuming independence:

$$
Y(i,q) \approx \prod_{\forall j \in \{j_1,j_2\}} \mathbf{p}[(j,i);001] \frac{r_u(i,q)}{p_u(i,q)E(i,q)} \tag{88}
$$

The probability $X(i,q)$ that a failure of Machine $M_u(i,q)$ is due to a failure of Machine M_i itself is expressed in terms of the probability of the single other reason why Machine $M_u(i,q)$ can be down:

$$
\begin{aligned}
X(i,q) \;&=\; 1 - Y(i,q) \\
&=\; 1 - \prod_{\forall j \in \{j_1,j_2\}} \mathbf{p}[(j,i);001] \frac{r_u(i,q)}{p_u(i,q)E(i,q)}
\end{aligned} \tag{89}
$$

We can hence express the resumption of flow equation for Machine $M_u(i,q)$ using the routing probability $d_{i,q} = 1$ as

$$
r_u(i,q) = \Big[r_i + K_3 \frac{r_u(i,q)}{p_u(i,q)}\Big] d_{i,q} \tag{90}
$$

with

$$
d_{i,q} \;=\; 1 \tag{91}
$$

$$
K_3 \;=\; \Big(1 - \prod_{\forall j \in \{j_1,j_2\}} (1 - r_u(j,i)) - r_i\Big) \prod_{\forall j \in \{j_1,j_2\}} \mathbf{p}[(j,i);001] \frac{1}{E(i,q)} \tag{92}
$$

Using the routing probability $d_{i,q} = 1$ in (90) leads to a form of the resumption of flow equation which is identical to those for an upstream machine related to a split machine (see (59) on Page 37). If there is just one line $L(j,i)$ with $(j,i) \in U(i)$, equation (90) reduces to the corresponding equation of the purely linear transfer line in [Gershwin, 1987].

2.6.2 Downstream Machine

Priority One line: To an observer in the priority one buffer $B_{j_1,i}$ of Line $L(j_1,i)$ in Figure 7 on Page 40, the virtual downstream machine $M_d(j_1,i)$ is down if Machine M_i is either down or blocked. Since we assume that a machine with two upstream buffers has exactly one downstream buffer, there is exactly one virtual downstream machine $M_d(i,q)$ which can block Machine M_i.

However, as there are two input buffers, Machine M_i can fail even if the priority one buffer is empty (and the priority two buffer is not empty). Thus, to an observer in Buffer $B_{j_1,i}$, the failures of Machine M_i do not appear to be completely operation dependent. Repairs, however, are independent of buffer levels and the repair behavior of Machine $M_d(j_1,i)$ is hence exactly as in a transfer line for which the resumption of flow equation

$$r_d(j_1,i)) = r_i F + K_4 \frac{r_d(j_1,i)}{p_d(j_1,i)} \tag{93}$$

with

$$F = 1 \tag{94}$$

$$K_4 = \Big(r_d(i,q) - r_i\Big)\mathbf{p}[(i,q); N(i,q), 10]\frac{1}{E(j_1,i)} \tag{95}$$

is given in [Gershwin, 1987]. The resumption of flow equation (93) has the same structure as (73) on Page 39.

Priority Two line: To an observer in the priority two buffer $B_{j_2,i}$, there is an additional reason to see Machine $M_d(j_2,i)$ down: The buffer of the priority one line $L(j_1,i)$ may be non-empty. In this case, the next part processed at Machine M_i will always be taken from the priority one buffer and the virtual machine $M_d(j_2,i)$ is down since nothing can be taken from the buffer of Line $L(j_2,i)$. Thus, three conditions must hold for Machine $M_d(j_2,i)$ to be up: Machine M_i must be up, it must not be blocked by its virtual downstream machine $M_d(i,q)$, and its priority one upstream buffer $B_{j_1,i}$ must be empty. Formally,

$$\{\alpha_d[(j_2,i),t]=1\} \quad \text{iff} \quad \{\alpha_i(t)=1\} \text{ and } \{n[(i,q),t-1]<N(i,q)\} \text{ and } \{n[(j_1,i),t-1]=0\}.$$

Machine $M_d(j_2,i)$ is down if it is not up:

$$\{\alpha_d[(j_2,i),t]=0\} \quad \text{iff} \quad \{\alpha_i(t)=0\} \text{ or } \{n[(i,q),t-1]=N(i,q)\} \text{ or } \{n[(j_1,i),t-1]>0\}. \tag{96}$$

The different reasons for Machine $M_d(j_2,i)$ to be down as given in (96) are not disjoint. Table 4 gives the six mutually exclusive and collectively exhaustive reasons for Machine $M_d(j_2,i)$ to be down at time t. In three out of the six cases, it is not possible that Machine $M_d(j_2,i)$ is up at time $t+1$.

In Case 1, M_i is down and not blocked while the priority one buffer is empty. Since M_i might be repaired, it is possible that the virtual machine $M_d(j_2,i)$ which is seen by an observer in the priority two buffer is up at time $t+1$.

Case 2 is a situation where $M_d(j_2,i)$ is down for two reasons: M_i is down and the priority one buffer $B_{j_1,i}$ is non-empty. Even if Machine M_i were repaired at time $t+1$, the priority one buffer would still be non-empty as nothing was removed at time t. For this reason it is not possible that Machine $M_d(j_2,i)$ is up at time $t+1$.

Case	Machine M_i at time t	Buffer $B_{i,q}$ at time $t-1$	Buffer $B_{j_1,i}$ at time $t-1$	$M_d(j_2,i)$ up at time $t+1$?
1	down	not full	empty	possible
2	down	not full	not empty	impossible
3	up	full	empty	possible
4	up	full	not empty	impossible
5	up	not full	$n[(j_1,i),t-1]=1$	possible
6	up	not full	$n[(j_1,i),t-1]>1$	impossible

Table 4: Disjoint Reasons for Machine $M_d(j_2,i)$ to be Down

Machine M_i is blocked at time t in Case 3 and the priority one buffer is empty. Since the blocking machine $M_d(i,q)$ might be repaired, it is possible that Machine $M_d(j_2,i)$ is up at time $t+1$.

In Case 4, however, Machine M_i is blocked and the priority one buffer is non-empty. Even if Machine $M_d(i,q)$ were repaired at time $t+1$, Machine $M_d(j_2,i)$ would still be down due to the non-empty priority one buffer in the previous period.

The priority one buffer is almost empty in Case 5, i.e. $n[(j_1,i),t-1]=1$, and Machine M_i can operate as it is up and not blocked. Since the virtual upstream machine related to the priority one buffer might be down or fail, it is possible that the priority one buffer is empty at time $t+1$. If this happens, $M_d(j_2,i)$ is up at time $t+1$.

In Case 6, there is more than one part in the priority one buffer $B_{j_1,i}$ at time t and therefore $B_{j_1,i}$ cannot be empty at time $t+1$. For this reason, $M_d(j_2,i)$ cannot be up at time $t+1$.

The resumption of flow probability $r_d(j_2,i)$ describing repairs of Machine $M_d(j_2,i)$ is defined as

$$\begin{aligned} r_d(j_2,i) &= \text{prob}\Big[\{\alpha_d[(j_2,i),t+1]=1\} \mid \\ &\qquad \{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \end{aligned} \tag{97}$$

and is evaluated by decomposing the conditioning event. However, in the decomposition we omit Cases 2, 4, and 6 in Table 4 since in these cases a transition to Machine $M_d(j_2,i)$ up at time $t+1$ is not possible:

$$r_d(j_2,i) = A(j_2,i)X(j_2,i) + B(j_2,i)Y(j_2,i) + C(j_2,i)Z(j_2,i) \tag{98}$$

where we define

$$\begin{aligned} A(j_2,i) &= \text{prob}\ \Big[\{\alpha_d[(j_2,i),t+1]=1\}\Big| \\ &\qquad \{\alpha_i(t)=0\} \text{ and } \{n[(j_1,i),t-1]=0\} \text{ and} \\ &\qquad \{n[(j_2,i),t-1]>0\}\Big] \end{aligned} \tag{99}$$

$$\begin{aligned} X(j_2,i) &= \text{prob}\ \Big[\{\alpha_i(t)=0\} \text{ and } \{n[(j_1,i),t-1]=0\} \text{ and} \\ &\qquad \{n[(j_2,i),t-1]>0\}\Big| \\ &\qquad \{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \end{aligned} \tag{100}$$

$$B(j_2,i) = \text{prob}\Big[\{\alpha_d[(j_2,i),t+1]=1\}\Big| \{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(j_1,i),t-1]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \quad (101)$$

$$Y(j_2,i) = \text{prob}\Big[\{n[(i,q),t-1]=N(i,q)\} \text{ and } \{n[(j_1,i),t-1]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big| \{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \quad (102)$$

$$C(j_2,i) = \text{prob}\Big[\{\alpha_d[(j_2,i),t+1]=1\}\Big| \{\alpha_i(t)=1\} \text{ and } \{n[(i,q),t-1]<N(i,q)\} \text{ and } \{n[(j_1,i),t-1]=1\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \quad (103)$$

$$Z(j_2,i) = \text{prob}\Big[\{\alpha_i(t)=1\} \text{ and } \{n[(i,q),t-1]<N(i,q)\} \text{ and } \{n[(j_1,i),t-1]=1\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big| \{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \quad (104)$$

Probabilities $A(j_2,i)$ in (99) and $X(j_2,i)$ in (100) are related to Case 1 in Table 4. Assuming that repairs of Machines M_i and $M_u(j_1,i)$ are independent, we find that $A(j_2,i)$ is the repair probability of Machine M_i times the probability that Machine $M_u(j_1,i)$ is not repaired, i.e.

$$A(j_2,i) = r_i(1-r_u(j_1,i)) \quad (105)$$

since the reason for the buffer of the priority one Line $L(j_1,i)$ to be empty at time t is a failure of Machine $M_u(j_1,i)$. Note that this is a term of a new kind which reflects the priorities assigned to the two buffers upstream of a merge machine: One machine (M_i) has to be repaired while a another machine ($M_u(j_1,i)$) must not be repaired: If Machine $M_u(j_1,i)$ were repaired, the priority one buffer would become non-empty and Machine $M_d(j_2,i)$ would stay down, but now for a different reason.

A similar reasoning applies for the probability $B(j_2,i)$ in (101) which is related to Case 3 in Table 4. The virtual downstream machine $M_d(i,q)$ must be repaired and Machine $M_u(j_1,i)$ must not be repaired in order for the flow to resume out of the priority two buffer:

$$B(j_2,i) = r_d(i,q)(1-r_u(j_1,i)) \quad (106)$$

In (102), $Y(j_2,i)$ is the probability that the buffer of Line $L(i,q)$ is full and the buffer of Line $L(j_1,i)$ empty given that Machine $M_d(j_2,i)$ is down. Assuming independence for buffer levels in Lines $L(j_1,i)$ and $L(i,q)$, we find, after a derivation similar to that leading to (41),

$$Y(j_2,i) \approx \mathbf{p}[(j_1,i);001]\mathbf{p}[(i,q);N(i,q)10]\frac{r_d(j_2,i)}{p_d(j_2,i)E(j_2,i)}. \tag{107}$$

We next analyze the the probability $Z(j_2,i)$ in (104) which is related to Case 5 in Table 4. Since the event $\{n[(j_1,i),t-1]=1\}$ implies that Machine $M_d(j_2,i)$ is down at time t, we can use the definition of conditional probability and write

$$\begin{aligned}
Z(j_2,i) &= \text{prob}\Big[\{\alpha_i(t)=1\} \text{ and } \{n[(i,q),t-1]<N(i,q)\} \text{ and} \\
&\qquad \{n[(j_1,i),t-1]=1\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] : \\
&\quad \text{prob}\Big[\{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \\
Z(j_2,i) &= \text{prob}[\{ \text{ Case 5 } \}] : \\
&\quad \text{prob}\Big[\{\alpha_d[(j_2,i),t]=0\} \text{ and } \{n[(j_2,i),t-1]>0\}\Big] \\
&= \text{prob}[\{ \text{ Case 5 } \}]\frac{r_d(j_2,i)}{E(j_2,i)p_d(j_2,i)}.
\end{aligned} \tag{108}$$

It is not necessary to derive the probability of Case 5 explicitly since this quantity can be canceled out in the term $C(j_2,i)Z(j_2,i)$ of the resumption of flow equation (98) after the derivation of $C(j_2,i)$.

To derive an expression for the conditional probability $C(j_2,i)$ in (103), we list in Table 5 the implications of Case (Table 4 on Page 45), i.e. of having one part in the buffer of the priority one Line $L(j_1,i)$ while Machine M_i is up and not blocked. These implications are used to decompose the conditional probability $C(j_2,i)$ in (103).

The first entry in Table 5 is related to Line $L(i,q)$. In Case 5, Machine M_i is up, not starved and not blocked. This implies that the buffer level $n[(i,q),t-1]$ is in the interval $[0,N(i,q)-1]$ and that Machine $M_u(i,q)$ is up. However, nothing is implied with respect to Machine $M_d(i,q)$. It can be up or down.

The second entry in Table 5 describes the situation in Line $L(j_1, i)$. Case 5 states explicitly the buffer level $n[(j_1, i), t-1] = 1$. Since Machine M_i is up and not blocked, Machine $M_d(j_1, i)$ is also up. Nothing is implied with respect to Machine $M_u(j_1, i)$.

Line	Buffer Level	Machine State
$L(i,q)$	$0 \leq n[(i,q),t-1] \leq N(i,q)-1$	$\alpha_u[(i,q),t] = 1,$ $\alpha_d[(i,q),t] \in \{0,1\}$
$L(j_1,i)$	$n[(j_1,i),t-1] = 1$	$\alpha_u[(j_1,i),t] \in \{0,1\},$ $\alpha_d[(j_1,i),t] = 1$

Table 5: Implications of Case 5

To reduce the notational effort when decomposing the conditioning event of $C(j_2, i)$, the auxiliary events a_1, a_2, a_3 as well as b_1 and b_2 are defined in Table 6.

$$
\begin{aligned}
a_1 &= \{0 \leq n[(i,q),t-1] \leq N(i,q)-2\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] \in \{0,1\}\} \\
a_2 &= \{n[(i,q),t-1] = N(i,q)-1\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] = 1\} \\
a_3 &= \{n[(i,q),t-1] = N(i,q)-1\} \text{ and} \\
&\quad \{\alpha_u[(i,q),t] = 1\} \text{ and } \{\alpha_d[(i,q),t] = 0\}
\end{aligned}
$$

$$
\begin{aligned}
b_1 &= \{n[(j_1,i),t-1] = 1\} \text{ and} \\
&\quad \{\alpha_u[(j_1,i),t] = 1\} \text{ and } \{\alpha_d[(j_1,i),t] = 1\} \\
b_2 &= \{n[(j_1,i),t-1] = 1\} \text{ and} \\
&\quad \{\alpha_u[(j_1,i),t] = 0\} \text{ and } \{\alpha_d[(j_1,i),t] = 1\}
\end{aligned}
$$

$$
\begin{aligned}
c &= \{\alpha_d[(j_2,i),t+1] = 1\} \\
d &= \{n[(j_2,i),t-1] > 0\}
\end{aligned}
$$

Table 6: Definition of Auxiliary Events

The mutually exclusive and collectively exhaustive events a_1, a_2, and a_3 describe the possible states of Line $L(i,q)$ implied by Case 5. In a similar manner, the auxiliary events b_1 and b_2 describe those of Line $L(j_1,i)$.

Given the definition of the auxiliary variables, we find

$$\text{prob}[\{\text{ Case 5 }\}] = \text{ prob}[\{a_1 \text{ or } a_2 \text{ or } a_3\} \text{ and } \{b_1 \text{ or } b_2\}] \tag{109}$$

and therefore

$$\begin{aligned} C(j_2,i) &= \text{prob}\Big[\, c \,\Big|\{a_1 \text{ or } a_2 \text{ or } a_3\} \text{ and } \{b_1 \text{ or } b_2\} \text{ and } d\Big] \\ &= \text{prob}\,\Big[c \,\Big|\, a_1\, b_1\, d \;\text{ or }\; a_1\, b_2\, d \;\text{ or} \\ &\qquad\qquad a_2\, b_1\, d \;\text{ or }\; a_2\, b_2\, d \;\text{ or} \\ &\qquad\qquad a_3\, b_1\, d \;\text{ or }\; a_3\, b_2\, d \;\text{ or }\Big] \end{aligned}$$

Each of the events $a_k\, b_l\, d$ with $k \in \{1,2,3\}$, $l \in \{1,2\}$ implies Case 5 and this implies that the priority two buffer is not empty, i.e. event d. The last implication is due to the assumption of operation dependent failures. As an approximation, we assume that the events related to different lines are independent. In this case $C(j_2,i)$ can be decomposed to find

$$\begin{aligned} C(j_2,i) = \Big[&\text{prob}[\, c \mid a_1\, b_1\,]\, \text{prob}[\, a_1\,]\, \text{prob}[\, b_1\,] + \\ &\text{prob}[\, c \mid a_1\, b_2\,]\, \text{prob}[\, a_1\,]\, \text{prob}[\, b_2\,] + \\ &\text{prob}[\, c \mid a_2\, b_1\,]\, \text{prob}[\, a_2\,]\, \text{prob}[\, b_1\,] + \\ &\text{prob}[\, c \mid a_2\, b_2\,]\, \text{prob}[\, a_2\,]\, \text{prob}[\, b_2\,] + \\ &\text{prob}[\, c \mid a_3\, b_1\,]\, \text{prob}[\, a_3\,]\, \text{prob}[\, b_1\,] + \\ &\text{prob}[\, c \mid a_3\, b_2\,]\, \text{prob}[\, a_3\,]\, \text{prob}[\, b_2\,]\Big] \cdot \\ &\frac{1}{\text{prob}[\{\text{ Case 5 }\}]} \end{aligned} \tag{110}$$

The conditional probabilities in (110) can be determined using the information in Table 6. The first conditional probability, prob[$c \mid a_1\, b_1$], is the probability that Machine $M_d(j_2,i)$ is up at time $t+1$, given that the buffer level in Line $L(i,q)$ which receives the part at time t, is below $N(i,q)-1$ at time $t-1$ (event a_1), that the priority one buffer is almost empty, i.e.

$n[(j,i),t-1]=1$, and that the Machine $M_u(j_1,i)$ is up at time t (event b_1). The priority one buffer looses a part at time t which is processed by Machine M_i. If Machine $M_u(j_1,i)$ fails at time t, the buffer becomes empty which is necessary for Machine $M_d(j_2,i)$ to be up at time $t+1$.

Thus, the following has to happen in order for Machine $M_d(j_2,i)$ to be up at time $t+1$:

- Machine M_i in the real system must not fail, with probability $(1-p_i)$.
- The virtual upstream machine $M_u(j_1,i)$ related to the priority one buffer must fail, with probability $p_u(j_1,i)$.

Given the buffer level in Line $L(i,q)$, Machine M_i cannot be blocked at time $t+1$. Since failures of M_i and $M_u(j_2,i)$ are assumed to be independent, we find

$$\text{prob}[\ c\ |\ a_1\, b_1\ ] = (1-p_i)\, p_u(j_1,i). \tag{111}$$

In the second conditional probability in (110), prob[c | $a_1\, b_2$], event b_2 says that Buffer $B_{j,i}$ which looses a part at time t is almost empty, i.e. $n[(j,i),t-1]=1$, and that Machine $M_u(j_1,i)$ is down at time t. In this situation, the following has to happen in order for Machine $M_d(j_2,i)$ to be up at time $t+1$:

- Machine M_i in the real system must not fail, with probability $(1-p_i)$.
- Machine $M_u(j_1,i)$ must not be repaired (because otherwise Buffer $B_{j_2,i}$ would become non-empty), with probability $(1-r_u(j_1,i))$.

The second conditional probability is therefore

$$\text{prob}[\ c\ |\ a_1\, b_2\ ] = (1-p_i)\,(1-r_u(j_1,i)). \tag{112}$$

In the third conditional probability in (110), prob[c | $a_2\, b_1$], Buffer $B_{i,q}$ is almost full and Machine $M_d(i,q)$ is up (event a_2). Furthermore, buffer $B_{j_1,i}$ is almost empty and Machine $M_u(j_1,i)$ is also up.

The following has to happen in order for Machine $M_d(j_2,i)$ to be up at time $t+1$:

- Machine M_i in the real system must not fail, with probability $(1-p_i)$.

- The virtual downstream machine $M_d(i,q)$ must not fail to avoid blockage of M_i, with probability $(1 - p_d(i,q))$.
- The virtual upstream machine $M_u(j_1,i)$ must fail for the priority one buffer $B_{j_1,i}$ to become empty, with probability $p_u(j_1,i)$.

We find

$$\text{prob}[\ c\ \mid a_2\, b_1\ ] = (1-p_i)\,(1-p_d(i,q))\,p_u(j_1,i). \tag{113}$$

The derivation for the three remaining conditional probabilities is completely analogous and we find:

$$\begin{aligned}
\text{prob}[\ c\ \mid a_2\, b_2\ ] &= (1-p_i)\,(1-p_d(i,q))\,(1-r_u(j_1,i)) && (114)\\
\text{prob}[\ c\ \mid a_3\, b_1\ ] &= (1-p_i)\,r_d(i,q)\,p_u(j_1,i) && (115)\\
\text{prob}[\ c\ \mid a_3\, b_2\ ] &= (1-p_i)\,r_d(i,q)\,(1-r_u(j_1,i)) && (116)
\end{aligned}$$

The special structure of these six conditional probabilities allows to factor (110). This yields

$$\begin{aligned}
&C(j_2,i)\\
&= (1-p_i)\\
&\quad \Big[\ \text{prob}[\ a_1\ ] + (1-p_d(i,q))\ \text{prob}[\ a_2\ ] + r_d(i,q))\ \text{prob}[\ a_3\ ]\Big]\cdot\\
&\quad \Big[p_u(j_1,i)\ \text{prob}[\ b_1\ ] + (1-r_u(j_1,i))\ \text{prob}[\ b_2\ ]\Big]\cdot\\
&\quad \frac{1}{\text{prob}[\{\ \text{Case 5}\ \}]}.
\end{aligned} \tag{117}$$

Using the information in Table 6, the unconditional probabilities can be expressed as

$$\begin{aligned}
\text{prob}[\ a_1\ ] &= \sum_{n=0}^{N(i,q)-2} \Big(\mathbf{p}[(i,q);n11] + \mathbf{p}[(i,q);n10]\Big)\\
\text{prob}[\ a_2\ ] &= \mathbf{p}[(i,q);N(i,q)-1,11]\\
\text{prob}[\ a_3\ ] &= \mathbf{p}[(i,q);N(i,q)-1,10]\\
\text{prob}[\ b_1\ ] &= \mathbf{p}[(j_1,i);111]\\
\text{prob}[\ b_2\ ] &= \mathbf{p}[(j_1,i);101]
\end{aligned}$$

and we find

$$
\begin{aligned}
C(j_2,i) \;=\;& (1-p_i)\cdot \\
& \Bigg[\sum_{n=0}^{N(i,q)-2}\Big(\mathbf{p}[(i,q);n11]+\mathbf{p}[(i,q);n10]\Big) \\
& \quad +(1-p_d(i,q)\mathbf{p}[(i,q);N(i,q)-1,11] \\
& \quad +r_d(i,q)\mathbf{p}[(i,q);N(i,q)-1,10])\Bigg]\cdot \\
& \Big[p_u(j_1,i)\mathbf{p}[(j_1,i);011] \\
& \quad +(1-r_u(j_1,i))\mathbf{p}[(j_1,i);001]\Big]\cdot \\
& \frac{1}{\text{prob}[\{\text{ Case 5 }\}]}
\end{aligned}
\tag{118}
$$

Multiplying $C(j_2,i)$ and $Z(j_2,i)$ as required in (98) yields

$$
\begin{aligned}
C(j_2,i)Z(j_2,i) \;=\;& (1-p_i)\cdot \\
& \Bigg[\sum_{n=0}^{N(i,q)-2}\Big(\mathbf{p}[(i,q);n11]+\mathbf{p}[(i,q);n10]\Big) \\
& \quad +(1-p_d(i,q)\mathbf{p}[(i,q);N(i,q)-1,11] \\
& \quad +r_d(i,q)\mathbf{p}[(i,q);N(i,q)-1,10])\Bigg]\cdot \\
& \Big[p_u(j_1,i)\mathbf{p}[(j_1,i);011] \\
& \quad +(1-r_u(j_1,i))\mathbf{p}[(j_1,i);001]\Big]\cdot \\
& \frac{r_d(j_2,i)}{E(j_2,i)p_d(j_2,i)},
\end{aligned}
\tag{119}
$$

i.e. the probability of Case can be canceled out.

The last probability we have to determine is $X(j_2,i)$ in (100), i.e. the conditional probability that Machine M_i is down and Buffer $B_{j_1,i}$ is empty given that Machine $M_d(j_2,i)$ is down. In Table 4 on Page 45, the situation that Machine M_i is down and the buffer of the priority one line is empty is referred to as Case 1. Since Table 4 contains all the disjoint reasons for Machine $M_d(j_2,i)$ to be down, we can write

$$\begin{aligned}
&\text{prob}\Big[\{\text{ Case 1}\}\Big|\{\alpha_d[(j_2,i),t]=0\}\text{ and }\{n[(j_2,i),t-1]>0\}\Big] \\
&= 1-\text{prob}\Big[\{\text{ Case 2 or 3 or 4 or 5 or 6}\}\Big| \\
&\qquad\qquad \{\alpha_d[(j_2,i),t]=0\}\text{ and }\{n[(j_2,i),t-1]>0\}\Big] \qquad (120)\\
&= 1-\frac{\text{prob}\Big[\ \{\text{Case 2 or 5 or 6}\}\text{ or }\{\text{ Case 3 or 4}\}\Big]}{\text{prob}[\alpha_d[(j_2,i),t]=0\text{ and }n[(j_2,i),t-1]>0]} \qquad (121)\\
&= 1-\frac{\text{prob}\Big[\ \{\text{Case 2 or 5 or 6}\}\Big]+\text{prob}\Big[\{\text{ Case 3 or 4}\}\Big]}{\text{prob}[\alpha_d[(j_2,i),t]=0\text{ and }n[(j_2,i),t-1]>0]} \qquad (122)
\end{aligned}$$

since all cases imply that Machine $M_d(j_2,i)$ is down. We see from Table 4 that the probability of the {Case 2 or 5 or 6} is the probability of having a non-empty buffer in the priority one line $L(j_1,i)$ and a non-full buffer in Line $L(i,q)$. As an approximation, we assume again that these buffer levels are independent, or

$$\begin{aligned}
&\text{prob}\ \Big[\{\text{ Case 2 or 5 or 6 }\}\ \Big] \\
&\quad\approx\ (1-\mathbf{p}[(i,q);N(i,q)10])(1-\mathbf{p}[(j_1,i);001]) \qquad (123)
\end{aligned}$$

and we also find that

$$\text{prob}\ [\ \{\text{Case 3 or 4 }\}\ ]=\mathbf{p}[(i,q);N(i,q)10] \qquad (124)$$

since in Cases 3 and 4, Machine M_i must be up (as a blocked machine cannot fail). We hence conclude that

$$\begin{aligned}
X(j_2,i) &= \text{prob}\Big[\ \{\text{Case 1}\}\mid\alpha_d[(j_2,i),t]=0\text{ and }n[(j_2,i),t-1]>0\Big] \\
&= 1-\Big[(1-\mathbf{p}[(i,q);N(i,q)10])(1-\mathbf{p}[(j_1,i);001]) \\
&\qquad +\mathbf{p}[(i,q);N(i,q)10]\Big]\frac{r_d(j_2,i)}{p_d(j_2,i)E(j_2,i)]}. \qquad (125)
\end{aligned}$$

The resumption of flow equation can thus be expressed as

$$r_d(j_2,i) = r_iF+K_4\frac{r_d(j_2,i)}{p_d(j_2,i)} \qquad (126)$$

where we define

$$
\begin{aligned}
F &= (1 - r_u(j_1, i)) \qquad (127)\\
K_4 &= \Big[r_d(i,q) F \mathbf{p}[(i,q); N(i,q)10] \mathbf{p}[(j_1,i); 001] + \\
&\quad (1 - p_i) \cdot \\
&\quad \Big[\sum_{n=0}^{N(i,q)-2} \mathbf{p}[(i,q); n10] + \mathbf{p}[(i,q); n11] \\
&\quad + (1 - p_d(i,q) \mathbf{p}[(i,q); N(i,q) - 1, 11] \\
&\quad + r_d(i,q) \mathbf{p}[(i,q); N(i,q) - 1, 10] \Big] \\
&\quad \Big[p_u(j_1,i) \mathbf{p}[(j_1,i); 111] + (1 - r_u(j_1,i)) \mathbf{p}[(j_1,i); 101] \Big] \\
&\quad - r_i F \Big[(1 - \mathbf{p}[(i,q); N(i,q)10])(1 - \mathbf{p}[(j_1,i); 001]) \\
&\quad + \mathbf{p}[(i,q); N(i,q)10] \Big] \Big] \frac{1}{E(j_2,i)]} \qquad (128)
\end{aligned}
$$

Note that the term K_4 in (128) does not contain parameters of Line $L(j_2, i)$. Now all resumption of flow equations for split and merge systems have been determined.

2.7 Boundary Equations

In the definition of the model, we assume that machines without preceding machines do not perform split operations. Thus, such an input machine M_i with $U(i) = \{\}$ has exactly one downstream buffer denoted as $B_{i,l}$. The observer in this buffer sees the original behavior of this input machine, so

$$
\begin{aligned}
p_u(i,l) &= p_i, \qquad \forall i \text{ with } U(i) = \{\} \qquad (129)\\
r_u(i,l) &= r_i, \qquad \forall i \text{ with } U(i) = \{\} \qquad (130)
\end{aligned}
$$

i.e. the parameters of Machine $M_u(i,l)$ are those of M_i if Machine M_i is an input machine.

We further assume that machines without succeeding machines do not perform merge operations. Thus, such an output machine M_l with $D(l) = \{\}$

has exactly one upstream buffer denoted as $B_{i,l}$. The observer in this buffer sees the original behavior of the output machine, so

$$p_d(i,l) = p_l, \qquad \forall l \text{ with } D(l) = \{\} \tag{131}$$

$$r_d(i,l) = p_l, \qquad \forall l \text{ with } D(l) = \{\} \tag{132}$$

i.e. the parameters of Machine $M_d(i,l)$ are those of M_l if Machine M_i is an output machine. Thus, the parameters of virtual machines which correspond to input or output machines in the real system are determined by the boundary conditions. The decomposition equations are required to determine the parameters for the other virtual machines not corresponding to input or output machines.

If a system has B buffers, there are B two-machine lines in the decomposition and $4B$ failure and repair probabilities of virtual machines have to be determined. This requires $4B$ equations which are given by the boundary equations in this subsection, the flow rate-idle time equations (14), (15), and the resumption of flow equations (59), (73), (90), (93), and (126).

2.8 Reformulation of Decomposition Equations

The decomposition equations (14), (15),

$$\frac{r_u(i,m)+p_u(i,m)}{r_u(i,m)} = \frac{1}{e_u(i,m)} = K_1$$
$$\frac{r_d(j,i)+p_d(j,i)}{r_d(j,i)} = \frac{1}{e_d(j,i)} = K_2$$

as well as (59), (73), (90), (93), and (126)

$$\begin{aligned} r_u(i,m) &= \left[r_i + K_3\frac{r_u(i,m)}{p_u(i,m)}\right] d_{i,m} \\ r_d(j,i) &= r_i F + K_4\frac{r_d(j,i)}{p_d(j,i)} \end{aligned}$$

can be solved for the parameters of Lines $L(i,m)$ and $L(j,i)$, respectively, to find:

$$r_u(i,m) = d_{i,m}\frac{K_3 + r_i(K_1-1)}{K_1-1} \quad (133)$$
$$p_u(i,m) = d_{i,m}\left(K_3 + r_i(K_1-1)\right) \quad (134)$$
$$r_d(j,i) = \frac{K_4 + r_i F(K_2-1)}{K_2-1} \quad (135)$$
$$p_d(j,i) = K_4 + r_i F(K_2-1) \quad (136)$$

This type of reformulation has been proposed in [Burman, 1995], in a Ph.D. thesis on the analysis of flow lines.

3 The Algorithm

3.1 Purpose, Background, and Basic Structure of the Algorithm

The purpose of the algorithm is to determine production rates and inventory levels for the model of a transfer line with split and merge operations. Since the real system is decomposed into a set of virtual two-machine lines, these quantities are approximated from the analysis of the virtual two-machine lines. The failure and repair probabilities for the virtual machines in these two-machine lines must satisfy the decomposition equations derived in the previous section. The basic idea of the algorithm is to solve the decomposition equations in an iterative way and to hope that the algorithm converges to a set of parameters for the two-machine lines that meets all the decomposition equations to some degree of accuracy. The performance measures of the two-machine lines are an approximation of those seen by an observer in the respective part of the real system.

The algorithm is based on the DDX-algorithm ([Dallery et al., 1988]) for linear transfer lines. It uses a reformulation of the decomposition similar to those proposed in [Burman, 1995] and evaluates the two-machine lines in a sequence similar to those proposed in [Gershwin, 1991] for assembly/disassembly systems.

The basic structure of the algorithm is as follows:

1. Determine an evaluation sequence for the pseudo-machines in the two-machine lines (Section 3.2).

2. Initialize the parameters of the two-machine models for each line (Section 3.3).

3. For each virtual upstream machine $M_u(i,q)$ (Section 3.4.1):

   (a) Update failure and repair probabilities $p_u(i,q)$ and $r_u(i,q)$.

   (b) Compute new steady-state probabilities and performance measures for the two-machine line of the virtual upstream machine.

4. For each virtual downstream machine $M_d(j,i)$ (Section 3.4.2):

   (a) Update failure and repair probabilities $p_d(j,i)$ and $r_d(j,i)$.

(b) Compute new steady-state probabilities and performance measures for the two-machine line of the virtual downstream machine.

5. Go to Step 3 if the production rates of the two-machine lines do not satisfy the conservation of flow equation and an upper limit on the number of iterations is not exceeded.

6. Stop.

In the remainder of this section, the details of the algorithm are discussed.

3.2 Determination of the Evaluation Sequence

The evaluation sequence consists of two parts. The first part denoted as S_U describes the order in which virtual upstream machines in Phase 3 of the algorithm are updated. The second part denoted as S_D describes the order for virtual downstream machines in Phase 4.

Consider the structure in Figure 8 where an index $I_{B_{i,q}}$ is assigned to each buffer $B_{i,q}$. The indices are depicted above the respective buffers. In systems with loops, there does not appear to be an obvious way how to assign these indices. As a general rule, we tried to follow the flow of material, i.e. to assign lower indices to buffers at earlier production stages.

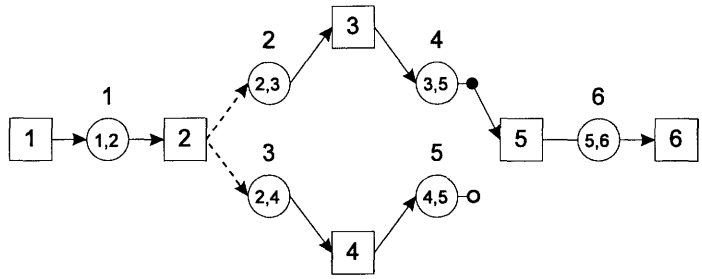

Figure 8: Structure with Indices Assigned to Buffers

We first construct S_U by starting with the pseudo-machines which are *immediately upstream* of priority two buffers. Priority two buffers are considered according to ascending indices. In the next step, the remaining pseudo-machine upstream of priority one or single buffers are added, again according to ascending indices.

In the structure in Figure 8, there is one priority two buffer: $B_{4,5}$. Machine $M_u(4,5)$ is therefore placed at the head of Sequence S_U. All other virtual upstream machines are related to priority one or single input buffers. They are appended to the sequence according to ascending indices, starting with Machine $M_u(1,2)$ related Buffer $B_{1,2}$ which has the lowest index $I_{B_{1,2}} = 1$. Using this simple rule for the given assignment of buffer indices, we find the following upstream sequence:

$$S_U = \{M_u(4,5), M_u(1,2), M_u(2,3), M_u(2,4), M_u(3,5), M_u(5,6)\}$$

That is, we first determine new parameters for Machine $M_u(4,5)$, then for $M_u(1,2)$, and so on. (Note that the parameters for Machine $M_u(1,2)$ are actually determined by the boundary conditions as M_1 is an input machine.)

To construct the downstream sequence which is processed in the reversed order, we replace upstream pseudo-machines by downstream pseudo-machines:

$$S_D = \{M_d(4,5), M_d(1,2), M_d(2,3), M_d(2,4), M_d(3,5), M_d(5,6)\}$$

We hence start with Machine $M_d(5,6)$ and go *backwards* through the sequence until we finally update the parameters for Machine $M_d(4,5)$.

Due to this sequence, the parameters of downstream machines related to priority two buffers (in this case, $M_d(4,5)$) are always updated *after* the parameters of the downstream machine of the corresponding priority one buffer (in the example $M_d(3,5)$). Numerical test which are not reported her indicate that this helps to achieve a reasonable convergence reliability especially for very small merge systems.

3.3 Initialization

The failure and repair parameters of the two-machine lines are initialized with the respective values of the corresponding machines in the real system.

For each line $L(i,m)$, set

$$p_u(i,m) = p_i, \tag{137}$$
$$r_u(i,m) = r_i, \tag{138}$$

$$p_d(i,m) = p_m, \tag{139}$$
$$r_d(i,m) = r_m. \tag{140}$$

The two-machine model by Gershwin and Schick used in the decomposition includes the workspace at the two machines in the extended buffer space. For this reason, set

$$N(i,m) = C_{i,m} + 2 \tag{141}$$

where $C_{i,m}$ is the number of physical buffer spaces between Machines M_i and M_m and $N(i,m)$ is the extended buffer size. Solve all the two-machine lines to determine initial steady state probabilities $\mathbf{p}[(i,m)n\alpha_u\alpha_d]$ and production rates $E(i,m)$ (see also [Gershwin, 1994, p. 76-93]).

3.4 Iterative Solution of the Decomposition Equations

The iterative procedure is used to update the parameters of virtual machines that do not correspond to input or output machines. The parameters of those virtual machines that correspond to input or output machines are given by the boundary equations (129)-(132) and stay at their initial values.

3.4.1 Upstream Phase

Consider sequentially all machines $M_u(i,m)$ in Sequence S_U except for *input machines*, i.e. machines without preceding machines. For each machine $M_u(i,m)$, determine $E(i)$ from (142) and $E(i,m)$ from (143):

$$E(i) = \sum_{(j,i)\in U(i)} E(j,i) \tag{142}$$
$$E(i,m) = d_{i,m}E(i) \tag{143}$$

Equation (143) reflects the random routing of the parts leaving Machine M_i.

Determine K_1 from (16):

$$K_1 = \frac{\frac{E(i)}{e_i} + \prod_{(l,i)\in U(i)}(1 - \frac{E(l,i)}{e_d(l,i)}) + \sum_{(i,q)\in D(i),q\neq m}(1 - \frac{E(i,q)}{e_u(i,q)})}{E(i,m)}$$

If Machine M_i performs split operations, determine K_3 from (60)

$$\begin{aligned} K_3 &= \Big[(r_u(j,i) - r_i)\mathbf{p}[(j,i);001] \\ &+ \sum_{(i,q)\in D(i), q\neq m} (r_d(i,q) - r_i)\mathbf{p}[(i,q);N(i,q)10] \\ &+ \sum_{(i,q)\in D(i), q\neq m} (1-p_i)F(i,q) - r_i \text{ prob}[\{\beta_i(t)=q\}]\Big]\frac{1}{E(i,m)} \end{aligned}$$

where $F(i,q)$ is given in (58) and prob$[\{\beta_i(t)=q\}]$ in (48).

If Machine M_i performs merge operations, determine K_3 from (92):

$$K_3 = \Big(1 - \prod_{j\in\{j_1,j_2\}} (1 - r_u(j,i)) - r_i\Big) \prod_{j\in\{j_1,j_2\}} \mathbf{p}[(j,i);001]\frac{1}{E(i,q)}$$

If Machine M_i performs neither split nor merge operations, either equation for K_3 can be used. Determine preliminary values for upstream parameters $r_u^*(i,m)$ from (133) and $p_u^*(i,m)$ from (134):

$$\begin{aligned} r_u^*(i,m) &= d_{i,m}\frac{K_3 + r_i(K_1 - 1)}{K_1 - 1} \\ p_u^*(i,m) &= d_{i,m}\left(K_3 + r_i(K_1 - 1)\right) \end{aligned}$$

Especially during early iterations, we sometimes observed estimates $p_u^*(i,m)$ and $r_u^*(i,m)$ which are either negative or larger than 1. Since these are meaningless values for probabilities, we imposed a set of hard constraints on the parameter updates to be described below. We encountered a few cases where the algorithm appeared to oscillate. To avoid this behavior whenever the algorithm did not converge within 100 iterations, we exponentially smoothed all parameters updates in iteration k in the following way

$$p_u^{**}(i,m) = \epsilon\, p_u^*(i,m) + (1-\epsilon)\, p_u^{k-1}(i,m) \quad (144)$$

$$r_u^{**}(i,m) = \epsilon\, r_u^*(i,m) + (1-\epsilon)\, r_u^{k-1}(i,m) \quad (145)$$

where $\mathrm{p}_u^{k-1}(i,m)$ and $r_u^{k-1}(i,m)$ denote the parameters from the previous iteration.

During early iterations, even smoothed parameters guesses $p_u^{**}(i,m)$ were occasionally larger than 1 or smaller than p_i, the failure probability of Machine M_i. Since the virtual machine $M_u(i,m)$ cannot possibly fail less often than Machine M_i in the real system, $p_u(i,m)$ cannot be smaller than p_i.

We also occasionally observed smoothed parameters guesses $r_u^{**}(i,m)$ larger than 1 or smaller than 0. In order to update parameters within the possible range of values only, we used the following updating scheme:

$$p_u^k(i,m) = \begin{cases} p_u^{**}(i,m) & \text{if } p_i \le p_u^{**}(i,m) < 1 \\ p_i + 0.5\,(p_u^{k-1}(i,m) - p_i) & \text{if } p_u^{**}(i,m) < p_i \\ p_u^{k-1}(i,m) + 0.5\,(1 - p_u^{k-1}(i,m)) & \text{if } p_u^{**}(i,m) \ge 1 \end{cases} \quad (146)$$

$$r_u^k(i,m) = \begin{cases} r_u^{**}(i,m) & \text{if } 0 < r_u^{**}(i,m) < 1 \\ 0.5\,r_u^{k-1}(i,m) & \text{if } r_u^{**}(i,m) \le 0 \\ r_u^{k-1}(i,m) + 0.5\,(1 - r_u^{k-1}(i,m)) & \text{if } r_u^{**}(i,m) \ge 1 \end{cases} \quad (147)$$

If $p_u^{**}(i,m)$ is smaller than 1 and larger than p_i, we use $p_u^{**}(i,m)$ in (146) to update $p_u^k(i,m)$. However, if $p_u^{**}(i,m)$ is below p_i or above 1, we reduce or increase the previous parameters value within these bounds. The procedure for the repair parameters (147) is similar.

When the virtual failure and repair probabilities $p_u^k(i,m)$ and $r_u^k(i,m)$ for Machine $M_u(i,m)$ in Line $L(i,m)$ have been determined, the new steady-state probabilities $\mathbf{p}[(i,m)n\alpha_u\alpha_d]$ and the production rate $E(i,m)$ are computed using the approach by Gerhwin and Schick in [Gershwin, 1994, p. 76-93].

3.4.2 Downstream Phase

Consider all machines $M_d(j,i)$ in the downstream part S_D of the evaluation sequence except for *output machines*, i.e. machines without succeeding machines. For each machine $M_d(j,i)$, determine $E(i)$ from (148).

$$E(i) = \text{Min}\left\{ e_i, \sum_{(i,q)\in D(i)} E(i,q) \right\} \quad (148)$$

During early iterations of the algorithm, the sum of the production rates $E(i,q)$ may be higher than the isolated efficiency e_i of Machine M_i or even higher than 1, the isolated production rate of a perfectly reliable machine. However, the flow out of Machine M_i in the real system, $\sum_{(i,q)\in D(i)} E_{i,q}$,

cannot possibly be higher than the isolated production rate e_i of Machine M_i. Furthermore, a production rate higher than 1 is meaningless in the context of the underlying model. We therefore impose in (148) an upper limit e_i on the approximated throughput $E(i)$ of Machine M_i. To determine $E(j,i)$, we have to consider three different cases:

- Case A: Machine M_i does not perform merge operations.
- Case B: Machine M_i does perform merge operations and we are considering the production rate $E(j_1,i)$ related to the priority one buffer.
- Case C: Machine M_i does perform merge operations and we are considering the production rate $E(j_2,i)$ related to the priority two buffer.

In Case A, we set $E(j,i) = E(i)$, as in proposed [Dallery et al., 1988] (see also [Gershwin, 1994, p. 152]) for the simple transfer line. In Cases B and C, we have to enforce that $E(j_1,i)$ and $E(j_2,i)$ for the priority one and priority two buffers add up to $E(i)$ as determined in (148). We found that the following procedure leads to a reasonable convergence behavior: During the first five iterations, update

$$E(j_1,i)^k = \frac{E(j_1,i)^{k-1}}{E(j_1,i)^{k-1} + E(j_2,i)^{k-1}} E(i)^k \quad (149)$$

$$E(j_2,i)^k = \frac{E(j_2,i)^{k-1}}{E(j_1,i)^{k} + E(j_2,i)^{k-1}} E(i)^k \quad (150)$$

where superscripts k and $k-1$ denote the respective iteration. During the first few iterations, the sum of $E(j_1,i)^{k-1}$ and $E(j_2,i)^{k-1}$ may be larger than $E(i)$, or even larger than 1. In order to propagate positive new values for $E(j_1,i)^k$ and $E(j_2,i)^k$ which add up to $E(i)^k$, we use the updating schemes in (149) and (150).

After five iterations, we usually find $E(j_1,i)^{k-1} < E(i)^k$ and $E(j_2,i)^{k-1} < E(i)^k$. We therefore switch to the following updating scheme:

$$E(j_1,i)^k = E(i)^k - E(j_2,i)^{k-1} \quad (151)$$

$$E(j_2,i)^k = E(i)^k - E(j_1,i)^k \quad (152)$$

Using $E(i)$ and $E(j,i)$, determine K_2 from (17):

$$K_2 = [\frac{\frac{E(i)}{e_i} + \sum_{(i,q)\in D(i)}(1 - \frac{E(i,q)}{e_u(i,q)}) - 1}{\prod_{(l,i)\in U(i), l\neq j}(1 - \frac{E(l,i)}{e_d(l,i)})} + 1]\frac{1}{E(j,i)}$$

In Cases A and B, set $F = 1$. If Machine M_i does in Case A not perform a merge operation, it either performs a split operation or it sends material to exactly one succeeding machine. In both cases, determine K_4 from (75):

$$K_4 = \sum_{(i,q)\in D(i)} (r_d(i,q) - r_i)\mathbf{p}[(i,q); N(i,q)10]\frac{1}{E(j,i)}$$

If in Case B, Machine M_i performs a merge operation and we are considering the priority one buffer, determine K_4 from (95):

$$K_4 = \big(r_d(i,q) - r_i\big)\mathbf{p}[(i,q); N(i,q), 10]\frac{1}{E(j_1,i)}$$

In Case C, set $F = (1 - r_u(j_1,i))$ and determine K_4 from (128):

$$\begin{aligned} K_4 = & \Big[r_d(i,q)F\mathbf{p}[(i,q); N(i,q)10]\mathbf{p}[(j_1,i); 001] + \\ & (1-p_i)\cdot \\ & \Big[\sum_{n=0}^{N(i,q)-2} \mathbf{p}[(i,q); n10] + \mathbf{p}[(i,q); n11] \\ & +(1 - p_d(i,q)\mathbf{p}[(i,q); N(i,q) - 1, 11] \\ & +r_d(i,q)\mathbf{p}[(i,q); N(i,q) - 1, 10]\Big] \\ & \Big[p_u(j_1,i)\mathbf{p}[(j_1,i); 111] + (1 - r_u(j_1,i))\mathbf{p}[(j_1,i); 101]\Big] \\ & -r_iF\Big[(1 - \mathbf{p}[(i,q); N(i,q)10])(1 - \mathbf{p}[(j_1,i); 001]) \\ & +\mathbf{p}[(i,q); N(i,q)10]\Big]\Big]\frac{1}{E(j_2,i)]} \end{aligned}$$

Compute preliminary values for downstream parameters $r_d^*(j,i)$ from (135) and $p_d^*(j,i)$ from (136):

$$
\begin{aligned}
r_d^*(j,i) &= \frac{K_4 + r_i F(K_2 - 1)}{K_2 - 1} \\
p_d^*(j,i) &= K_4 + r_i F(K_2 - 1)
\end{aligned}
$$

As for the upstream machines, compute smoothed updates $r_d^{**}(j,i)$ from (154) and $p_d^{**}(j,i)$ from (153).

$$
p_d^{**}(j,i) = \epsilon\, p_d^*(j,i) + (1-\epsilon)\, p_d^{k-1}(j,i) \tag{153}
$$

$$
r_d^{**}(j,i) = \epsilon\, r_d^*(j,i) + (1-\epsilon)\, r_d^{k-1}(j,i) \tag{154}
$$

Finally, impose the hard constraints in (155) and (156) to determine the updated failure and repair probabilities for Machine $M_d(j,i)$ in iteration k:

$$
p_d^k(j,i) = \begin{cases} p_d^{**}(j,i) & \text{if } p_i \le p_d^{**}(j,i) < 1 \\ p_i + 0.5\,(p_d^{k-1}(j,i) - p_i) & \text{if } p_d^{**}(j,i) < p_i \\ p_d^{k-1}(j,i) + 0.5\,(1 - p_d^{k-1}(j,i)) & \text{if } p_d^{**}(j,i) \ge 1 \end{cases} \tag{155}
$$

$$
r_d^k(j,i) = \begin{cases} r_d^{**}(j,i) & \text{if } 0 < r_d^{**}(j,i) < 1 \\ 0.5\, r_d^{k-1}(j,i) & \text{if } r_d^{**}(j,i) \le 0 \\ r_d^{k-1}(j,i) + 0.5\,(1 - r_d^{k-1}(j,i)) & \text{if } r_d^{**}(j,i) \ge 1 \end{cases} \tag{156}
$$

When the virtual failure and repair probabilities $p_d^k(j,i)$ and $r_d^k(j,i)$ for Machine $M_d(j,i)$ in Line $L(j,i)$ have been computed, the new steady-state probabilities $\mathbf{p}[(j,i)n\alpha_u\alpha_d]$ and the production rate $E(j,i)$ are determined.

Termination. Stop the procedure if the conservation of flow (2) equations for all machines are met to a sufficient degree of accuracy for a number of successive iterations.

A proof of convergence for this type of algorithm is not available, but numerical experiments to be reported below show a high convergence reliability for a wide range of system parameters.

3.5 General Comments on Implementation and Algorithm Behavior

We always start the algorithm with a smoothing parameter of $\epsilon = 1.0$, i.e. without exponential smoothing. In the very few cases where the algorithm

does not converge within 100 iterations, we reduce ϵ to 0.5 for an additional 100 iterations and finally to 0.25 for a last 100 iterations before we abort the evaluation. In almost all cases, reducing ϵ to 0.5 or 0.25 is not necessary. Furthermore, we encountered cases where the estimated performance measures appeared to be a function of the smoothing parameter. For this reason, we suggest not to use the exponential smoothing unless it is necessary to achieve convergence.

In our numerical study to be described below, we terminated the algorithm when the conservation of flow equation (2) was met within 0.01 % for ten successive iterations. In most cases, these ten iterations lead to an even higher accuracy than 0.01 %. Our numerical results show a few cases where the algorithm failed to converge.

4 Numerical Results

4.1 Introduction

4.1.1 Overview

In order to evaluate both the behavior of manufacturing systems with split and merge operations and the performance of our decomposition method, we performed a numerical study based on artificial problems of different size and structure. In this study, we compared the results of our decomposition approach to results from a discrete-event simulation. The simulation was programmed in C. For each structure, i.e. each arrangement of machines and buffers, we studied a set of 100 randomly generated problems. The purpose of this first part of the study was to evaluate the algorithm with respect to convergence reliability and accuracy over a wide range of machine and buffer parameters. For each of the random problems, we ran a simulation of 20 independent runs over 31,000 time units where the first 1,000 time units of each run were omitted as a warm-up period.

In the second part of the study, we varied in a systematic way

- the probabilities of failures and repairs,
- the size of the buffers, and
- the routing probabilities

in order to discover systematic effects. Here, the focus was primarily on the manufacturing system behavior. We started with smaller pure split or merge systems and later considered larger structures with loops where both split and merge operations were performed. Due to the systematic variation especially of buffer sizes, some of the systems were rather similar and therefore hard to compare by simulation results. In order to get production rate estimates with tight 95 % confidence intervals, we increased the simulation time by running 20 independent runs over 110,000 time units. For each run, we omitted the first 10,000 time units as a warm-up period.

4.1.2 Generating Random Problems

To generate the random problems, we used parts of a procedure proposed in [Burman, 1995, p. 92-94] in the context of flow line analysis. In what follows,

let RAN denote a pseudo random number generated by a call to the rnd() function in Visual Basic. This random number generator returns a number uniformly distributed between 0 and 1. Each reference to RAN_h in the formulas below represents a different value of this variable which is obtained by a separate call h of rnd(). Following Burman's approach [Burman, 1995, p. 93], we first generated a single random number

$$x = 1 + (9RAN_1) \tag{157}$$

for a given structure by a single call to the random number generator. In the next step, a set of different random numbers

$$y_i = -(1 + RAN_{2,i}), \tag{158}$$

one for each machine M_i, was generated. We then computed repair probabilities

$$r_i = x^{y_i} \tag{159}$$

for each machine M_i. Due to this approach, the repair probabilities are between 0.01 and 1. They are similar in their order of magnitude for all machines in a structure. Given these repair probabilities, an average repair takes between 1 and 100 periods. This covers a wide range of possible repair times.

The failure probabilities p_i are generated in a way that results in isolated efficiencies between 50 % and 99 %:

$$p_i = r_i * 10^{-(0.66RAN_{3,i}+0.66RAN_{4,i}+0.66RAN_{5,i})} \tag{160}$$

The isolated efficiency cannot be higher than 100 %. An isolated efficiency below 50 % says that a machine is down more often than it is up. We do not consider this to be a very realistic assumption.

In general, this results in systems that are roughly balanced with respect to the isolated efficiencies $e_i = \frac{r_i}{r_i+p_i}$ of the machines. This is desirable to avoided random cases with slow bottleneck machines that are usually easy to analyze but not very realistic. However, due to split and merge operations, the bottleneck issue gets more complicated than it was in Burman's study of purely linear flow lines. In two of the cases to be described below, additional adjustments were necessary to generate random cases that were not extremely unbalanced.

The physical buffer sizes $C_{i,j}$ in a well-designed system are more or less proportional to the number of parts produced during the average repair time of the machines immediately up- or downstream of the buffer. If the buffer sizes are much smaller, an average failure propagates quickly through the system as machines are blocked and starved before the broken machine is repaired.

The numerical solution technique used for the two-machine model in the decomposition is based on the assumption that there are at least two physical buffer spaces $C_{i,j}$ *between* any two machines.

To this minimal buffer size $C_{i,j} = 2$ we randomly added up to three times the amount of space required for the parts produced during a failure of average duration at the adjacent machines:

$$C_{i,j} = 2 + MAX[\frac{1}{r_i}, \frac{1}{r_j}]3RAN_{6,j,i} \tag{161}$$

Since the two-machine model by Gershwin and Schick includes the workspaces at the machines in the calculation of the available storage, we have an extended buffer size $N(i,j) = C_{i,j} + 2$ of at least 4 in the two-machine lines used in the decomposition.

4.2 Pure Split Structures

4.2.1 Structure S1

We first studied Structure S1 depicted in Figure 9 for 100 random cases using the approach described above. The numbers above the buffers are the buffer indices used to determine the evaluation sequence. The routing probability $d_{2,3}$ was set to 0.9. We asked for the production rate in the branch between Machines M_2 and M_3.

The 100 random cases were evaluated both by simulation and by our approximation technique and sorted according to ascending simulated production rates. Figure 10 shows the simulated production rates for the 100 random cases. We do not display the approximated production rate estimates in Figure 10 as they are very close to the simulation results. Instead, we display in Figure 11 the percentage error $(PR_{Ap} - PR_{Si})/PR_{Si}$ for each case. The algorithm converged in all 100 cases and the average over the absolute values of the relative errors was 0.81 %. The probably somewhat less important buffer levels estimates are typically less accurate as we discuss

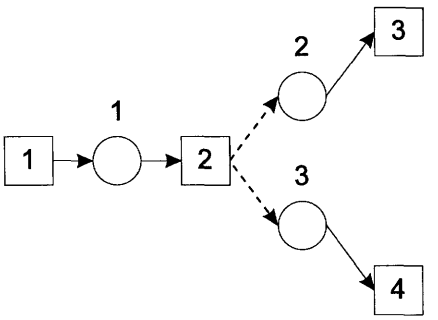

Figure 9: Structure S1

Class S1C1: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, r_i = 0.1, \forall i$						
Case	$d_{2,3}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C1S1	0.95	13	6.809	.7375 ± .0024	.7532	2.1
S1C1S2	0.9	13	7.452	.7014 ± .0027	.7165	2.2
S1C1S3	0.7	13	8.172	.5505 ± .0016	.5620	2.1
S1C1S4	0.5	13	8.271	.3951 ± .0012	.4023	1.8
S1C1S5	0.3	12	8.163	.2357 ± .0011	.2409	2.2
S1C1S6	0.1	12	7.553	.0779 ± .0006	.0796	2.2
S1C1S7	0.05	12	6.838	.0388 ± .0003	.0396	2.1

Table 7: Results for Class S1C1 (Small Buffers)

in more detail below. Usually, the algorithm terminated after less then 20 iterations.

In the second part of the study for Structure S1, we analyzed 44 different sets of parameters. These 44 problems were arranged in 8 different problem classes. The performance measure was again the production rate in the branch between Machines M_2 and M_3 in Figure 9. In the first problem class, referred to as S1C1, we used the parameters in the upper part of Table 7. The results for different values of the routing probability $d_{2,3}$ are given in the lower part of Table 7.

In this problem class, the routing probability $d_{2,3}$ changed from 0.95 to 0.05. The last row in Table 7 reports the relative difference between the approximated and the simulated production rate for the branch between Machines M_2 and M_3. (It is not necessary to report results for more than one branch of the network since the *ratio* of the respective production rates in different branches is determined by the routing probabilities $d_{i,q}$ in the

Figure 10: Structure S1 - Simulated Production Rates for Random Problems

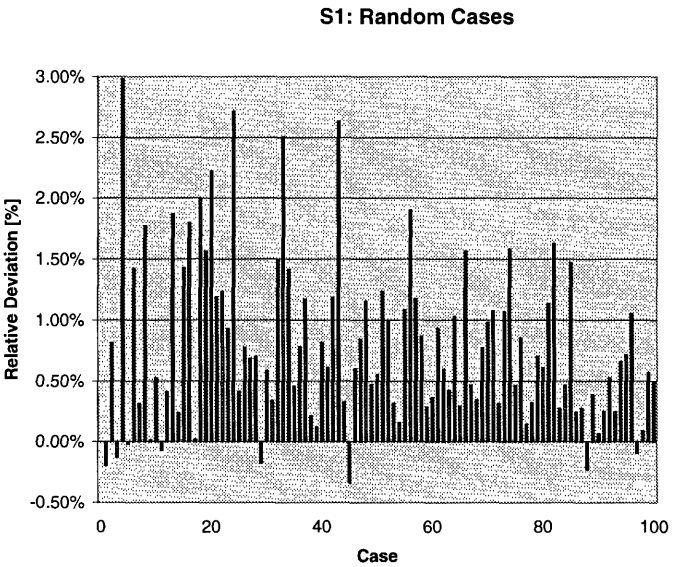

Figure 11: Structure S1 - Percentage Errors for Random Problems

Class S1C2: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, r_i = 0.1, \forall i$						
Case	$d_{2,3}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C2S1	0.95	13	2.926	.7790 ± .0022	.7878	1.1
S1C2S2	0.9	13	2.944	.7441 ± .0024	.7502	0.8
S1C2S3	0.7	13	2.860	.5851 ± .0013	.5903	0.9
S1C2S4	0.5	13	2.939	.4201 ± .0012	.4232	0.7
S1C2S5	0.3	12	2.874	.2499 ± .0010	.2530	1.2
S1C2S6	0.1	12	2.884	.0820 ± .0004	.0834	1.6
S1C2S7	0.05	12	2.912	.0410 ± .0004	.0415	1.1

Table 8: Results for Class S1C2 (Larger Buffers)

underlying model. For this reason, the accuracy of the production rate approximation is identical for all branches of pure split structures.) We stopped the iterative approximation when the conservation of flow condition (2) was met to a relative accuracy of 0.0001 for each machine in the system for 10 consecutive iterations. That is, the approximation needed three iterations for Case S1C1S1 in Table 7 to meet the conservation of flow equation at an accuracy of 0.01 % and then stayed within this level of accuracy for the next 10 iterations. (In most of the cases this led to an even higher accuracy with respect to the conservation of flow equation.)

The fourth row is a measure of the accuracy of the buffer level estimate. We determined for each buffer b the difference between the average buffer level from the decomposition approach $\overline{n}_{App}(b)$ and the the average buffer level from the simulation $\overline{n}_{Sim}(b)$. The absolute value of this difference is related to the extended buffer size $N(b)$. The average over these relative quantities is computed as follows

$$BL = \frac{1}{B} \sum_{b=1}^{B} \frac{|\overline{n}_{App}(b) - \overline{n}_{Sim}(b)|}{N(b)} \tag{162}$$

where B is the number of buffers in the system.

The fifth row gives the average simulated production rate and the half-width of the respective 95 % confidence interval.

The results in Table 7 show that the buffer level estimate is less accurate than the production rate estimate, that the production rate estimate itself is very accurate and that the algorithm converges quickly for this problem class.

Class S1C3: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 1,2,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.9$						
Case	p_3	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C3S1	0.01	13	7.493	.7027 ± .0016	.7165	2.0
S1C3S2	0.04	13	6.285	.5874 ± .0023	.6100	3.8
S1C3S3	0.07	12	10.011	.5038 ± .0016	.5251	4.2
S1C3S4	0.1	13	12.140	.4415 ± .0015	.4586	3.9
S1C3S5	0.2	13	15.018	.3134 ± .0018	.3191	1.8

Table 9: Results for Class S1C3 (Small Buffers)

Class S1C4: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 1,2,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.9$						
Case	p_3	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C4S1	0.01	13	2.916	.7404 ± .0019	.7502	1.3
S1C4S2	0.04	14	2.359	.6473 ± .0027	.6613	2.2
S1C4S3	0.07	13	4.536	.5534 ± .0014	.5679	2.6
S1C4S4	0.1	13	5.601	.4817 ± .0017	.4907	1.9
S1C4S5	0.2	14	6.364	.3293 ± .0019	.3318	0.7

Table 10: Results for Class S1C4 (Larger Buffers)

In Class S1C2 we studied the impact of increasing all physical buffer sizes $C_{i,j}$ between machines from 2 to 8 and again varied the routing probabilities. The results in Table 8 are more accurate than those for smaller buffer sizes in Table 7. This is a typical pattern for decomposition approaches.

We next varied the failure probability of Machine M_3 from 0.01 to 0.2 for physical buffer sizes $C_{i,j}$ of 2 and 8 and the fixed routing probability $d_{2,3} = 0.9$. The results in Tables 9 and 10 indicate that the production rate estimate is still quite precise. The buffer level estimate deteriorates as the system becomes more and more unbalanced.

Problem Classes S1C5 and S1C6 are like Problem Classes S1C3 and S1C4 except for a different routing probability $d_{2,3} = 0.1$ instead of 0.9. The results in Tables 11 and 12 indicate that the algorithmic behavior is very similar with respect to accuracy. However, the manufacturing system behavior is completely different: In Classes S1C3 and S1C4, Machine M_3 becomes a bottleneck as its failure probability increases up to 0.2. This is because Machine M_3 has to process 90 % of the parts that leave Machine M_2. When we reduce the routing probability $d_{2,3}$ to 0.1, Machine M_3 does not become a bottleneck for failure probabilities up to 0.2 and the production rate in

Class S1C5: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 1,2,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.1$						
Case	p_3	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C5S1	0.01	12	7.453	.0779 ± .0005	.0796	2.2
S1C5S2	0.04	12	7.521	.0783 ± .0004	.0794	1.4
S1C5S3	0.07	12	7.506	.0774 ± .0004	.0791	2.3
S1C5S4	0.1	12	7.528	.0770 ± .0003	.0789	2.4
S1C5S5	0.2	12	7.585	.0764 ± .0004	.0780	2.1

Table 11: Results for Class S1C5 (Small Buffers)

Class S1C6: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 1,2,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.1$						
Case	p_3	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C6S1	0.01	12	2.915	.0828 ± .0005	.0834	0.7
S1C6S2	0.04	12	2.996	.0826 ± .0004	.0834	0.9
S1C6S3	0.07	12	2.970	.0820 ± .0005	.0834	1.6
S1C6S4	0.1	12	2.948	.0823 ± .0004	.0833	1.2
S1C6S5	0.4	12	2.877	.0819 ± .0005	.0833	1.7

Table 12: Results for Class S1C6 (Larger Buffers)

Tables 11 and 12 remains almost unchanged.

In Problem Classes S1C7 and S1C8, we varied the failure probability of Machine M_1 and set the routing probability $d_{2,3}$ back to 0.9. Since all parts have to be processed by machine M_1 and we start with a balanced system, we observe in Tables 13 and 14 a strong reduction of the production rate which is precisely predicted by the approximation approach.

Class S1C7: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 2,3,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.9$						
Case	p_1	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C7S1	0.01	13	7.473	.7010 ± .0021	.7165	2.2
S1C7S2	0.04	13	5.533	.5720 ± .0019	.5861	2.5
S1C7S3	0.07	13	5.927	.4841 ± .0019	.4935	1.9
S1C7S4	0.1	13	5.807	.4204 ± .0014	.4257	1.2
S1C7S5	0.2	13	4.504	.2888 ± .0015	.2913	0.9

Table 13: Results for Class S1C7 (Small Buffers)

Class S1C8: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 2,3,4$; $r_i = 0.1, \forall i$; $d_{2,3} = 0.9$						
Case	p_1	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S1C8S1	0.01	13	2.967	.7431 ± .0020	.7502	1.0
S1C8S2	0.04	13	3.652	.6092 ± .0024	.6158	1.1
S1C8S3	0.01	13	3.391	.5104 ± .0024	.5153	0.9
S1C8S4	0.1	13	3.021	.4393 ± .0017	.4418	0.6
S1C8S5	0.2	13	2.160	.2979 ± .0015	.2981	0.1

Table 14: Results for Class S1C8 (Larger Buffers)

4.2.2 Structure S2

In a next step we added another machine downstream of Machine M_2. The resulting Structure S2 depicted in Figure 12 was evaluated for different routing probabilities. We asked for the production rate in the branch between Machines M_2 and M_3. We again generated 100 random problems which were sorted according to simulated production rates (see Figure 13). The routing probabilities were $d_{2,3} = 0.9$, $d_{2,4} = 0.05$, and $d_{2,4} = 0.05$. The percentage errors for each case are given in Figure 14. The algorithm converged in 99 out 100 cases and the average absolute value of the percentage error over the 99 random cases was 0.60 %. In the one case where the algorithm did not converge, all repair probabilities were very close to 1.

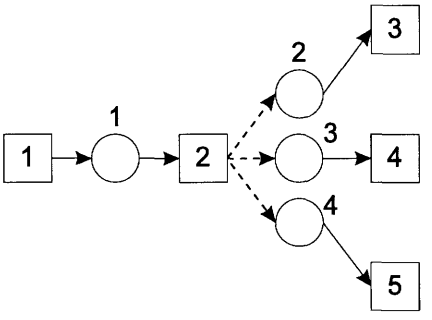

Figure 12: Structure S2

In the second part of the experiments for Structure S2, we briefly evaluated the effect of different routing probabilities. The results reported in Tables 15 and 16 show that the accuracy again increases as the buffer sizes increase. The decrease of the approximated production rate of Machine M_3

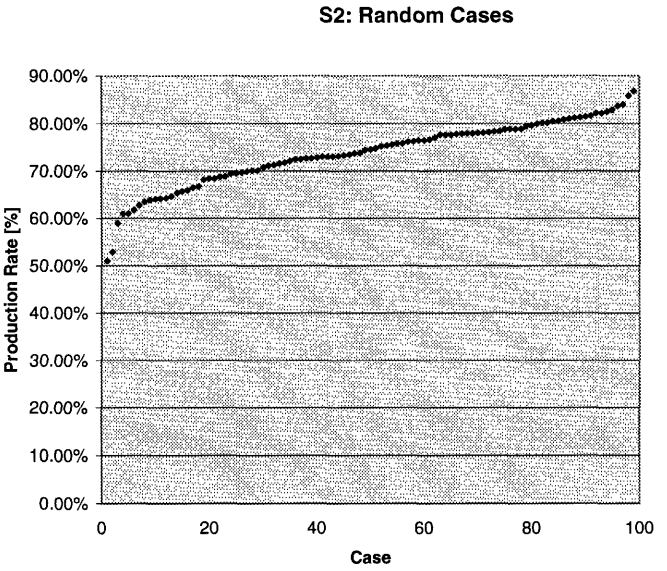

Figure 13: Structure S2 - Simulated Production Rates for Random Problems

Figure 14: Structure S2 - Percentage Errors for Random Problems

in Table 15 from 0.7168 to 0.4843 is almost proportional to the decrease of the routing probability $d_{2,3}$ from 0.9 to 0.6.

Class S2C1: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, r_i = 0.1, \forall i$								
Case	$d_{2,3}$	$d_{2,4}$	$d_{2,5}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S2C1S1	0.9	0.05	0.05	13	5.752	.7016 ± .0022	.7168	2.2
S2C1S2	0.8	0.1	0.1	13	6.035	.6283 ± .0016	.6414	2.1
S2C1S3	0.6	0.3	0.1	13	6.146	.4754 ± .0023	.4843	1.9

Table 15: Results for Class S2C1 (Small Buffers)

Class S2C2: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, r_i = 0.1, \forall i$								
Case	$d_{2,3}$	$d_{2,4}$	$d_{2,5}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S2C2S1	0.9	0.05	0.05	13	2.488	.7430 ± .0015	.7502	1.0
S2C2S2	0.8	0.1	0.1	13	2.175	.6651 ± .0019	.6715	1.0
S2C2S3	0.6	0.3	0.1	12	2.184	.5048 ± .0012	.5081	0.7

Table 16: Results for Class S2C2 (Larger Buffers)

4.2.3 Structure S3

We next studied System S3 depicted in Figure 15, a network with multiple split operations. In the first part of the study, we again generated 100 random problems. In all of these problems, we set the routing probabilities to $d_{2,3} = 0.9, d_{2,5} = 0.1, d_{5,6} = 0.5, d_{5,7} = 0.5$ and asked for the production rate in the branch between Machines M_3 and M_4.

Since the procedure to generate the failure and repair probabilities leads to machines with roughly similar isolated efficiencies, the random cases for Structure S3 are extremely unbalanced. Due to the routing probabilities, the average workload of Machine M_4 is 18 times as high as the workload of Machines M_6 or M_7. Since we assume identical processing times at all machines and generate similar isolated efficiencies, Machines M_6 and M_7 are idle most of the time. We created these artificial problems merely to show the limits of the numerical method. Out of the 100 random problems, the decomposition approach was able to analyze 83. It failed to converge in 17 cases. In Figure 16 we display the simulated production rate estimates for these 83 cases, sorted according to the simulated production rate. Figure 17

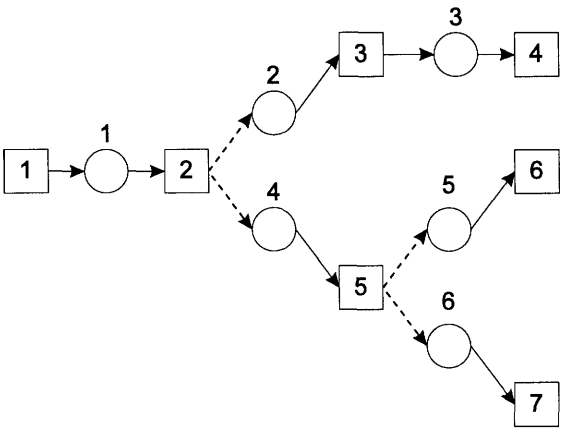

Figure 15: Structure S3

gives the respective percentage errors of the decomposition approach. The average over the 83 absolute values of the percentage error was 1.26 %. That is, the algorithm was less reliable for this larger and unbalanced system, but when it converged the results were still accurate.

Class S3C1:
$C_{i,j} = 2, \forall(i,j)$
$p_i = 0.01, i = 1,2,3,4; p_i = 0.1, i = 5,6,7; r_i = 0.1, \forall i$

Case	$d_{2,3}$	$d_{2,5}$	$d_{5,6}$	$d_{5,7}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S3C1S1	0.9	0.1	0.5	0.5	14	5.616	.6578 ± .0025	.6813	3.6
S3C1S2	0.5	0.5	0.5	0.5	14	6.213	.2876 ± .001	.3022	5.1

Table 17: Results for Class S3C1 (Small Buffers)

Class S3C2:
$C_{i,j} = 8, \forall(i,j)$
$p_i = 0.01, i = 1,2,3,4; p_i = 0.1, i = 5,6,7; r_i = 0.1, \forall i$

Case	$d_{2,3}$	$d_{2,5}$	$d_{5,6}$	$d_{5,7}$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
S3C2S1	0.9	0.1	0.5	0.5	14	3.041	.7254 ± .0016	.7398	2.0
S3C2S2	0.5	0.5	0.5	0.5	14	2.611	.3694 ± .0016	.3766	2.0

Table 18: Results for Class S3C2 (Larger Buffers)

We briefly studied different routing probabilities with a more evenly dis-

Figure 16: Structure S3 - Simulated Production Rates for Random Problems

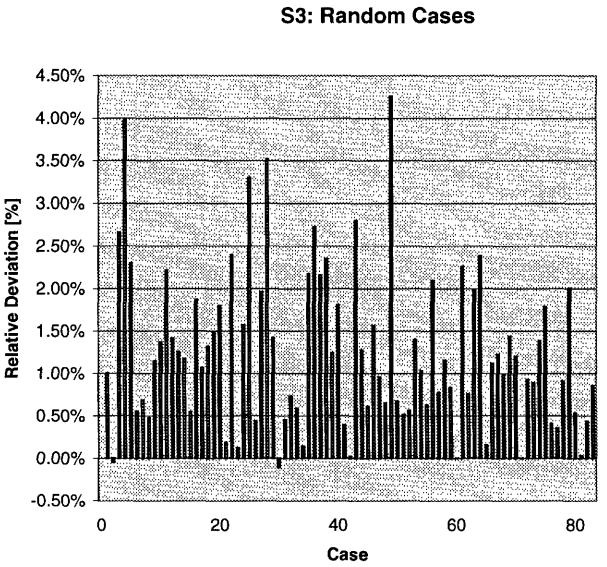

Figure 17: Structure S3 - Percentage Errors for Random Problems

tributed routing of the parts, for smaller as well as for larger buffer sizes. The parameters and the respective results are given in Tables 17 and 18 for the production rate in the branch between Machines M_3 and M_4.

Note that the isolated efficiencies of the machines are not identical. The results suggest that the deviation between simulated and approximated production rates increases slightly as we add another split operation, but the decomposition approach is still fairly accurate.

4.3 Pure Merge Networks

4.3.1 Structure M1

To study networks with merge operations, we started with Structure M1 in Figure 18. In this structure, Machine M_3 always tries to take the next part from its priority one input buffer between Machines M_1 and M_3. We asked for the production rate in the branch between Machines M_3 and M_4. Out of the 100 random problems generated for Structure M1, the decomposition algorithm was able to solve 91. A general observation is that the algorithm may fail to converge if the machine performing the merge operation is almost never starved and the starvation probability hence approaches zero. A possible explanation is that this leads to numerical problems in the flow rate-idle time equation (15) where a division by this starvation probability is performed.

The simulated production rates for these 91 cases are displayed in Figure 19, sorted according to simulated production rates. Figure 20 gives the respective percentage errors.

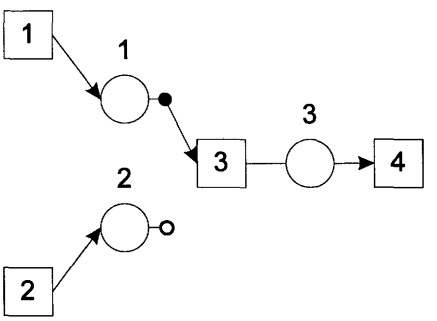

Figure 18: Structure M1

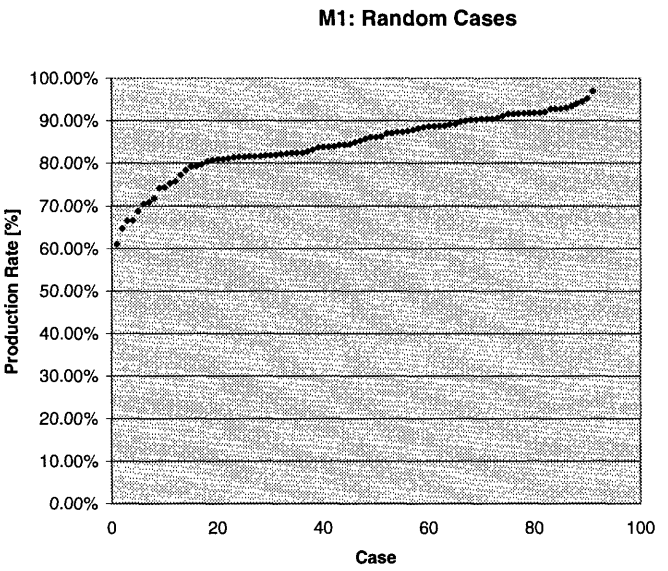

Figure 19: Structure M1 - Simulated Production Rates for Random Problems

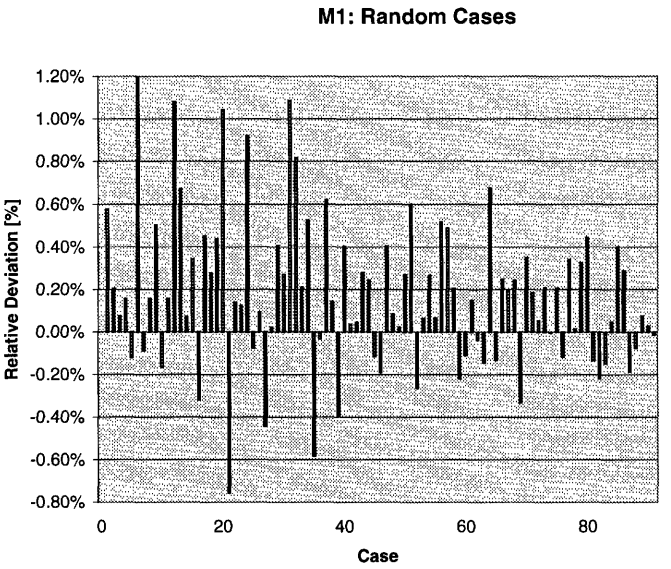

Figure 20: Structure M1 - Percentage Errors for Random Problems

Class M1C1: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 2,3,4$; $r_i = 0.1, \forall i$						
Case	p_1	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C1S1	0.001	21	24.771	.8347 ± .0022	.8409	0.7
M1C1S2	0.01	18	20.318	.8326 ± .0026	.8407	1.0
M1C1S3	0.04	17	16.085	.8271 ± .0025	.8380	1.3
M1C1S4	0.07	17	14.382	.8229 ± .0023	.8351	1.5
M1C1S5	0.1	17	14.094	.8160 ± .0025	.8323	2.0
M1C1S6	0.4	15	13.029	.7977 ± .0025	.8211	2.9
M1C1S7	0.7	15	10.758	.7898 ± .0026	.8143	3.1

Table 19: Results for Class M1C1 (Small Buffers)

The average absolute value of the percentage error was 0.29 % in the 91 cases where the algorithm converged. The convergence reliability for merge systems is in general smaller than for split systems. However, the production rate estimate for the merge machine is usually relatively accurate whenever the algorithm converges.

We then asked for systematic effects with respect to the manufacturing system behavior. In the first two classes of systems, we analyzed the impact of the failure probability p_1 and the buffer size $C_{i,j}$. The results in Tables 19 and 20 indicate the production rate estimate for the branch between Machines M_3 and M_4 is still quite accurate. The accuracy increases as the buffer sizes increase. We also see that there is only a modest decrease in the production rates of Machines M_3 and M_4 as the failure probability p_1 increases from 0.001 to 0.7. This is because of the high isolated efficiency of Machine M_2: As Machine M_1 fails more frequently, Machine M_2 is blocked less often, produces more and eventually carries the major part of the workload at this production stage.

Tables 21, 22, and 23 give detailed results for failure probabilities 0.001, 0.07, and 0.7 and physical buffer sizes $C_{i,j} = 2$. (The results for the simulated buffer levels include the half-width of the 95 % confidence intervals.)

We see in Table 21 that the simulated production rate of Machine M_2 is only 0.007 for $p_1 = 0.001$ and increases up to 0.6684 for $p_1 = 0.7$ (see Table 23). In this case (System M1C1S7), the isolated efficiency of Machine M_1 is only $\frac{0.1}{0.1+0.7} = 0.125$ which is close to the simulated production rate of 0.1214.

The accuracy of the production rate estimate is now no longer the same for all branches of the network. Tables 21 and 22 show that there may be

Class M1C2: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 2,3,4$; $r_i = 0.1, \forall i$						
Case	p_1	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C2S1	0.001	22	9.872	.8502 ± .0016	.8562	0.7
M1C2S2	0.01	19	8.033	.8535 ± .0015	.8562	0.3
M1C2S3	0.04	18	8.215	.8500 ± .0024	.8557	0.7
M1C2S4	0.07	17	6.417	.8488 ± .0018	.8545	0.7
M1C2S5	0.1	17	8.163	.8448 ± .0021	.8538	1.1
M1C2S6	0.4	15	6.425	.8371 ± .0023	.8473	1.2
M1C2S7	0.7	15	4.118	.8294 ± .0019	.8436	1.7

Table 20: Results for Class M1C2 (Larger Buffers)

Line	BL_{Si}	BL_{Ap}	%	PR_{Si}	PR_{Ap}	%
(1 , 3)	2.15 ± .002	3.06	22.8	.8277 ± .0024	.8313	0.4
(2 , 3)	2.00 ± .000	3.98	49.5	.0070 ± .0005	.0097	38.9
(3 , 4)	1.92 ± .004	2.00	2.0	.8347 ± .0022	.8409	0.7

Table 21: Results for Case M1C1S1 (Failure Probability $p_1 = 0.001$)

Line	BL_{Si}	BL_{Ap}	%	PR_{Si}	PR_{Ap}	%
(1 , 3)	1.39 ± .005	1.68	7.2	.5173 ± .0025	.4822	-6.8
(2 , 3)	2.01 ± .002	3.41	35.2	.3056 ± .0024	.3529	15.5
(3 , 4)	1.89 ± .004	1.92	0.8	.8229 ± .0023	.8351	1.5

Table 22: Results for Case M1C1S4 (Failure Probability $p_1 = 0.07$)

Line	BL_{Si}	BL_{Ap}	%	PR_{Si}	PR_{Ap}	%
(1 , 3)	0.40 ± .003	.71	7.8	.1214 ± .0006	.1177	-3.1
(2 , 3)	2.00 ± .005	2.87	21.7	.6684 ± .0027	.6966	4.2
(3 , 4)	1.82 ± .007	1.70	-2.8	.7898 ± .0026	.8143	3.1

Table 23: Results for Case M1C1S7 (Failure Probability $p_1 = 0.7$)

high *relative* deviations for the *priority two input buffer* between Machines M_2 and M_3 if its production rate is very low in *absolute terms*. However, due to the low absolute values this appears to be only a minor problem in economic terms. It is interesting to compare some of the results for Structure M2 to results for two-machine lines. The production rate of a two-machine line consisting only of Machines M_3 and M_4 as given in Table 19 is 0.8409. It is an upper bound on the production rate the system in Problem Class M1C1. Compare this upper bound with the result for Case M1C1S1 in Table 19: In this case, Machine M_3 is very rarely starved and the simulated production rate is very close to the upper bound. The approximated production rate also approaches this bound as Machine M_1 fails less frequently.

Class M1C3: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 1,3,4$; $r_i = 0.1, \forall i$						
Case	p_2	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C3S1	0.001	18	20.565	.8331 ± .0030	.8409	0.9
M1C3S2	0.01	18	20.360	.8326 ± .0025	.8407	1.0
M1C3S3	0.04	19	19.470	.8277 ± .0021	.8398	1.5
M1C3S4	0.07	21	19.007	.8239 ± .0028	.8394	1.9
M1C3S5	0.1	20	19.059	.8190 ± .0016	.8399	2.6
M1C3S6	0.4	29	18.892	.8016 ± .0023	.8398	4.8
M1C3S7	0.7	32	18.484	.7975 ± .0019	.8394	5.3

Table 24: Results for Class M1C3 (Small Buffers)

Class M1C4: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 1,3,4$; $r_i = 0.1, \forall i$						
Case	p_2	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C4S1	0.001	19	7.481	.8539 ± .0013	.8562	0.3
M1C4S2	0.01	19	8.002	.8503 ± .0022	.8562	0.7
M1C4S3	0.04	25	7.668	.8511 ± .0021	.8560	0.6
M1C4S4	0.07	28	7.262	.8502 ± .0016	.8557	0.6
M1C4S5	0.1	27	6.487	.8473 ± .0020	.8552	0.9
M1C4S6	0.4	32	9.011	.8425 ± .0026	.8560	1.6
M1C4S7	0.7	28	12.313	.8385 ± .0022	.8562	2.1

Table 25: Results for Class M1C4 (Larger Buffers)

For the next two problem classes we varied p_2, the failure probability of the machine upstream of the priority two input buffer of Machine M_3. We

see in Tables 24 and 25 that this does again lead to a slight reduction of the simulated production rate which is not very accurately predicted by the decomposition approach.

Class M1C5: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 3,4$; $r_i = 0.1, \forall i$						
Case	$p_1 = p_2$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C5S1	0.001	21	24.845	.8352 ± .0028	.8409	0.7
M1C5S2	0.01	18	20.284	.8323 ± .0018	.8407	1.0
M1C5S3	0.04	18	12.561	.8079 ± .0024	.8283	2.5
M1C5S4	0.07	17	13.666	.7650 ± .0018	.8052	5.3
M1C5S5	0.1	18	7.272	.7204 ± .0023	.7567	5.0
M1C5S6	0.4	16	5.949	.3812 ± .0011	.3897	2.2
M1C5S7	0.7	15	4.135	.2468 ± .0007	.2488	0.8

Table 26: Results for Class M1C5 (Small Buffers)

Class M1C6: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 3,4$; $r_i = 0.1, \forall i$						
Case	$p_1 = p_2$	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C6S1	0.001	22	9.868	.8530 ± .0021	.8562	0.4
M1C6S2	0.01	19	8.040	.8509 ± .0023	.8562	0.6
M1C6S3	0.04	24	5.285	.8437 ± .0016	.8527	1.1
M1C6S4	0.07	25	2.883	.8220 ± .0018	.8354	1.6
M1C6S5	0.1	26	4.960	.7873 ± .0020	.8128	3.2
M1C6S6	0.4	16	3.741	.3983 ± .0014	.3987	0.1
M1C6S7	0.7	14	1.816	.2499 ± .0007	.2500	0.0

Table 27: Results for Class M1C6 (Larger Buffers)

In Classes M1C1 to M1C4, Machines M_1 and M_2 had a joint isolated production rate that was higher than the isolated production rates of Machines M_3 and M_4. We next studied the opposite situation and increased the failure probabilities p_1 and p_2 simultaneously in order to make these machines the bottleneck. Tables 26 and 27 show that there is now a strong decrease of the production rate. In Case M1C6S7 in Table 27, both bottleneck machines fail so often that they are almost never blocked and the simulated production rate of the network is very close to the sum of the two isolated production rates of Machines M_1 and M_2 of 0.25.

Class M1C7: $C_{i,j} = 2, \forall(i,j)$; $p_i = 0.01, i = 1,2,3$; $r_i = 0.1, \forall i$						
Case	p_4	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C7S1	0.001	18	23.363	.8985 ± .0021	.9021	0.4
M1C7S2	0.01	18	20.312	.8330 ± .0026	.8407	0.9
M1C7S3	0.04	20	27.667	.6687 ± .0029	.6758	1.1
M1C7S4	0.07	19	30.100	.5573 ± .0024	.5635	1.1
M1C7S5	0.1	19	30.865	.4801 ± .0026	.4830	0.6
M1C7S6	0.4	126	32.847	.1984 ± .0010	.1984	0.0
M1C7S7	0.7	****	****	.1245 ± .0006	****	****

Table 28: Results for Class M1C7 (Small Buffers)

Class M1C8: $C_{i,j} = 8, \forall(i,j)$; $p_i = 0.01, i = 1,2,3$; $r_i = 0.1, \forall i$						
Case	p_4	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M1C8S1	0.001	18	10.885	.9050 ± .0015	.9052	0.0
M1C8S2	0.01	19	8.052	.8518 ± .0022	.8562	0.5
M1C8S3	0.04	19	11.103	.6896 ± .0023	.6921	0.4
M1C8S4	0.07	20	12.156	.5731 ± .0026	.5761	0.5
M1C8S5	0.1	24	12.477	.4928 ± .0021	.4926	0.0
M1C8S6	0.4	****	****	.1993 ± .0008	****	****
M1C8S7	0.7	****	****	.1248 ± .0006	****	****

Table 29: Results for Class M1C8 (Larger Buffers)

In a last experiment for Structure M2, we varied the failure probability of Machine M_4, i.e. the bottleneck was now downstream of the merging machine. We found in Tables 28 and 29 that if the algorithm converged, the accuracy of the results was relatively high. However, for very low isolated production rates of Machine M_4 of $\frac{0.1}{0.1+0.4} = 0.2$ (System M1C8S6) or $\frac{0.1}{0.1+0.7} = 0.125$ (Systems M1C7S7 and M1C8S7), the algorithm failed to converge. In these situations, the probability to have *both* input buffers of the merging machine empty at the same time becomes very small and we conjecture that this causes problems in the flow rate-idle time equation as the probability that Machine M_3 is starved approaches 0.

4.3.2 Structure M2

We also examined the effect of having multiple merge operations and different priority assignments. The structure in Figure 21 contains two machines performing merge operations.

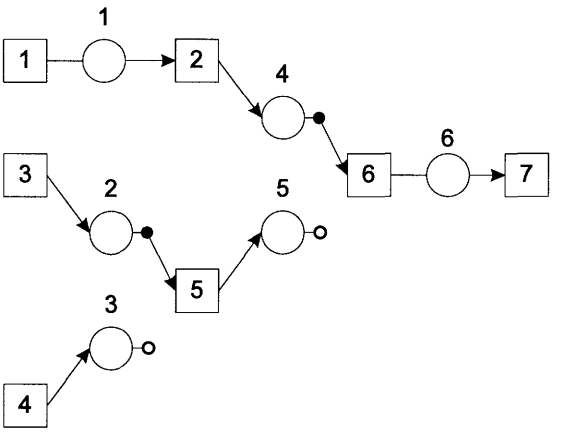

Figure 21: Structure for Class M2C1

We first generated 100 problems with machines of roughly the same isolated efficiency. Due to the two merge machines in Structure M2, these systems were strongly unbalanced and both of the merge machines were very rarely starved. Out of these 100 random cases, the algorithm could solve only 25, i.e. it failed to converge for 75 cases out of 100. We conjectured that this was due to the fact that the systems tended to be very unbalanced. In order to test this hypothesis, we modified the failure probabilities of the machines upstream of merge machines in order to create more balanced systems. For the 100 random cases, we changed the failure probabilities of Machines M_1, M_2, and M_5 in a way that reduced their isolated efficiencies by exactly 50 % and those of Machines M_3 and M_4 by exactly 75 % unless this resulted in failure probabilities higher than 0.9. In these cases, we set the failure probabilities to 0.9. Due to the merge operations, the systems were now more evenly balanced.

All the repair probabilities and buffer sizes remained unchanged. Out of these 100 random problems, the algorithm could solve 85 and failed to converge for 15. The average absolute percentage error over the 85 solved cases was 2.27 %. We conclude that structures with merge operations tend

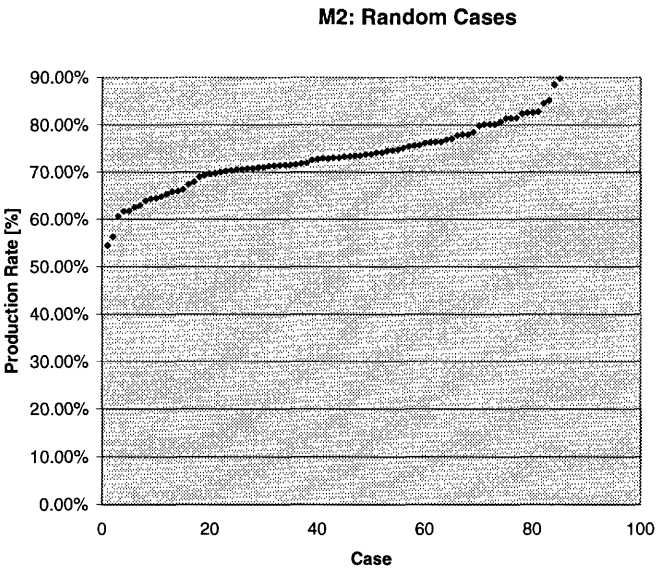

Figure 22: Structure M2 - Simulated Production Rates for Random Problems

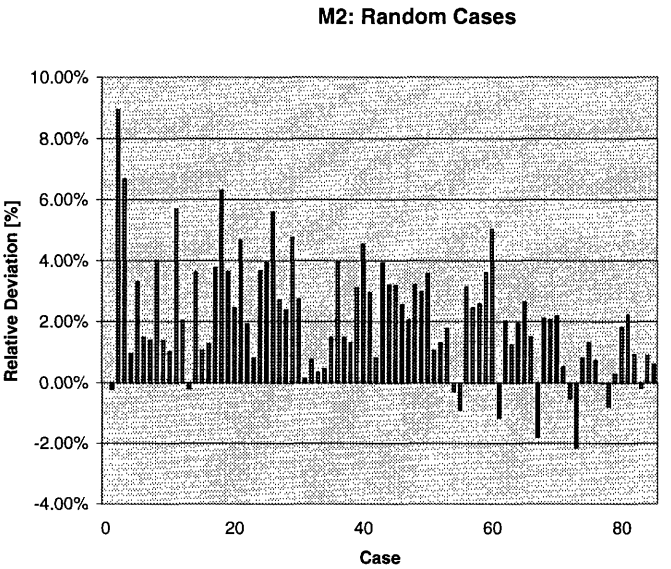

Figure 23: Structure M2 - Percentage Errors for Random Problems

$p_1 = p_2 = p_5 = 0.1, p_3 = p_4 = 0.2, p_6 = p_4 = 0.01$
$r_i = 0.1, \forall i$

System	$C_{i,j}, \forall(i,j)$	$B_{3,5}$	$B_{4,5}$	$B_{2,6}$	$B_{5,6}$
M2C1S1	2	Priority 1	Priority 2	Priority 1	Priority 2
M2C1S2	8	Priority 1	Priority 2	Priority 1	Priority 2
M2C2S1	2	Priority 1	Priority 2	Priority 2	Priority 1
M2C2S2	8	Priority 1	Priority 2	Priority 2	Priority 1
M2C3S1	2	Priority 2	Priority 1	Priority 1	Priority 2
M2C3S2	8	Priority 2	Priority 1	Priority 1	Priority 2
M2C4S1	2	Priority 2	Priority 1	Priority 2	Priority 1
M2C4S2	8	Priority 2	Priority 1	Priority 2	Priority 1

Table 30: Parameters for Cases M2C1S1 to M2C4S2

Case	Iter.	BL %	PR_{Si}	PR_{Ap}	%
M2C1S1	20	8.969	.6393 $\pm$ .0020	.6881	7.6
M2C1S2	18	8.024	.7470 $\pm$ .0018	.7786	4.2
M2C2S1	19	9.739	.6282 $\pm$ .0019	.6992	11.3
M2C2S2	27	7.862	.7407 $\pm$ .0017	.7619	2.9
M2C3S1	20	8.974	.6361 $\pm$ .0021	.6881	8.2
M2C3S2	18	8.009	.7463 $\pm$ .0019	.7786	4.3
M2C4S1	19	9.471	.6203 $\pm$ .0023	.6992	12.7
M2C4S2	27	7.435	.7323 $\pm$ .0019	.7619	4.0

Table 31: Results for Structure M2

to be more difficult to solve with respect to convergence reliability. The production rates and percentage errors are displayed in Figures 22 and 23.

We next changed the priority assignments. The priority assignments for the input buffers in Figure 21 are as in Cases M2C1S1 and M2C1S2 in Table 30. The results for the production rate in the branch between Machines M_6 and M_7 in Table 31 suggest that the quality of the approximation deteriorates slightly as we add machines performing merge operations. However, for a physical buffer size $C_{i,j} = 8$, the maximum relative deviation between approximated and simulated production rate was only 4.3 %. The priority assignment did not appear to have a major impact on the production rate.

We conclude that the algorithm is somewhat less accurate and reliable for pure merge networks than for pure split networks.

4.4 Structures with Loops

The last part of our numerical experiment was directed at more general structures that contain both split and merge operations and have loops. These loops in the flow of material occur if occasionally bad parts are produced, reworked, and sent back into the main line. Machines that perform split and merge operations are a building block of these more complex systems. Given that our decomposition works reasonably for pure split and pure merge structures, it remains to be shown that they can work together for a variety of different structures with random routing and loops.

4.4.1 Structure L1

We started with Structure L1 which is depicted in Figure 24. At Machine M_3, good parts are sent to Machine M_4 and finally leave the line. Bad parts are reworked at Machine M_5 and then sent to Machine M_2 where they have priority over the parts coming directly from Machine M_1. We asked for the production rate in the branch between Machines M_3 and M_4.

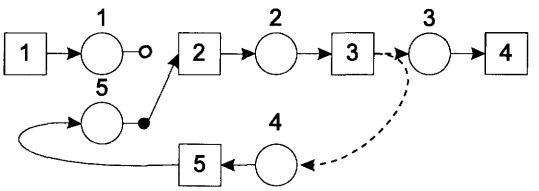

Figure 24: Structure L1

We first tested the algorithm for 100 random cases with roughly similar efficiencies over all machines. In these random cases, we assumed that 90 % of the parts are good and 10 % are bad. Out of the 100 random cases, 99 could be solved analytically by our decomposition approach. These were sorted according to the simulated production rates and are displayed in Figures 25 and 26. The average absolute percentage error over the 99 cases was 0.99 %.

We next studied systematically generated cases with different buffer sizes, frequencies of failures and repairs, and routing probabilities. Problem Classes L1C1 and L1C3 in Table 32 differ with respect to the variability of failures and repairs. In both cases, all machines have an isolated production rate of about 0.9091. However, in Class L1C3 failures (and repairs) occur much less often than in Class L1C1. The expected duration $\frac{1}{r_i}$ of a repair is 10

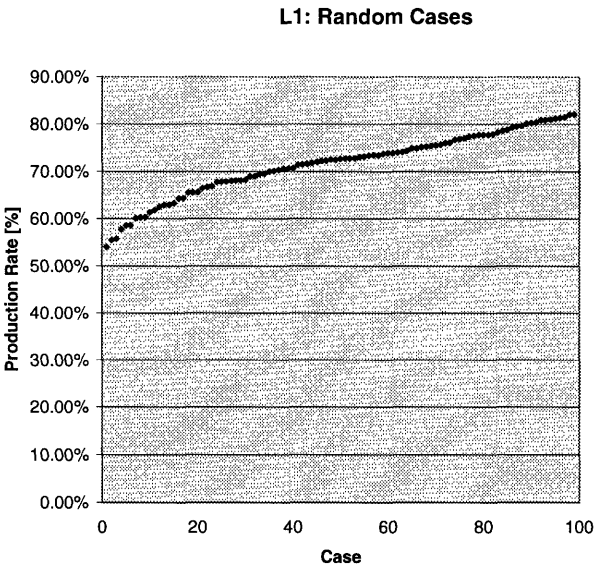

Figure 25: Structure L1 - Simulated Production Rates for Random Problems

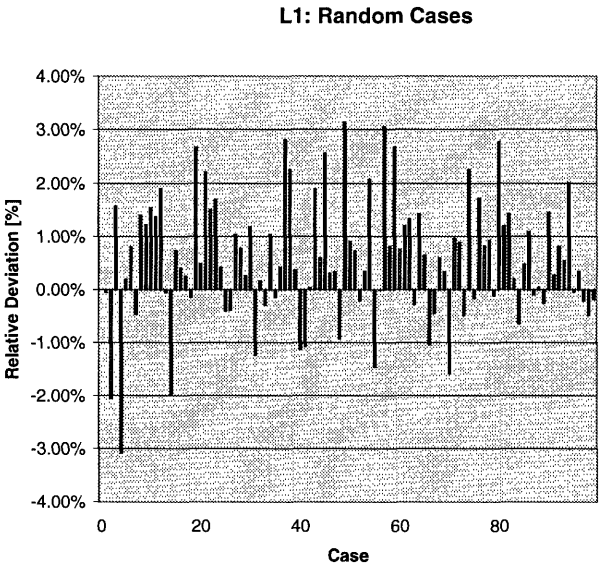

Figure 26: Structure L1 - Percentage Errors for Random Problems

Class L1C1			Class L1C3		
$p_i = 0.01, r_i = 0.1, \forall i, d_{3,4} = 0.9$			$p_i = 0.001, r_i = 0.01, \forall i, d_{3,4} = 0.9$		
System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$	System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$
L1C1S1	2	4	L1C3S1	2	4
⋮	⋮	⋮	⋮	⋮	⋮
L1C1S17	18	20	L1C3S17	18	20

Table 32: Parameters for Problem Classes L1C1 and L1C3

Class L1C2		Class L1C4	
$p_i = 0.01, r_i = 0.1, \forall i, C_{i,j} = 8, \forall(i,j)$		$p_i = 0.001, r_i = 0.01, \forall i, C_{i,j} = 8, \forall(i,j)$	
System	$d_{3,4}$	System	$d_{3,4}$
L1C2S1	0.9	L1C4S1	0.9
⋮	⋮	⋮	⋮
L1C2S9	0.1	L1C4S9	0.1

Table 33: Parameters for Problem Classes L1C2 and L1C4

time units in Class L1C1 and 100 time units in Class L1C3. Of the parts processed by Machine M_3, 90 % were routed to Machine M_4.

The graph in Figure 27 for Class L1C1 shows that the production rate increases (from 0.666 to 0.767) as we add buffer spaces. The decomposition approach overestimates the production rate slightly and becomes very accurate for larger buffer sizes. Compare this to the graph for Class L1C3 in Figure 28 with the larger expected repair times: First, there is almost no increase of the (simulated) production rate as we add 16 spaces for each buffer. This is due to the very large repair times: If a failure occurs, buffers become full or empty long before the machine is repaired, even for 'larger' buffer sizes. This example shows how crucial the impact of infrequent failures with long repair times is. The graph also shows that the decomposition approach fails with respect to accuracy if buffer sizes are very small with respect to repair times.

We now study the impact of the routing probability and vary $d_{3,4}$ from 0.9 to 0.1 for a physical buffer sizes $C_{i,j} = 8$, i.e. $N(i,j) = 10$ in all two-machine models used in the decomposition. Figure 29 shows that there is an almost perfectly linear decrease of the production rate as $d_{3,4}$ decreases and $d_{3,5}$ increases. We also see that the approximation works reasonable

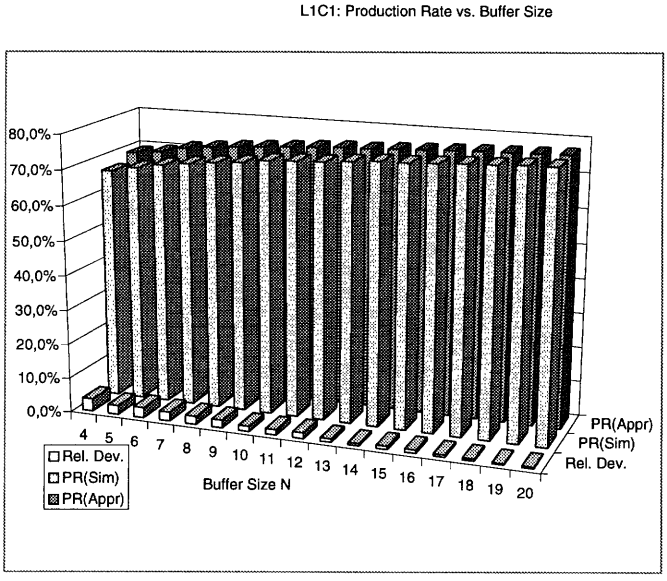

Figure 27: Results for Class L1C1

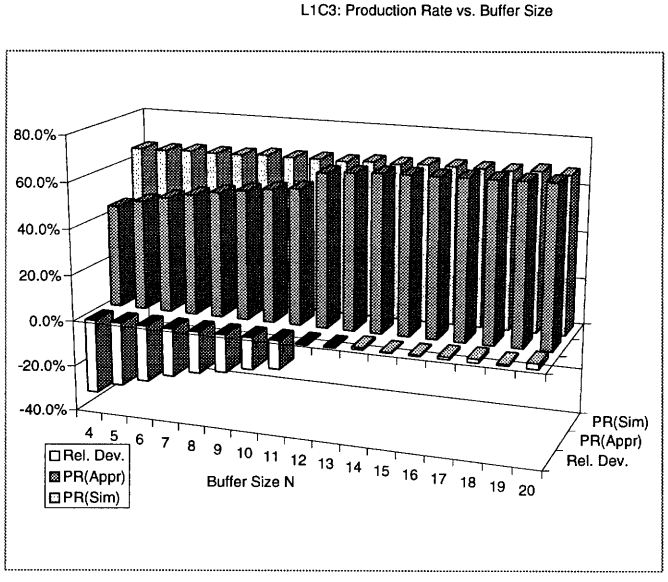

Figure 28: Results for Class L1C3

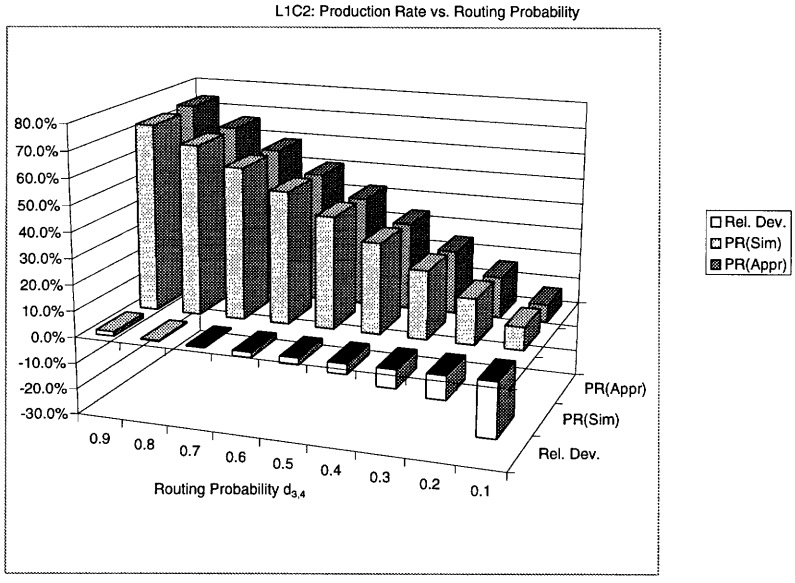

Figure 29: Results for Class L1C2

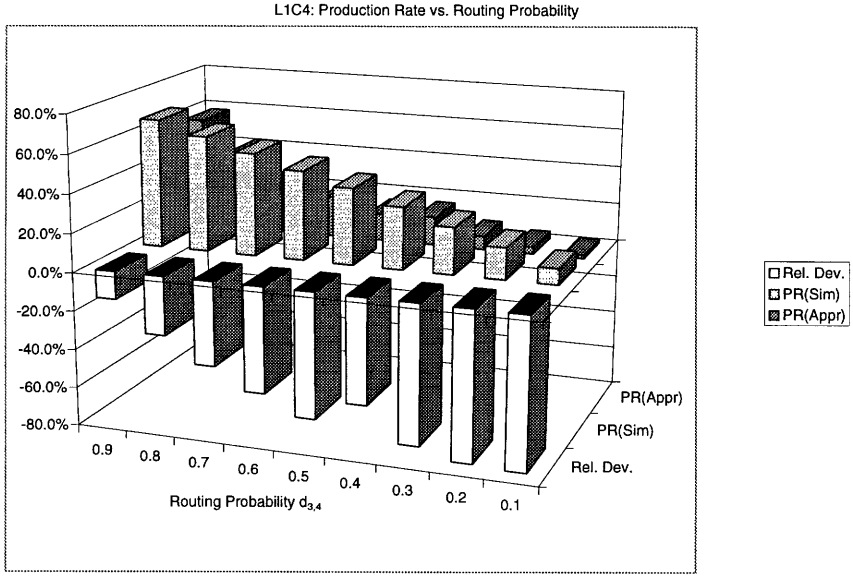

Figure 30: Results for Class L1C4

Class L1C5	
$p_i = 0.01, r_i = 0.1, \forall i, C_{i,j} = 2, \forall (i,j)$	
System	$d_{3,4}$
L1C5S1	0.9
⋮	⋮
L1C5S9	0.1

Table 34: Parameters for Problem Class L1C5

over a wide range of values. Again, this picture changes completely with respect to accuracy if the variability increases: Figure 30 indicates that the decomposition should not be used due to large deviations if high variability and high feedback probabilities occur together.

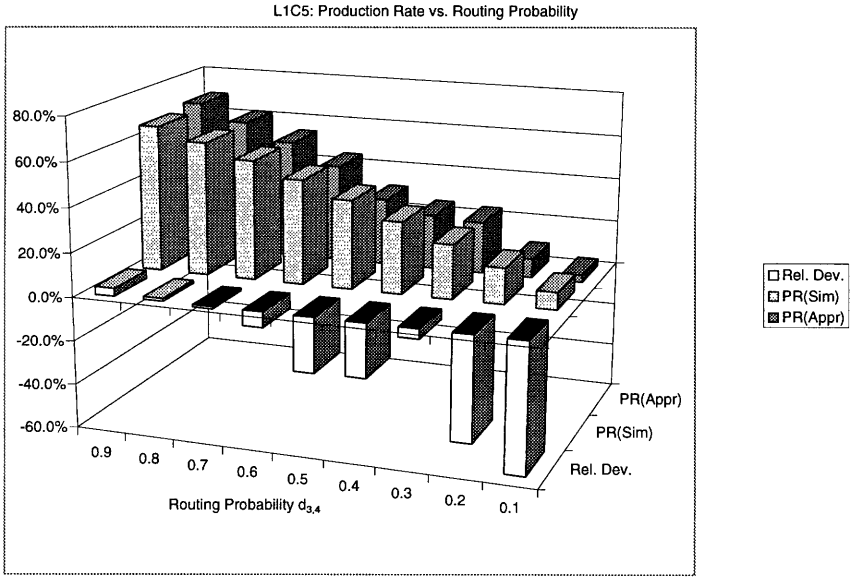

Figure 31: Results for Class L1C5

In a last experiment for Structure L1 (see Table 34), we studied the case of small buffer sizes ($C_{i,j} = 2, \forall (i,j)$), low variability, and varying routing probabilities. The results in Figure 31 are satisfying for routing probabilities $d_{3,4}$ higher than 0.7. For the purpose of modeling rejection and rework, this may still be acceptable. However, given the results in Figure 30, it is not

surprising that the approach failed to produce any useful results (which are not reported here) when we combined the small physical buffer size ($C_{i,j} = 2, \forall(i,j)$) with the high variability ($p_i = 0.001, r_i = 0.01, \forall i$).

4.4.2 Structure L2

To evaluate the effect of having a larger number of machines in the feedback loop, we next studied Structure L2 in Figure 32. Note that the loop in this structure consists of six machines as opposed to the three machines in Structure L1 in Figure 24. In the 100 random problems for this structure, we set the routing probabilities $d_{5,6} = 0.9$ and $d_{5,7} = 0.1$, i.e. we assumed that again 90 % of the parts are good. The algorithm converged in 95 out of 100 cases and the average absolute value of the percentage error was 1.38 % in these 95 cases. The results are depicted in Figures 33 and 34.

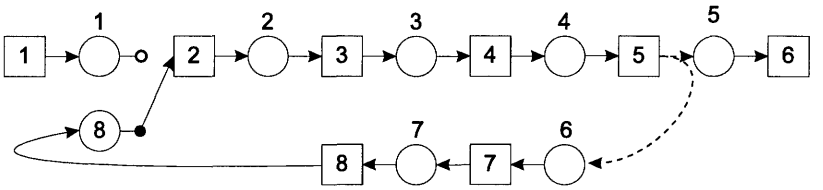

Figure 32: Structure L2

In the systematic part of the study, we started with Problem Classes L2C1 and L2C3 (see Table 35) which are analogous to L1C1 and L1C3. Figure 35 suggests that the approximation is slightly less accurate than for Structure L1 (compare with Figure 27). We observe a very similar increase of the production rate as we add buffer spaces. However, if we compare the results for the high-variability cases L2C3 in Figure 36 and L1C3 in Figure 28, we observe that for the larger network there are *fewer* cases where the decomposition resulted in very large deviations. The reason for this behavior does not appear to be obvious. We again observe that in the high-variability cases of Class L2C3 adding only 16 spaces to each buffer has almost no impact on the production rate (see Figure 36) as the amount produced during an average failure is still more than five times higher than the maximum buffer content.

A similar picture emerged when we varied the routing probability $d_{5,6}$ (see Table 36). Compare Figures 37 and 29: The approximation appears to be

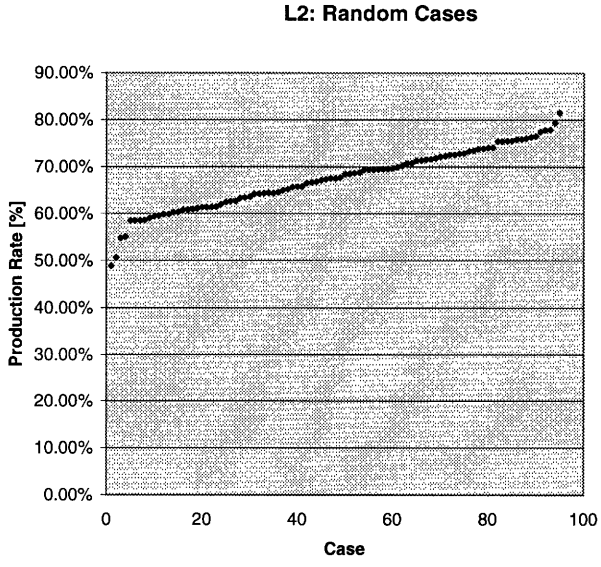

Figure 33: Structure L2 - Simulated Production Rates for Random Problems

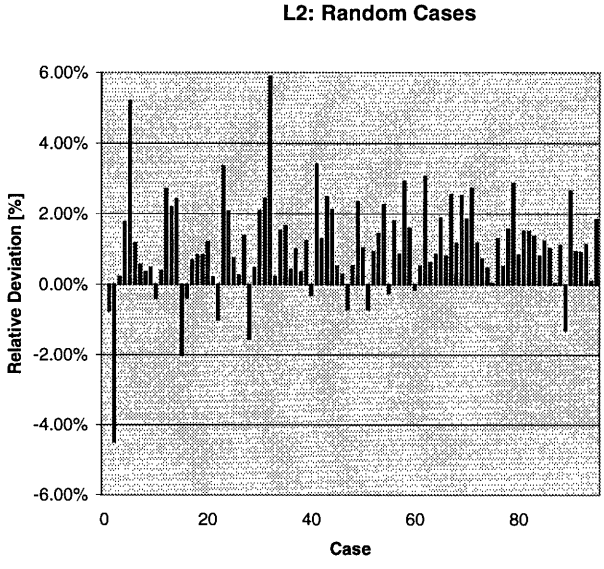

Figure 34: Structure L2 - Percentage Errors for Random Problems

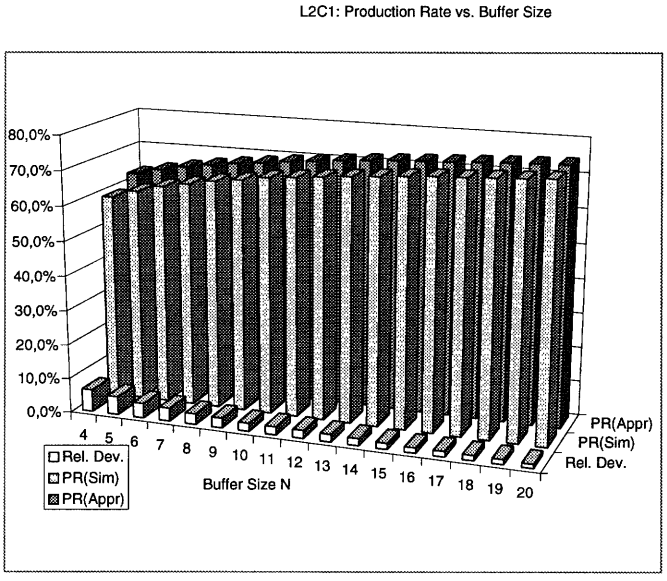

Figure 35: Results for Class L2C1

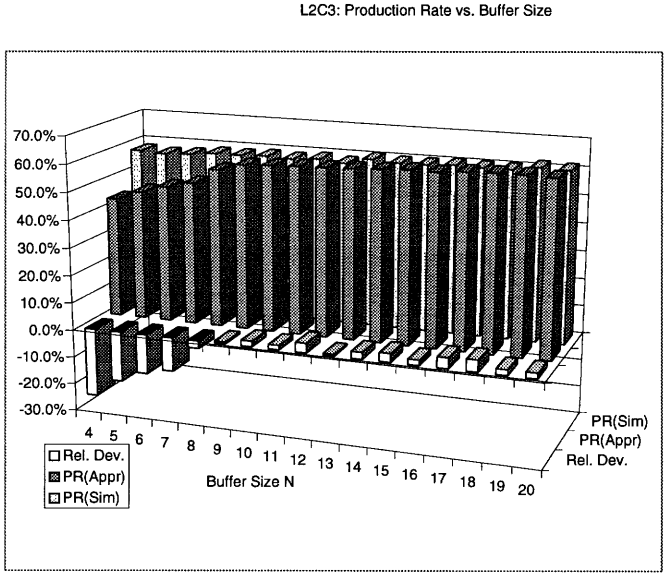

Figure 36: Results for Class L2C3

Class L2C1			Class L2C3		
$p_i = 0.01, r_i = 0.1, \forall i, d_{5,6} = 0.9$			$p_i = 0.001, r_i = 0.01, \forall i, d_{5,6} = 0.9$		
System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$	System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$
L2C1S1	2	4	L2C3S1	2	4
⋮	⋮	⋮	⋮	⋮	⋮
L2C1S17	18	20	L2C3S17	18	20

Table 35: Parameters for Problem Classes L2C1 and L2C3

Class L2C2		Class L2C4	
$p_i = 0.01, r_i = 0.1, \forall i, C_{i,j} = 8, \forall(i,j)$		$p_i = 0.001, r_i = 0.01, \forall i, C_{i,j} = 8, \forall(i,j)$	
System	$d_{5,6}$	System	$d_{5,6}$
L2C2S1	0.9	L2C4S1	0.9
⋮	⋮	⋮	⋮
L2C2S9	0.1	L2C4S9	0.1

Table 36: Parameters for Problem Classes L2C2 and L2C4

more accurate for the larger loop. This does even hold for the high-variability case L2C4S1 with the highest routing probability $d_{5,6} = 0.9$ (see Figure 38).

In Class L2C5 we studied the impact of very small buffers (see Table 37). Figure 39 compared to Figure 31 shows again that the decomposition works better for the larger loop as there are fewer extreme deviations.

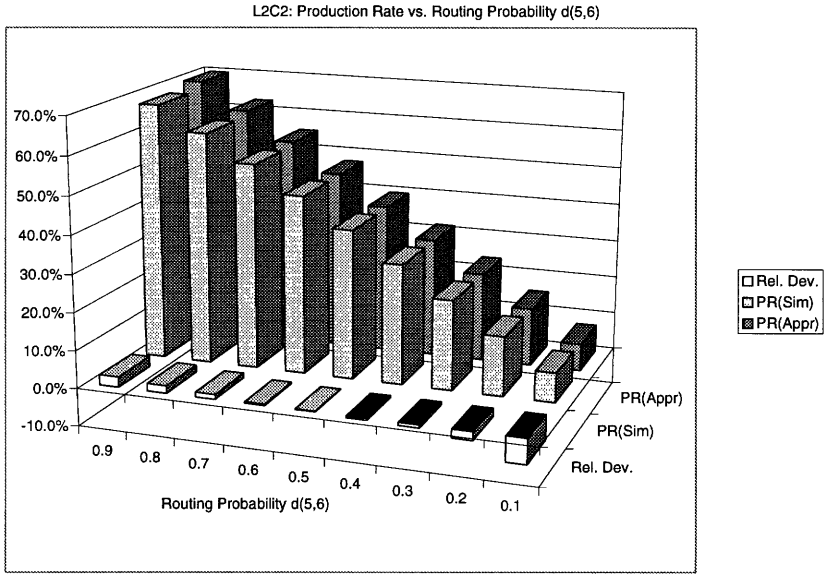

Figure 37: Results for Class L2C2

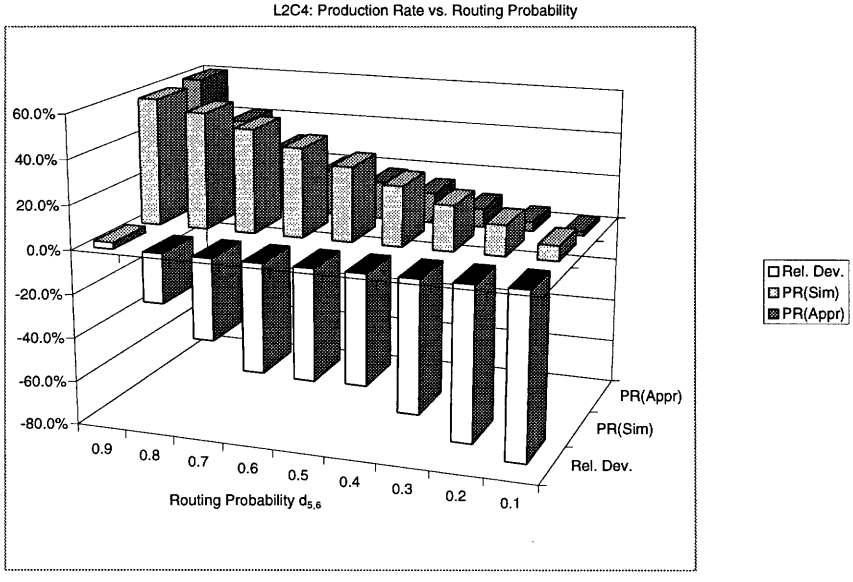

Figure 38: Results for Class L2C4

Class L2C5	
$p_i = 0.01, r_i = 0.1, \forall i, C_{i,j} = 2, \forall (i,j)$	
System	$d_{3,4}$
L2C5S1	0.9
⋮	⋮
L2C5S9	0.1

Table 37: Parameters for Problem Class L2C5

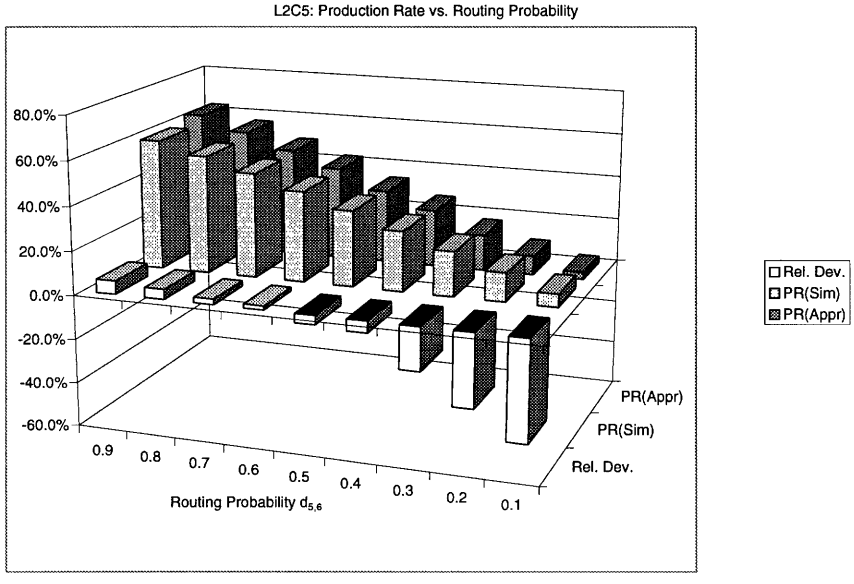

Figure 39: Results for Class L2C5

4.4.3 Structure L3

In Structures L1 and L2 we studied loops where a fraction of the material was randomly routed back to previous processing stages (feedback loop). We will now study two structures with feedforward loops using a similar experimental design. The first Structure L3 is depicted in Figure 40. In the 100 random cases for this structure, we set the routing probabilities to $d_{2,3} = d_{2,4} = 0.5$. Out of the 100 random problems, the decomposition approach solved 78. The average error was 1.68 %. This suggest that feedforward loops are more difficult to analyze than feedback loops. Other cases to be described below confirm this impression. The results are depicted in Figures 41 and 42.

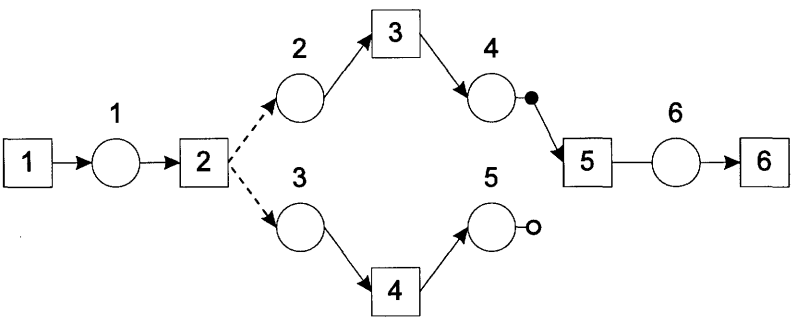

Figure 40: Structure L3

In the next step, we again systematically varied buffer sizes and routing probabilities. Figures 43 and 44 give the results for the parameters in Table 38. They suggest that the decomposition results in less extreme deviations compared to what we saw for the feedback loops in Structures L1 and L3 (compare with Figures 27 and 28). Interestingly, the quality of the approximation does not become better as we add buffer spaces for this feedforward structure.

We also varied the routing probability $d_{2,3}$, see Table 39. Figure 45 shows that the procedure worked reasonable for all values of $d_{2,3}$ but 0.9. In this case it failed to converge. Note that the simulated production rate is highest for $d_{2,3} = 0.5$. This is what we would expect for a system where the parallel branches have identical parameters. Unlike the results for structures with feedback loops, we have reasonable results even for the high-variability case in Figure 46.

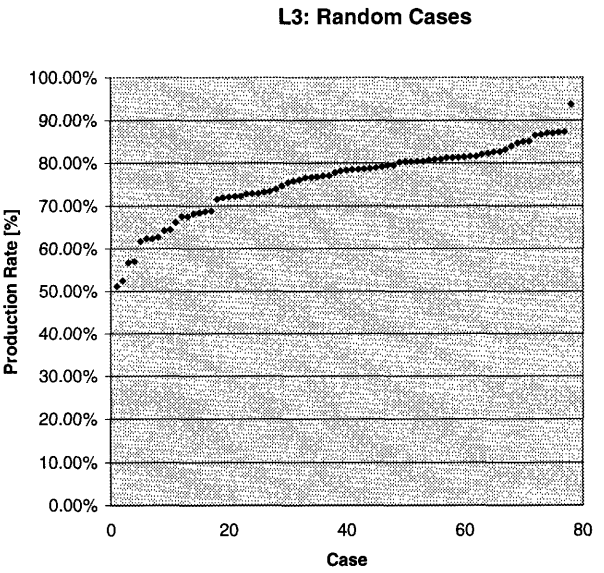

Figure 41: Structure L3 - Simulated Production Rates for Random Problems

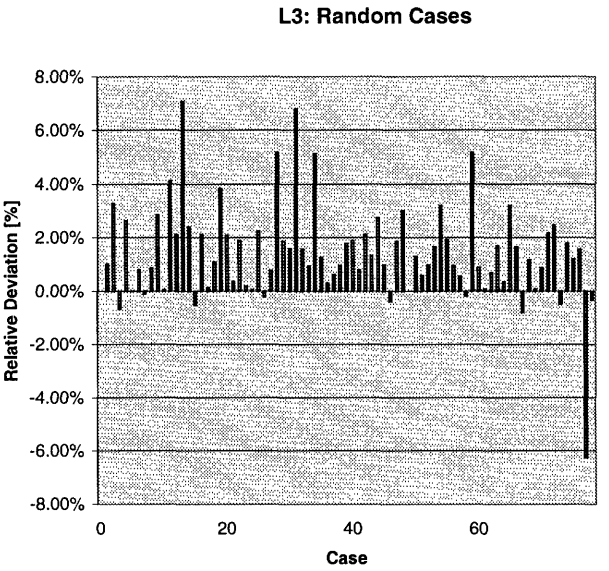

Figure 42: Structure L3 - Percentage Errors for Random Problems

Class L3C1			Class L3C3		
$p_1 = p_2 = p_5 = p_6 = 0.01$ $p_3 = p_4 = 0.1$ $r_i = 0.1, \forall i, d_{2,3} = d_{2,4} = 0.5$			$p_1 = p_2 = p_5 = p_6 = 0.001$ $p_3 = p_4 = 0.01$ $r_i = 0.01, \forall i, d_{2,3} = d_{2,4} = 0.5$		
System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$	System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$
L3C1S1	2	4	L3C3S1	2	4
⋮	⋮	⋮	⋮	⋮	⋮
L3C1S17	18	20	L3C3S17	18	20

Table 38: Parameters for Problem Classes L3C1 and L3C3

Class L3C2		Class L3C4	
$p_1 = p_2 = p_5 = p_6 = 0.01$ $p_3 = p_4 = 0.1$ $r_i = 0.1, \forall i$ $C_{i,j} = 8, \forall(i,j)$		$p_1 = p_2 = p_5 = p_6 = 0.001$ $p_3 = p_4 = 0.01$ $r_i = 0.01, \forall i$ $C_{i,j} = 8, \forall(i,j)$	
System	$d_{2,3}$	System	$d_{2,3}$
L3C2S1	0.1	L3C4S1	0.1
⋮	⋮	⋮	⋮
L3C2S9	0.9	L3C4S9	0.9

Table 39: Parameters for Problem Classes L3C2 and L3C4

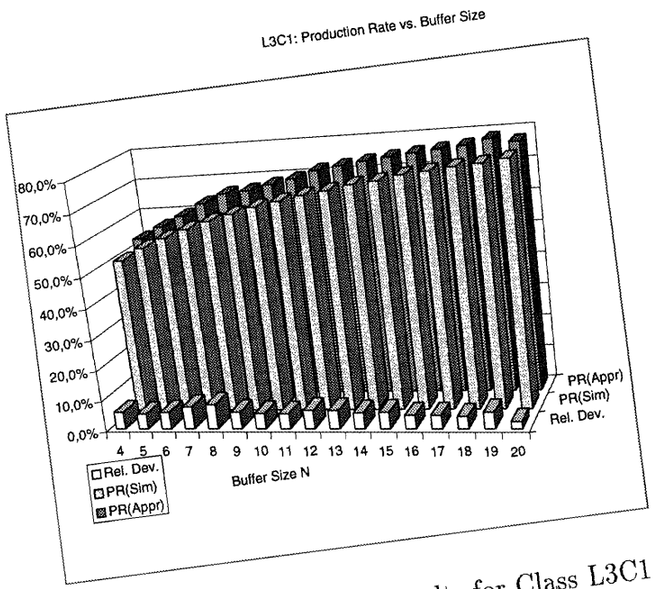

Figure 43: Results for Class L3C1

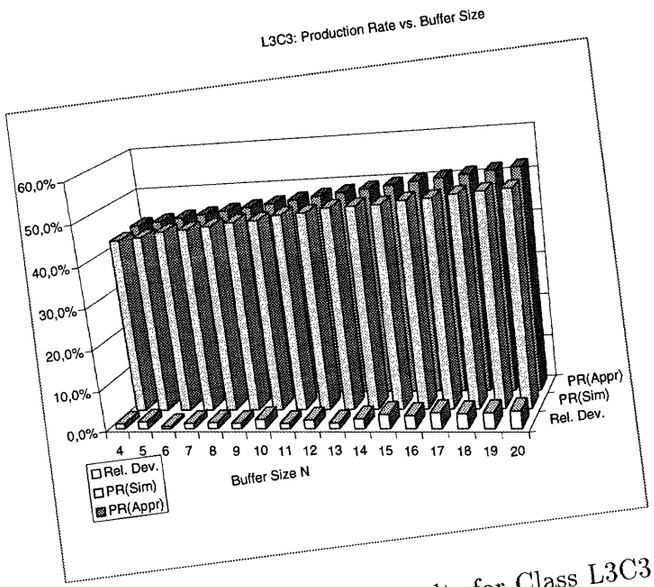

Figure 44: Results for Class L3C3

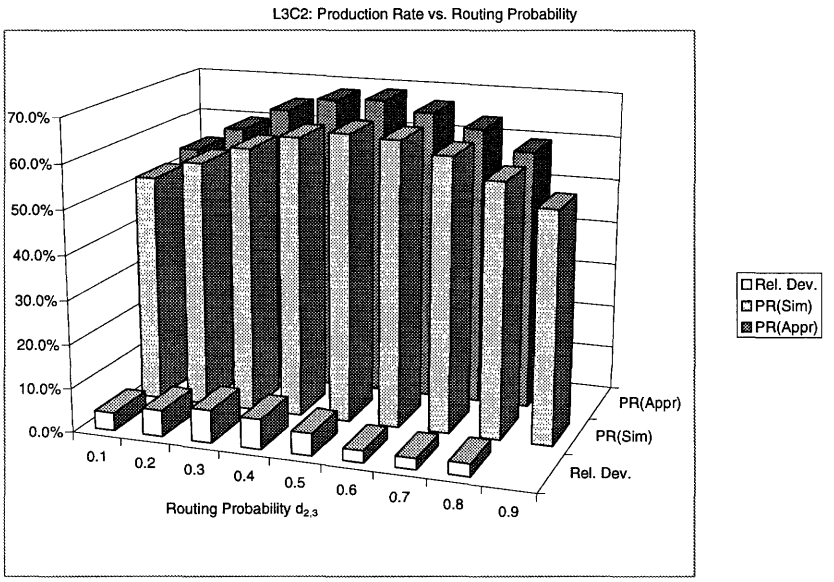

Figure 45: Results for Class L3C2

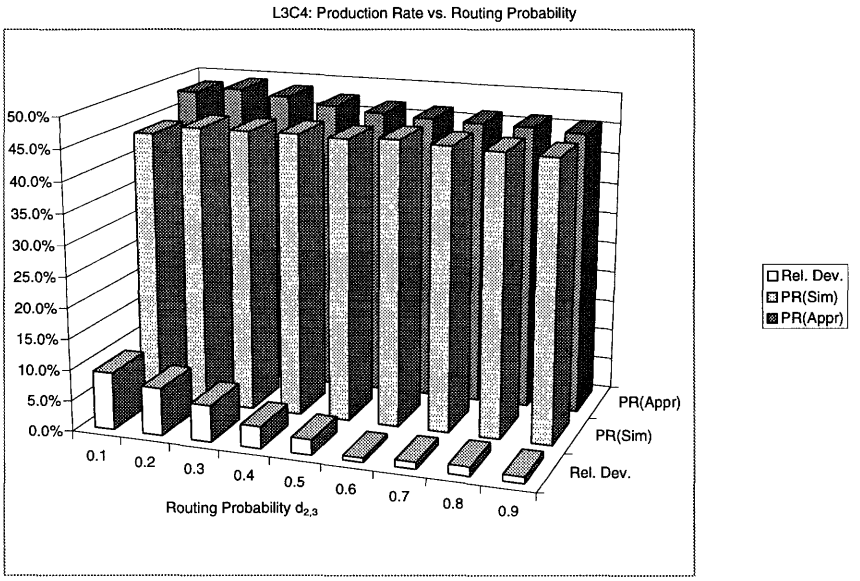

Figure 46: Results for Class L3C4

4.4.4 Structure L4

We now study Structure L4 depicted in Figure 47 with four instead of two machines in the parallel branches. Remember that we observed a higher accuracy of the decomposition for larger feedback loops. The routing probabilities in the 100 random cases were $p_{2,3} = p_{2,5} = 0.5$. Out of these 100 random cases, 77 could be solved with an average absolute value of the percentage error of 1.77 %. The results for the random cases are given in Figures 48 and 49.

The results in Figure 50 and 51 for the parameters in Table 40 indicate that the deviations get *larger* as we increase the number of machines in a feedforward loop. This does also appear to hold for varying routing probabilities (see Table 41 and Figures 52 and 53).

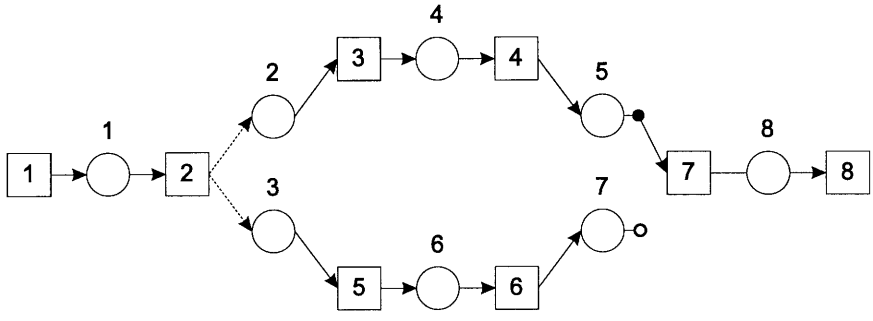

Figure 47: Structure L4

Class L4C1			Class L4C3		
$p_1 = p_2 = p_7 = p_8 = 0.01$ $p_3 = p_4 = p_5 = p_6 = 0.1$ $r_i = 0.1, \forall i, d_{2,3} = d_{2,5} = 0.5$			$p_1 = p_2 = p_7 = p_8 = 0.001$ $p_3 = p_4 = p_5 = p_6 = 0.01$ $r_i = 0.01, \forall i, d_{2,3} = d_{2,5} = 0.5$		
System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$	System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$
L4C1S1	2	4	L4C3S1	2	4
⋮	⋮	⋮	⋮	⋮	⋮
L4C1S17	18	20	L4C3S17	18	20

Table 40: Parameters for Problem Classes L4C1 and L4C3

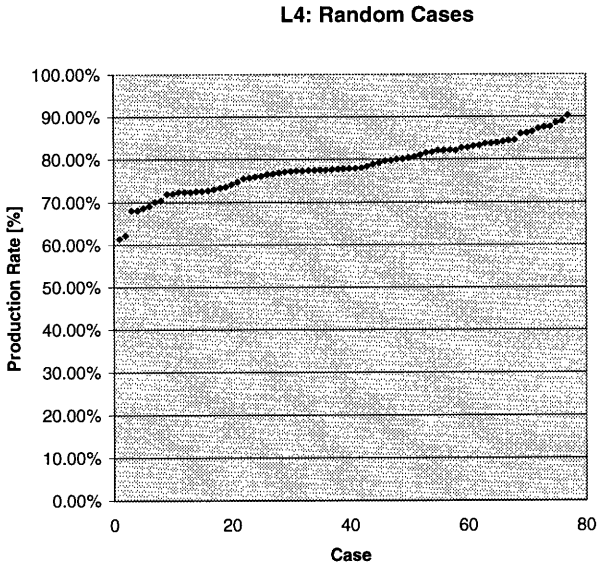

Figure 48: Structure L4 - Simulated Production Rates for Random Problems

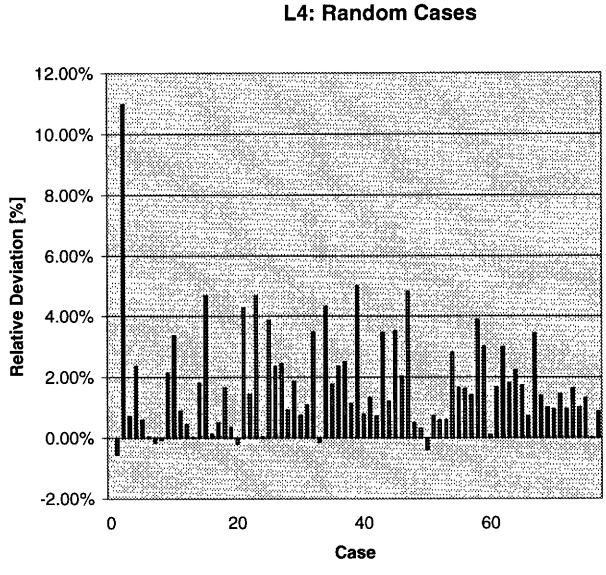

Figure 49: Structure L4 - Percentage Errors for Random Problems

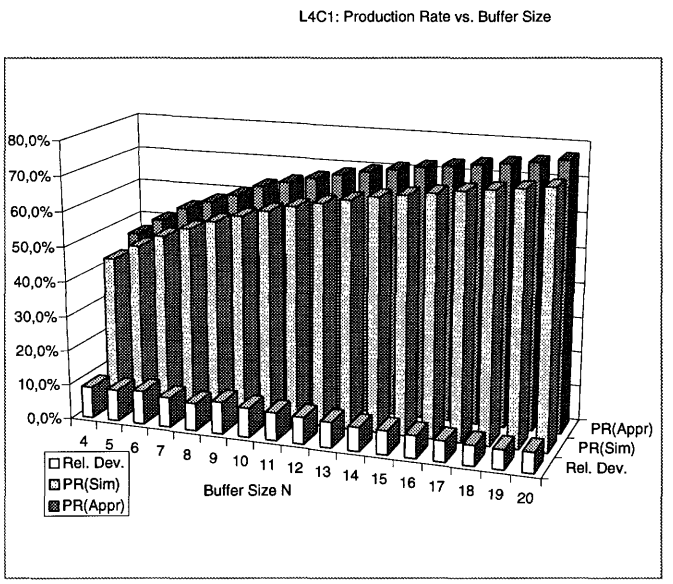

Figure 50: Results for Class L4C1

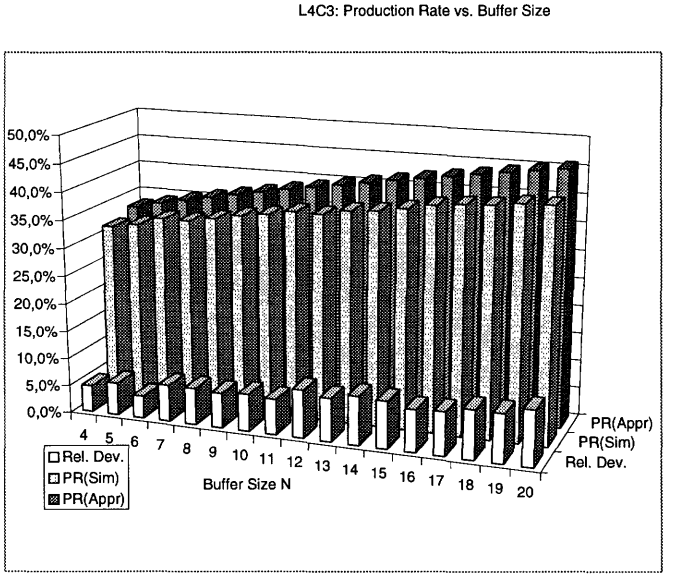

Figure 51: Results for Class L4C3

Class L4C2		Class L4C4	
$p_1 = p_2 = p_7 = p_8 = 0.01$ $p_3 = p_4 = p_5 = p_6 = 0.1$ $r_i = 0.1, \forall i$ $C_{i,j} = 8, \forall (i,j)$		$p_1 = p_2 = p_7 = p_8 = 0.001$ $p_3 = p_4 = p_5 = p_6 = 0.01$ $r_i = 0.01, \forall i$ $C_{i,j} = 8, \forall (i,j)$	
System	$d_{2,3}$	System	$d_{2,3}$
L4C2S1	0.1	L4C4S1	0.1
⋮	⋮	⋮	⋮
L4C2S9	0.9	L4C4S9	0.9

Table 41: Parameters for Problem Classes L4C2 and L4C4

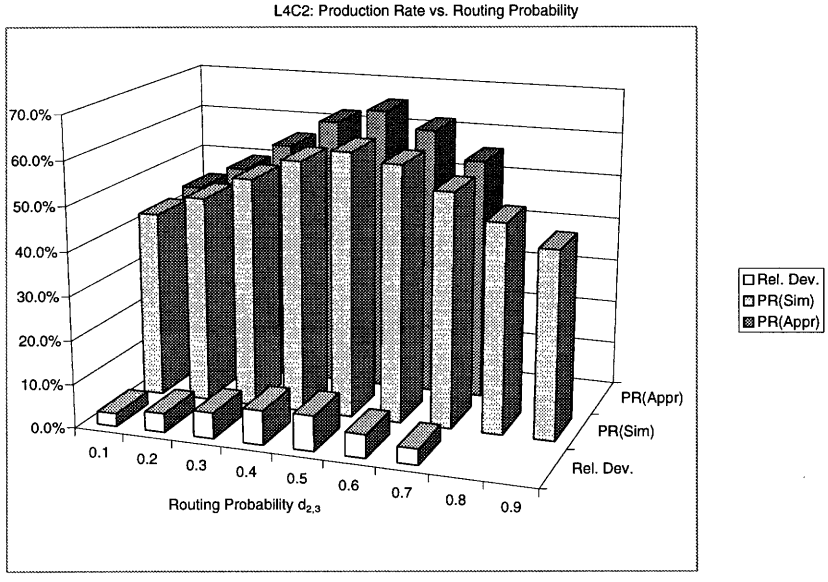

Figure 52: Results for Class L4C2

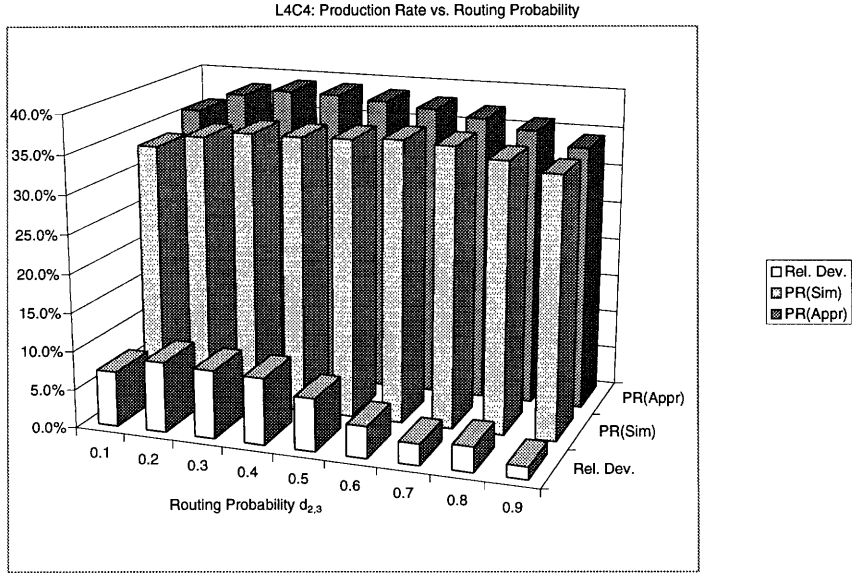

Figure 53: Results for Class L4C4

4.4.5 Structures L5 and L6

We next studied two different structures with multiple split and merge operations and a larger number of machines and buffers. Structure L5 is depicted in Figure 54. We asked for the production rate in the branch between Machines M_9 and M_{10}.

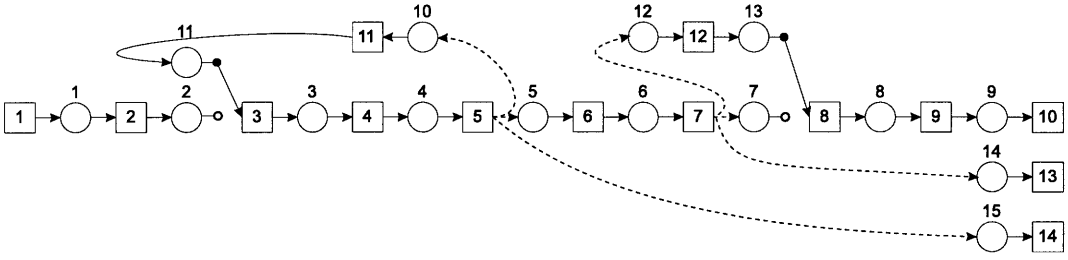

Figure 54: Structure L5

The routing probabilities in the 100 random cases were identical to those of the systematic study of Class L5C1 as given in in the upper left part of Table 42. 99 out of 100 random cases for Structure L5 could be solved with an average absolute value of the percentage error of 1.55 %. The results are given in Figures 55 and 56. For the systematically generated problems we used the parameters in the left column of Table 42.

Class L5C1			Class L6C1		
$p_i = 0.01, r_i = 0.1, \forall i$ $d_{5,6} = 0.8, d_{5,11} = 0.15, d_{5,14} = 0.05$ $d_{7,8} = 0.7, d_{7,12} = 0.25, d_{7,13} = 0.05$			$p_i = 0.01, r_i = 0.1, \forall i$ $d_{4,9} = 0.7, d_{4,14} = 0.2, d_{4,18} = 0.1$ $d_{8,9} = 0.7, d_{8,15} = 0.2, d_{8,19} = 0.1$ $d_{10,11} = 0.65, d_{10,16} = 0.3, d_{10,17} = 0.05$		
System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$	System	$C_{i,j}, \forall(i,j)$	$N_{i,j}, \forall(i,j)$
L5C1S1	2	4	L6C3S1	2	4
⋮	⋮	⋮	⋮	⋮	⋮
L5C1S17	18	20	L6C3S17	18	20

Table 42: Parameters for Problem Classes L5C1 and L6C1

Figure 57 shows that the decomposition works surprisingly well even for this rather complex structure if the number of the buffers is not too small.

We finally studied Structure L6 depicted in Figure 58 where we asked for the production rate in the branch between machines M_{12} and M_{13}. Note

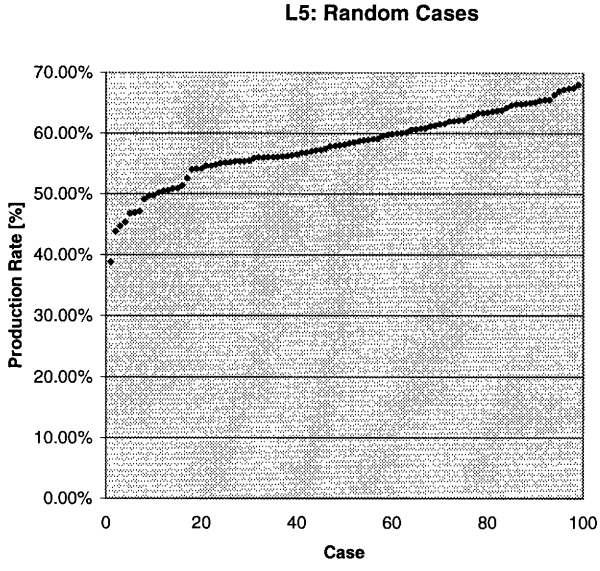

Figure 55: Structure L5 - Simulated Production Rates for Random Problems

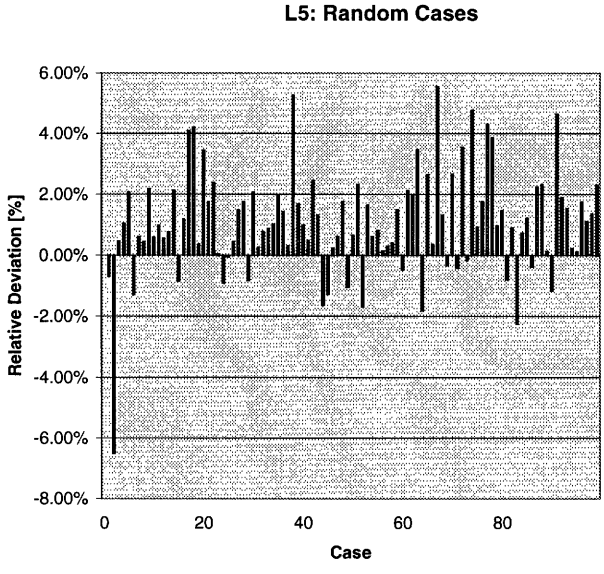

Figure 56: Structure L5 - Percentage Errors for Random Problems

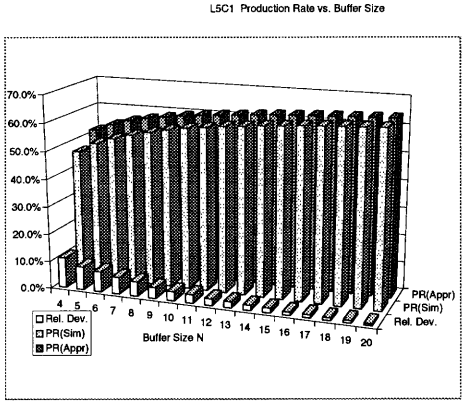

Figure 57: Results for Class L5C1

that this is basically a large merge structure with some additional loops. We used the routing probabilities in the upper right part of Table 42.

Given our experience with Structure M2, we expected this Structure to be difficult to analyze if all machines have similar isolated efficiencies and the system is hence very unbalanced. In fact, the algorithm converged for 62 out of 100 cases. We conjectured that the convergence problems were due to the merge operation performed by Machine M_9. For this reason, we modified the failure probabilities of all machines upstream of Machine M_9 in such a way that their isolated efficiencies were reduced by exactly 50 % whereever this was possible without exceeding an upper limit on the failure probabilities of 0.9. (Compare this to the generation of random cases for Structure M2). Out of these 100 modified random problems, the algorithm converged for 76 and failed for 24. The average absolute value of the percentage error was 1.65 %. We conclude that the convergence reliability decreases somewhat as the complexity of the structure increases, but when the algorithm converges, it still tends to produce surprisingly accurate results.

Figure 61 shows that the production rate increases in a way very similar to Problem Class L5C1 for the parameters on the right side of Table 42.

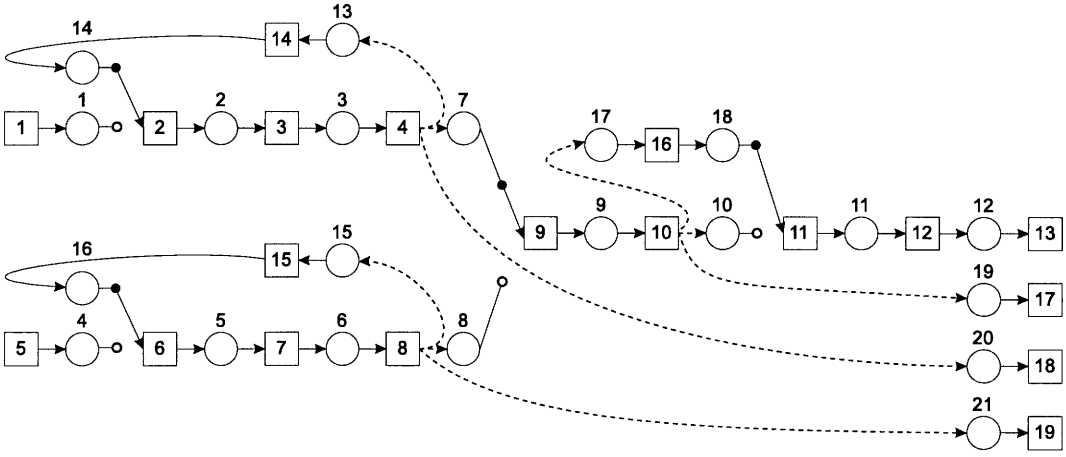

Figure 58: Structure L6

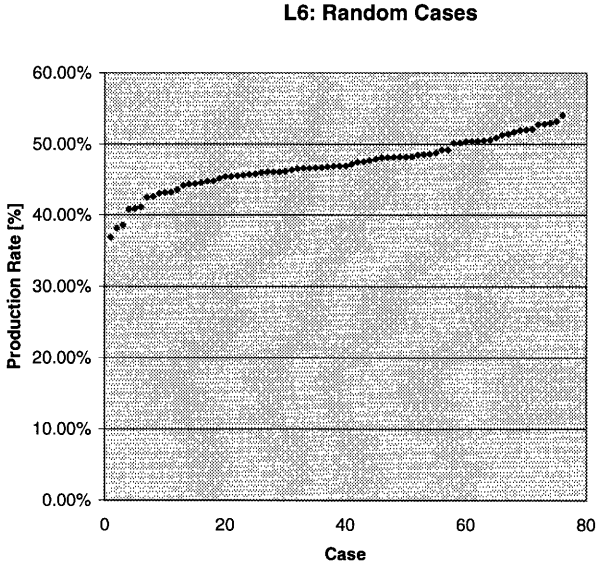

Figure 59: Structure L6 - Simulated Production Rates for Random Problems

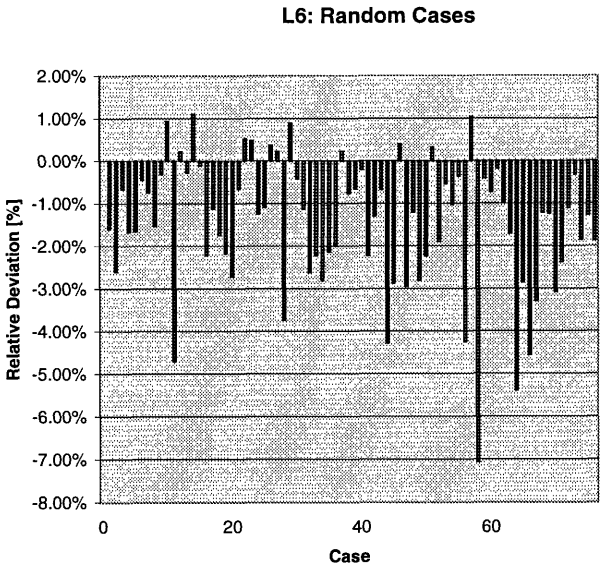

Figure 60: Structure L6 - Percentage Errors for Random Problems

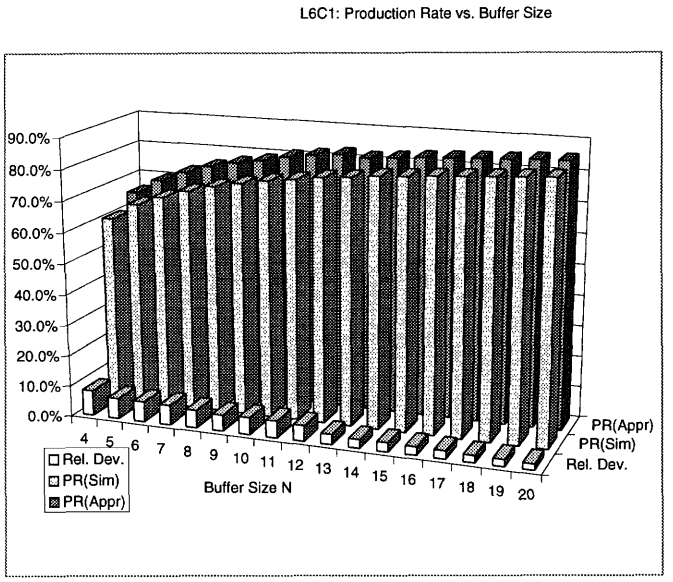

Figure 61: Results for Class L6C1

4.5 Summary of the Numerical Results

The numerical study shows that our decomposition approach provides good production rate estimates for a wide range of systems that perform split and/or merge operations. It appears to be very accurate and reliable for systems that perform split operations only, i.e. systems without rework of bad parts. For pure split structures, the accuracy increases as the buffer sizes increase and the algorithm converges very quickly. It is therefore a valuable tool for the evaluation of systems where bad parts are scrapped.

The decomposition for merge operations does provide very precise production rate estimates for the machine that has two input buffers. However, if the throughput of the priority two buffer is very low in absolute terms, the production rate estimate for this priority two buffer tends to be very inaccurate in relative terms. Since we assume that reworked parts in systems with loops will always be routed to the priority one buffer, the priority two buffer will usually not have such a low production rate unless almost all parts need to be reworked.

The iterative algorithm may fail to converge if the merging machine is almost never starved, for example because of an extreme bottleneck downstream. We conclude that the decomposition works well for a wide range of parameters that may be of practical interest.

Our study of more general systems with loops in the flow of material suggests that the method works well for a wide range of systems and parameters. For systems with feedback loops, the method tends to fail if buffer sizes are very small, compared to the number of parts processed during the expected repair time and a very large fraction of the parts has to be reworked. However, for reasonable buffer sizes and rejection probabilities, the method works well. Its accuracy appears to increase as buffer sizes increase.

For systems with feedforward loops, i.e. parallel branches, we observed fewer very large deviations, but also fewer very small deviations. Surprisingly, for this type of system the accuracy of the production rate estimate tended to decrease as the number of buffers increased. The convergence reliability appears to be lower than for feedback loops.

We eventually analyzed two larger systems with multiple loops and found that unless buffer sizes are very small, useful production rate estimates can be obtained even for systems with a much more complex flow of material than in a transfer line for which the first decomposition approaches have originally been developed.

5 Conclusions and Suggestions for Further Research

We have presented an approximate decomposition approach for unreliable transfer lines with split and merge operations and limited buffer capacity. We assumed identical deterministic processing times at all machines and took the processing time as the time unit in a discrete time model. The model as well as the decomposition equations are a generalization of the transfer line model in [Gershwin, 1987]. The DDX-type of algorithm to solve the equations is based on the work in [Dallery et al., 1988] and [Burman, 1995]. Despite some crude approximations, it was relatively tedious to develop some of the decomposition equations. However, a numerical study of more than 1400 different artificial system with split and/or merge operations indicated that the method works reasonable for a wide range of cases. It was even possible to analyze systems with loops in the flow of material.

Several extensions to our work are possible and important: It should be rather straightforward to combine the model presented in this paper and the assembly/disassembly model in [Gershwin, 1991, Gershwin, 1994] as both are generalizations of the transfer line model in [Gershwin, 1987]. It should also be possible to include split and merge operations in models with exponentially distributed processing times, times to failure and times to repair and the respective decomposition approaches. Since in reality machines often have deterministic but different processing times, it should be worthwhile to include split and merge operations in the continuous material model in [Burman, 1995]. Preliminary results in this area are encouraging, but the derivation of approximation equations is not easier. Another important field of study is the optimization of storage allocation. It should be possible to use an algorithm like the one developed in [Schor, 1995] to find the buffer allocation that maximizes the system production rate given the total available buffer space.

References

[Altiok, 1996] Altiok, T. (1996). *Performance Analysis of Manufacturing Systems.* Springer, New York et al.

[Artamonov, 1977] Artamonov, G. (1977). Productivity of a two-instrument discrete processing line in the presence of failures. *Cybernetics*, 12:464–468.

[Bürger, 1997] Bürger, M. (1997). *Konfigurationsplanung flexibler Fließproduktionssysteme.* PhD thesis. Submitted to the University of Koeln.

[Burman, 1995] Burman, M. H. (1995). *New results in flow line analysis.* PhD thesis, Massachusetts Institute of Technology. Also available as Report LMP-95-007, MIT Laboratory for Manufacturing and Productivity.

[Buxey et al., 1973] Buxey, G., Slack, N., and Wild, R. (1973). Production flow line system design - A review. *AIIE Transactions*, 5:37–48.

[Buzacott, 1972] Buzacott, J. (1972). The effect of station breakdowns and random processing times on the capacity of flow lines. *AIIE Transactions*, 4:308–312.

[Buzacott, 1967] Buzacott, J. A. (1967). Automatic transfer lines with buffer stocks. *International Journal of Production Research*, 5(3):183–200.

[Buzacott and Hanifin, 1978] Buzacott, J. A. and Hanifin, L. E. (1978). Models of automatic transfer lines with inventory banks — a review and comparison. *AIIE Transactions*, 10(2):197–207.

[Buzacott and Shanthikumar, 1993] Buzacott, J. A. and Shanthikumar, J. G. (1993). *Stochastic Models of Manufacturing Systems.* Prentice Hall, NJ, USA.

[Choong and Gershwin, 1987] Choong, Y. and Gershwin, S. (1987). A decomposition method for the approximate evaluation of capacitated transfer lines with unreliable machines and random processing times. *IIE Transactions*, 19:150–159.

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

[Dallery et al., 1988] Dallery, Y., David, R., and Xie, X.-L. (1988). An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers. *IIE Transactions*, 20(3):280–283.

[Dallery and Gershwin, 1992] Dallery, Y. and Gershwin, S. B. (1992). Manufacturing flow line systems: A review of models and analytical results. *Queuing Systems Theory and Applications*, 12(1-2):3–94. Special issue on queuing models of manufacturing systems.

[Di Mascolo et al., 1991] Di Mascolo, M., David, R., and Dallery, Y. (1991). Modeling and analysis of assembly systems with unreliable machines and finite buffers. *IIE Transactions*, 23(4):315–330.

[Gershwin, 1991] Gershwin, S. B. (1991). Assembly/disassembly systems: An efficient decomposition algorithm for tree-structured networks. *IIE Transactions*, 23(4):302–314.

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

[Gershwin and Schick, 1980] Gershwin, S. and Schick, I. (1980). Continuous model of an unreliable two-stage material flow system with a finite interstage buffer. Technical Report LIDS-R-1039, Massachusetts Institute of Technology, Cambridge, Massachusetts.

[Gershwin and Schick, 1983] Gershwin, S. and Schick, I. (1983). Modeling and analysis of three-stage transfer lines with unreliable machines and finite buffers. *Operations Research*, 31(2):354–380.

[Gershwin, 1989] Gershwin, S. B. (1989). An efficient decomposition algorithm for unreliable tandem queueing systems with finite buffers. In Perros, H. G. and Altiok, T., editors, *Queueing Networks with Blocking*, pages 127–146. North Holland, Amsterdam.

[Gershwin, 1994] Gershwin, S. B. (1994). *Manufacturing Systems Engineering.* PTR Prentice Hall, Englewood Cliffs, New Jersey.

[Gershwin and Berman, 1981] Gershwin, S. B. and Berman, O. (1981). Analysis of transfer lines consisting of two unreliable machines with random

processing times and finite storage buffers. *AIIE Transactions*, 13(1):2–11.

[Gopalan and Kannan, 1994] Gopalan, M. and Kannan, S. (1994). Expected duration analysis of a two-stage transfer-line production system subject to inspection and rework. *Journal of the Operational Research Society*, 45(7):797–805.

[Helber, 1995] Helber, S. (1995). Exact analysis of the two-machine transfer line with limited buffer capacity and geometrically distributed processing times, times to failure and times to repair. Technical report, Ludwig-Maximilians-Universität München, Ludwigstr. 28 RG/V., D-80539 München, Germany.

[Helber, 1997] Helber, S. (1997). Decomposition of unreliable assembly/dissassembly networks with limited buffer capacity and random processing times. Technical report, Ludwig Maximilians Universität, Ludwigstr. 28 RG, D-80539 München, Germany.

[Hillier and Boling, 1967] Hillier, F. and Boling, R. (1967). Finite queues in series with exponential or Erlang service times - A numerical approach. *Operations Research*, 16:286–303.

[Jafari and Shanthikumar, 1987] Jafari, M. A. and Shanthikumar, J. G. (1987). An approximate model of multistage automatic transfer lines with possible scrapping of workpieces. *IIE Transactions*, 19(3):252–265.

[Koenigsberg, 1959] Koenigsberg, E. (1959). Production lines and internal storage - a review. *Management Science*, 5:410–433.

[Okamura and Yamashina, 1977] Okamura, K. and Yamashina, H. (1977). Analysis of the effect of buffer storage capacity in transfer line systems. *AIEE Transactions*, 9:127–135.

[Papadopoulus et al., 1993] Papadopoulus, H., Heavey, C., and Browne, J. (1993). *Queueing Theory in Manufacturing Systems Analysis and Design.* Chapman & Hall, London et al.

[Pourbabai, 1990] Pourbabai, B. (1990). Optimal utilization of a finite capacity intergrated assembly system. *International Journal of Production Research*, 28(2):337–352.

[Sastry and Awate, 1988] Sastry, B. and Awate, P. (1988). Analysis of a two-station flow line with machine processing subject to inspection and rework. *Opsearch*, 25:89–97.

[Schmidbauer, 1995] Schmidbauer, H. (1995). *Zur Zuverlässigkeit von stochastischen Fertigungssystemen.* Habilitationsschrift, Fakultät für Philosophie, Wissenschaftstheorie und Statistik, Ludwig-Maximilians-Universität München.

[Schmidbauer and Rösch, 1994] Schmidbauer, H. and Rösch, A. (1994). A stochastic model of an unreliable kanban production system. *Operations Research Spectrum*, 16(2):155–160.

[Schor, 1995] Schor, J. E. (1995). Efficient algorithms for buffer allocation. Master's thesis, Massachusetts Institute of Technology. Also available as MIT Laboratory for Manufacturing and Productivity report LMP-95-006.

[Sevast'yanov, 1962] Sevast'yanov, B. A. (1962). Influence of storage bin capacity on the average standstill time of a production line. *Theory of Probability and Its Applications*, 7:429–438.

[Shanthikumar and Tien, 1983] Shanthikumar, J. and Tien, C. (1983). An algorithmic solution to two-stage transfer lines with possible scrapping of units. *Management Science*, 29:1069–1086.

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

[Yeralan and Muth, 1987] Yeralan, S. and Muth, E. J. (1987). A general model of a production line with intermediate buffer and station breakdown. *IIE Transactions*, 19(2):130–139.

[Yu and Bricker, 1993] Yu, K.-Y. C. and Bricker, D. L. (1993). Analysis of a markov chain model of a multistage manufacturing system with inspection, rejection, and rework. *IIE Transactions*, 25(1):109–112.

[Zimmern, 1956] Zimmern, B. (1956). Etudes de la propagation des arrêts aleatoires dans les chaines de production. *Review Statististical Applications*, 4:85–104.