George Liberopoulos
Chrissoleon T. Papadopoulos
Barış Tan · J. MacGregor Smith
Stanley B. Gershwin
Editors

Stochastic Modeling of Manufacturing Systems

Advances in Design, Performance Evaluation, and Control Issues

Springer

Stochastic Modeling of Manufacturing Systems

Advances in Design, Performance Evaluation, and Control Issues

G. Liberopoulos · C. T. Papadopoulos · B. Tan
J. MacGregor Smith · S. B. Gershwin
Editors

Stochastic Modeling of Manufacturing Systems

Advances in Design, Performance Evaluation, and Control Issues

With 121 Figures
and 91 Tables

Springer

George Liberopoulos
Department of Mechanical
and Industrial Engineering
University of Thessaly
38334 Volos
Greece
E-mail: glib@mie.uth.gr

J. M. Smith
Department of Mechanical
and Industrial Engineering
University of Massachusetts
Amherst, Massachusetts 01003
USA
E-mail: jmsmith@ecs.umass.edu

Chrissoleon T. Papadopoulos
Department of Economic Sciences
Aristotle University of Thessaloniki
54124 Thessaloniki
Greece
E-mail: hpap@econ.auth.gr

Stanley B. Gershwin
Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139-4307
USA
E-mail: gershwin@mit.edu

Barış Tan
Graduate School of Business
Koç University
80910 Sariyer, Istanbul
Turkey
E-mail: btan@ku.edu.tr

Parts of the papers of this volume have been published in the journal *OR Spectrum*.

Library of Congress Control Number: 2005930501

ISBN-10 3-540-26579-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-26579-5 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springeronline.com

Printed in Germany

Cover design: Erich Kirchner
Production: Helmut Petri
Printing: Strauss Offsetdruck

SPIN 11506560 Printed on acid-free paper – 42/3153 – 5 4 3 2 1 0

Editorial – Stochastic Modeling of Manufacturing Systems: Advances in Design, Performance Evaluation, and Control Issues

Manufacturing systems rarely perform exactly as expected and predicted. Unexpected events always happen: customers may change their orders, equipment may break down, workers may be absent, raw parts may not arrive on time, processed parts may be defective, etc. Such randomness affects the performance of the system and complicates decision-making. Responding to unexpected disturbances occupies a significant amount of time of manufacturing managers. There are two possible plans of action for addressing randomness: reduce it or respond to it in a way that limits its corrupting effect on system performance. This volume is devoted to the second. It includes fifteen novel chapters on stochastic models for the design, coordination, and control of manufacturing systems. The advantage of modeling is that it can lead to the deepest understanding of the system and give the most practical results, provided that the models apply well to the real systems that they are intended to represent. The chapters in this volume mostly focus on the development and analysis of performance evaluation models using decomposition-based methods, Markovian and queuing analysis, simulation, and inventory control approaches. They are organized into four distinct sections to reflect their shared viewpoints.

Section I includes a single chapter (Chapter 1) on factory design. In this chapter, Smith raises several concerns that must be addressed before even choosing a modeling approach and developing and testing a model. Specifically, he discusses a number of dilemmas in factory design problems and the paradoxes that they lead to. These paradoxes give rise to new paradigms that can bring on new approaches and insights for solving them.

Section II includes Chapters 2–7 on unreliable production lines with in-process buffers.

More specifically, in Chapter 2, Enginarlar, Li, and Meerkov analyze a tandem production line and determine the minimum buffer levels that are necessary to obtain a desired line-efficiency. The work considers tandem lines with non-exponential stations and extends prior work on tandem lines with exponential servers. A fairly detailed simulation study is conducted to analyze the performance of the tandem lines. The results are used to derive an empirical law that provides an upper bound on the desired buffer levels.

In Chapter 3, Helber uses decomposition to analyze flow lines with Cox-2 distributed processing times and limited buffer capacity. First, he derives an exact solution for a two-station line. Based on this solution, he then derives an approximate, decomposition-based solution for larger flow lines. Finally, he compares the

results obtained by his decomposition method against those obtained by Buzacott, Liu, and Shanthikumar.

In Chapter 4, Colledani, Matta, and Tolio present a decomposition method to evaluate the performance of a production line with multiple failure modes and multiple products. They solve analytically the two-part-type, two-machine line and derive the decomposition equations for longer lines. They use an algorithm similar to the DDX algorithm to solve these equations to determine the production rate and other performance measures approximately.

In the next chapter (Chapter 5), Matta, Runchina, and Tolio address the question of how to increase the production rate of production lines by using a shared buffer within the system in order to avoid blocking. Simulation is used to demonstrate the gain in the mean production rate when a common buffer is used. In addition, an application of the shared buffer approach to a real case is reported.

In Chapter 6, Kim and Gershwin ask what happens if machines in a production line can either fail catastrophically (stop producing), or fail to produce good parts while continuing to produce. First, they develop a Markov process model for machines with both quality and operational failures. Then, they develop models for two-machine systems, for which they calculate total production rate, effective production rate, and yield. Using these models, they conduct numerical studies on the effect of the buffer sizes on the effective production rate.

Finally, in Chapter 7, Lee and Lee consider a flow line with finite buffers that repetitively produces multiple items in a cyclic order. They develop an exact method for evaluating the performance of a two-station line with exponentially or phase-type distributed processing times by making use of the matrix geometric structure of the associated Markov chain. They then present a decomposition-based approximation method for evaluating larger lines. They report on the accuracy of their proposed method and they discuss the effects of job variation and job sequence on performance.

Section III includes Chapters 8–13 on queueing network models of manufacturing systems.

More specifically, in Chapter 8, Van Vuuren, Adan, and Resing-Sassen consider multi-server tandem queues with finite buffers and generally distributed service times. They develop an effective approximation technique based on a spectral expansion method. Numerous experiments are utilized to demonstrate the effectiveness of their performance methodology when compared with simulation of the same systems. Their approximation methodology should be very useful for production line design.

In Chapter 9, Koukoumialos and Liberopoulos present an analytical approximation method for the performance evaluation of multi-stage, serial systems operating under nested or echelon kanban control. Full decomposition is utilized along with an associated set of algorithms to effectively analyze the performance of these systems. Finally, these approximation algorithms are utilized to accurately optimize the design parameters of the system.

In the next chapter (Chapter 10), Spanjers, van Ommeren, and Zijm consider closed-loop, two-echelon repairable item systems with repair facilities at a number of local service centers and at a central location. They use an approximation method

based on a general multi-class marginal distribution analysis algorithm to evaluate the performance of the system. The performance evaluation results are then used to find the stock levels that maximize the availability given a fixed configuration of machines and servers and a certain budget for storing items.

In Chapter 11, Van Nyen, Bertrand, van Ooijen, and Vandaele present a heuristic that minimizes the relevant costs and satisfies the customer service levels in multi-product, multi-machine production-inventory systems characterized by job-shop routings and stochastic arrival, set-up, and processing times. The numerical results derived from the heuristic are compared against simulation.

In Chapter 12, Van Houtum, Adan, Wessels, and Zijm study a production system consisting of several parallel machines, where each machine has its own queue and can produce a particular set of job types. When a job arrives to the system, it joins the shortest queue among all queues capable of serving that job. Under the assumption of Poisson arrivals and identical exponential processing times they derive upper and lower bounds for the mean waiting time and investigate how the mean waiting time is effected by the number of common job types that can be produced by different machines.

Finally, in Chapter 13, Geraghty and Heavey review two approaches that have been followed in the literature for overcoming the disadvantages of kanban control in non-repetitive manufacturing environments. The first approach has been concerned with developing new, or combining existing, pull control strategies and the second approach has focused on combining JIT and MRP. A comparison between a Production Control Strategy (PCS) from each approach is presented. Also, a comparison of the performance of several pull production control strategies in an environment with low variability and a light-to-medium demand load is carried out.

The last section (Section IV) includes Chapters 14 and 15 on production planning and assembly.

In Chapter 14, Axsäter considers a multi-stage assembly network, where a number of end items must be delivered at certain due dates. The operation times at all stages are independent stochastic variables. The objective is to choose starting times for different operations in order to minimize the total expected holding and back-order costs. An approximate decomposition technique, which is based on repeated application of the solution of a simpler single-stage problem, is proposed. The performance of the approximate technique is compared to exact results in a numerical study.

In Chapter 15, Yıldırım, Tan, and Karaesmen study a stochastic, multi-period production planning and sourcing problem of a manufacturer with a number of plants and subcontractors with different costs, lead times, and capacities. The demand for each product in each period is random. They present a methodology for deciding how much, when, and where to produce, and how much inventory to carry, given certain service level constraints. The randomness in demand and related probabilistic service level constraints are integrated in a deterministic mathematical program by adding a number of additional linear constraints. They evaluate the performance of their methodology analytically and numerically.

This volume is a reprint of a special issue of OR Spectrum (Vol. 27, Nos. 2–3) on stochastic models for the design, coordination, and control of

manufacturing systems, with the addition of Chapters 7 and 12 that appeared as articles in other issues of OR Spectrum. That special issue of OR Spectrum originated from the 4th Aegean International Conference on Analysis of Manufacturing Systems, which was held in Samos Island, Greece, in July 1–4 2003. The purpose of that issue was not to simply publish the proceedings of the conference. Rather it was to put together a select set of rigorously refereed articles, each focusing on a novel topic. Collected into a single issue the articles aimed to serve as a useful reference for manufacturing systems researchers and practitioners, and as reading materials for graduate courses and seminars.

We wish to thank Professor Dr. Hans-Otto Guenther, Managing Editor of OR Spectrum, and his staff for supporting the special issue of OR Spectrum and seeing that it becomes a published reality as well as for supporting its subsequent reprint into this volume with the addition of Chapters 7 and 12.

G. Liberopoulos, University of Thessaly, Greece
C. T. Papadopoulos, Aristotle University of Thessaloniki, Greece
B. Tan, Koç University, Turkey
J. M. Smith, University of Massachusetts, USA
S. B. Gershwin, Massachusetts Institute of Technology, USA

Contents

Section I: Factory Design

Dilemmas in factory design: paradox and paradigm
J. MacGregor Smith 3

Section II: Unreliable Production Lines

Lean buffering in serial production lines with non-exponential machines
Emre Enginarlar, Jingshan Li and *Semyon M. Meerkov* 29

Analysis of flow lines with Cox-2-distributed processing times
and limited buffer capacity
Stefan Helber 55

Performance evaluation of production lines with finite buffer capacity
producing two different products
M. Colledani, A. Matta and *T. Tolio* 77

Automated flow lines with shared buffer
A. Matta, M. Runchina and *T. Tolio* 99

Integrated quality and quantity modeling of a production line
Jongyoon Kim and *Stanley B. Gershwin* 121

Stochastic cyclic flow lines with blocking: Markovian models
Young-Doo Lee and *Tae-Eog Lee* 149

Section III: Queueing Network Models of Manufacturing Systems

Performance analysis of multi-server tandem queues
with finite buffers and blocking
Marcel van Vuuren, Ivo J. B. F. Adan and *Simone A. E. Resing-Sassen* 169

An analytical method for the performance evaluation
of echelon kanban control systems
Stelios Koukoumialos and *George Liberopoulos* 193

Closed loop two-echelon repairable item systems
L. Spanjers, J. C. W. van Ommeren and *W. H. M. Zijm* 223

A heuristic to control integrated multi-product multi-machine production-inventory systems with job shop routings and stochastic arrival, set-up and processing times
P. L. M. van Nyen, J. W. M. Bertrand, H. P. G. van Ooijen and *N. J. Vandaele* . . . 253

Performance analysis of parallel identical machines with a generalized shortest queue arrival mechanism
G. J. Van Houtum, I. J. B. E. Adan, J. Wessels and *W. H. M. Zijm* 289

A review and comparison of hybrid and pull-type production control strategies
John Geraghty and *Cathal Heavey* 307

Section IV: Stochastic Production Planning and Assembly

Planning order releases for an assembly system with random operation times
Sven Axsäter 333

A multiperiod stochastic production planning and sourcing problem with service level constraints
Işıl Yıldırım, Barış Tan and *Fikri Karaesmen* 345

Section I: Factory Design

Dilemmas in factory design: paradox and paradigm*

J. MacGregor Smith

Department of Mechanical and Industrial Engineering, University of Massachusetts, Amherst, MA 01003, USA (e-mail: jmsmith@ecs.umass.edu)

Abstract. The problems of factory design are notorious for their complexity. It is argued in this paper that factory design problems represent a class of problems for which there are crucial dilemmas and correspondingly deep-seated underlying paradoxes. These paradoxes, however, give rise to novel paradigms which can bring about fresh approaches as well as insights into their solution.

Keywords: Factory design – Dilemmas – Paradox – Paradigm

1 Introduction

The purpose of this paper is to develop a new paradigm for factory design that integrates much of the theoretical underpinnings of the problems and processes encountered in the author's experiences with factory design. As a side benefit to this paper, many of the ideas discussed within point towards a new direction for which manufacturing and industrial engineering professionals might re-align themselves, since the paradigms which have guided these fields are in need of a new vision and repair.

1.1 Motivation

The origins of this paper stem from an invitation to give a keynote address at a conference on the **Analysis of Manufacturing Systems** [1] where the idea of the

* I would like to thank the referees for their insights and suggestions and pointing out some problems in earlier drafts. My approach to factory design has evolved over the years, and is still evolving, and it is largely due to the influence of Professor Horst Rittel, my professor at the University of California at Berkeley during my formative undergraduate days, who instilled much of the basis of this philosophy.

[1] 4th Aegean Conference on: "The Analysis of Manufacturing Systems", Samos Island Greece, July 1st-July 4th, 2003

address was to recount the author's philosophy about manufacturing systems design and in particular an approach to factory design problems.

Concurrently with the conference there appeared a related conundrum on the email listserv: *iefac.list@deming.ces.clemson.edu* of the Industrial Engineering faculty about an "identity" crisis within the industrial engineering community and the direction of the profession and more practically speaking what fundamental courses should be taught students of industrial engineering. It is not the first time this identity crisis has arisen in IE, nor is the crisis one exclusive to industrial engineers, as it commonly occurs throughout most professions from time-to-time. Paradoxically, all professions have a vested interest in their clients, but cannot be trusted to act in their clients best interests, "a conspiracy against the laity."[21,17].

Since, the Factory Design Problem (FDP) is a very important aspect within manufacturing and industrial engineering, it became obvious that the subject matter of the keynote address and the crisis in industrial engineering education are two closely related matters. So while not attempting to be presumptuous, the resulting paper was a response partly to this crisis and also more importantly to demonstrate the author's philosophy about factory design. The viewpoint and conclusions in the paper may also apply to the problems of factory planning and control, but the focus for the present paper is on the FDP problem.

1.2 Outline of paper

Section 2 of this paper provides necessary background, definitions, and notation on the problem of factory design. Section 3 describes a case study used to illustrate many of the ideas within the paper, while Section 4 provides the theoretical background of the many concepts in the paper. Section 5 describes the implication for the manufacturing and IE profession and Section 6 concludes the paper.

2 Background

Many manufacturing and industrial engineering professionals view the FDP as a complex queueing network, where one has to manufacture or produce a series of products $(1, 2, \ldots, n)$ from different raw materials and possible sources. The average arrival rate of type j raw material from source k is defined as $\lambda_{jk}(j = 1, 2, \ldots, J; k = 1, 2, \ldots, K)$. People, machines, manufacturing processes and the material handling system are necessary to transform the raw materials into finished goods for shipment to consumers at throughput rates $\theta_1, \theta_2, \ldots \theta_n$. Figure 1 is a

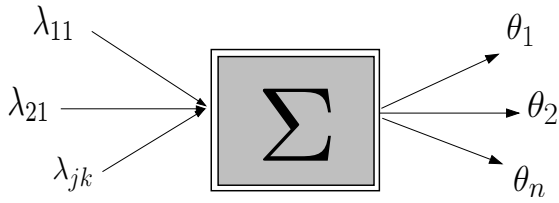


Fig. 1. Factory flow design paradigm

useful caricature of the flow paradigm. The Σ represents the mathematical model of the queueing network underlying the people, resources, products and their flow relationships.

The professionals (especially the academics) would like to know the set of underlying equations Σ (no questions asked) which would allow them to design the factory to maximize the overall throughput (Θ) of the products and also minimize the work-in-process (WIP) inside the plant.

The desire to find all these equations, or laws [9] as some people would like to characterize them, is largely attributed to the scientific foundation of Industrial Engineering education with a strong physics, chemistry, and mathematics background. A sterling example of one of these laws is Little's Law $L = \lambda W$ which is an extremely robust, effective tool to calculate numbers of machines, throughput, and waiting times in queueing processes[9]. What will be shown in the following is that this scientific approach is deficient. The problems of factory design cannot be answered with just a scientific background, but need to be augmented with other knowledge-based skills. The scientific background is necessary but not sufficient to solve the problem.

In order to realize this factory flow paradigm, most IE professionals systematically define the multiple products (there can be hundreds) and their input rates and raw material requirements, the constraint relationships with the machines, people, resources, and materials handling equipment, and the functional equations for achieving the WIP and throughput objectives, utilization, cycle time, lateness, etc. This factory flow paradigm is often realized as a series of well-defined steps or phases similar to the following top-down approach (see Fig. 2).

This top-down approach is also a hallmark of an operations research (OR) paradigm typically argued for in OR textbooks found in the Industrial Engineering curriculum. While this top-down ("waterfall") [3] paradigm has its merits, mainly for project management, it will be argued in this paper that other paradigms are warranted, ones more realistically appropriate for treating FDPs. A key criticism of the top-down approach is that no feedback loops occur at the detailed stages, which is clearly unrealistic. A bottom-up approach, on the other hand, is really not much better, since one has no real overall knowledge of what is being constructed. One needs a paradigm that is paradoxically top-down and bottom up at the same time. Unfortunately, very few individuals are capable of this prescient feat, thus necessitating development of new external aids.

It will also be argued later on in this paper, that the recommended paradigm has strong implications for changes in the profession and in the education of manufacturing and industrial engineers.

2.1 Definitions

Before we proceed too far along, it would be good to posit some of the key definitions and notation utilized throughout the paper [6].

Dilemma: (Late Greek) *dilEmmat, dilEmmatos*- an argument presenting two or more conclusive alternatives against an opponent; a problem involving a difficult choice; a perplexing predicament.

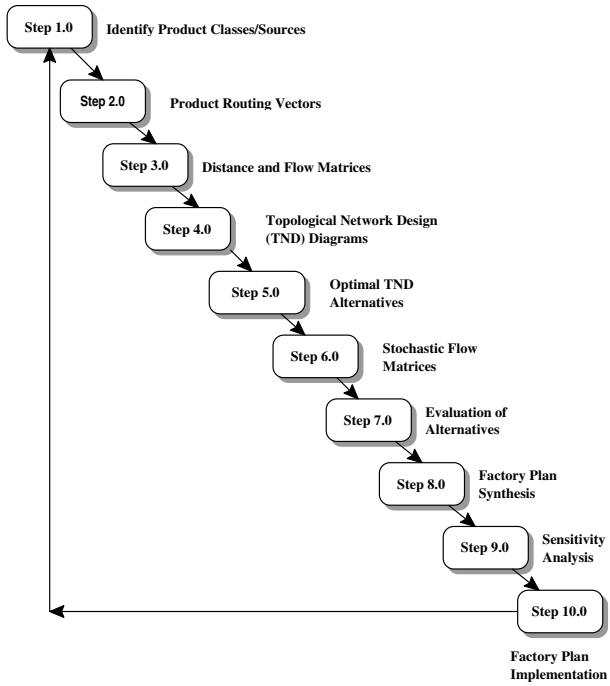


Fig. 2. Factory design process paradigm

Paradox: (Greek) *paradoxon, paradoxos*- A tenet contrary to received opinion. A statement that is seemingly contradictory or opposed to common sense.
Paradigm: (Greek) *paradeigma, paradeiknynai*- To show side by side a pattern- an outstandingly clear example or archetype (*a.k.a.* a philosophy)

The notion of a dilemma in Factory Design is that we are often faced with difficult issues of what to do, and, occasionally, we must select between two alternatives that are not necessarily desirable.

The notion of paradox is important because it helps frame the seemingly contradictory elements which are contrary to common sense.

Dilemmas give rise to paradoxes which in turn underly paradigms for solution. Paradigm is a particularly appropriate word when one thinks of it as a "pattern", since this is often what we employ in resolving design problems because of its modular structure.

All three of these concepts are crucial underpinnings to what is to follow and they form the basis of the general design "philosophy" purported in this paper. The fact that these three concepts are derived from the Greek philosophers is an indication of their importance.

2.2 Notation

The following notation shall be utilized to aid the discussion:

- Δ:= Dilemma
- χ:= Paradox
- $\delta_{\mathbf{i}}$:= Deontic issue
- $\epsilon_{\mathbf{i}}$:= Causal or explanatory issue
- $\iota_{\mathbf{i}}$:= Instrumental issue
- $\phi_{\mathbf{i}}$:= Factual issue
- $\pi_{\mathbf{i}}$:= Planning Issue
- **FDP**:= Factory Design Problem
- **WP**:= Wicked Problem
- **TP**:= Tame Problem
- **IBIS**:= Issue Based Information System
- **NI**:= Non-Inferior set of solutions

3 Case study: polymer recycling project

In order to place things in perspective, a case study will be utilized to characterize the ideas and concepts of the paper. One project completed eight years ago stands out as a compelling example of the ideas in this paper. It was concerned with the FDP of a polymer re-processing plant in Western, Massachusetts.

3.1 Problem description

Essentially, this plant represented a manufacturing/warehouse capacity design problem. The plant maintains a dynamic material handling system which operates 3 shifts 24 hours a day.

The problem as first posed to the factory design team largely revolved around space capacity and equipment needs since the business was growing and there was some real concern about the ability of the present site to accommodate future growth of the business. The business is largely concerned with manufacturing essentially four different polymer products *PC, PC/ABS, PS, ABS* and their combinations. In fact, the unit load of the plant is 1000# gaylords (raw materials and finished goods) filled with various plastic pellets. As will unfold, forecasting the ability of the plant to respond to fluctuations in demand over time also became a critical part of the study.

3.2 Links to paper

Figure 3 illustrates the initial layout of the plant that formed the basis of the layout and systems model about to be discussed. One can see the $4' \times 4'$ gaylords spread throughout the facility in Figure 3.

As one can see in the plant, there is little room for expansion and there is a restricted material handling system where the forklift traffic coming and going must traverse the same aisles.

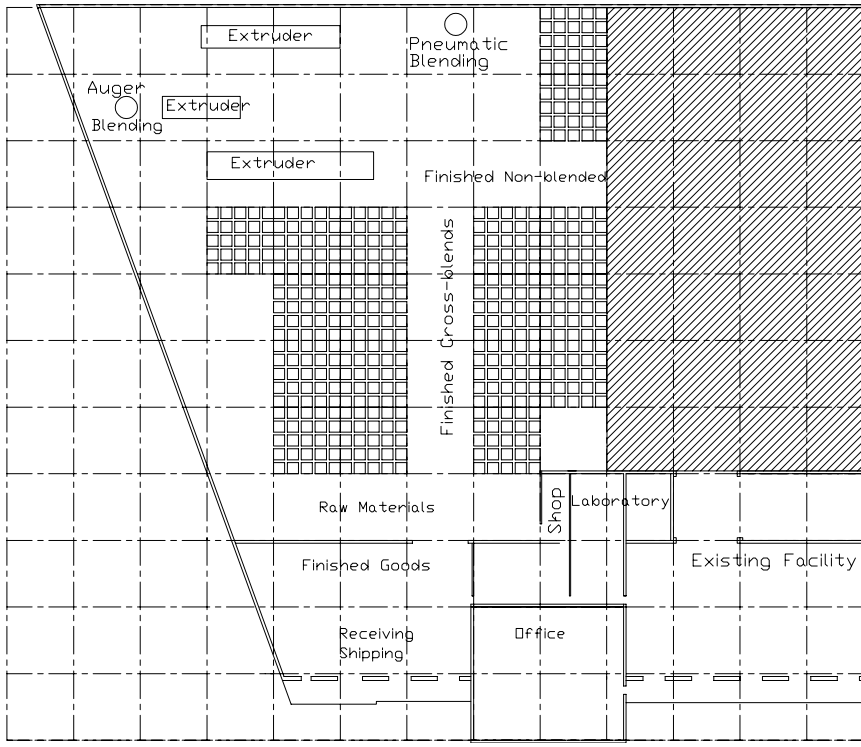


Fig. 3. Existing polymer re-processing plant

4 Dilemmas in factory design

The notion of the dilemmas in factory design stems from a seminal paper of Horst Rittel and Mel Webber [17] on wicked problems. They outline the characteristics of wicked problems and go on to recount how many planning problems are actually wicked problems. In fact they argue that there are essentially two classes of problems:

- *Tame Problems (TPs)*
- *Wicked Problems (WPs)*

Tame problems are like puzzles: precisely described, with a finite (or countably infinite) set of solutions, although perhaps extremely difficult to solve. Problems solved via numerical and combinatorial algorithms can be grouped in this category. The relationship of Computational Complexity and its classes $\mathcal{P}, \mathcal{NP}, \mathcal{NP}-$Complete, and $\mathcal{NP}-$Hard are very appropriate characterizations for tame problems. Also, more recently, designing large scale interacting systems has been shown to be $\mathcal{NP}$- *complete* [5].

It will be argued that the $\mathcal{NP}$ Complexity classification is a useful way of characterizing TPs. On the other hand, Wicked problems are the exact opposite of tame problems, and while not "evil" in themselves, present particulary nasty characteristics which Rittel and Webber feel justly to deserve the approbation. Their wicked

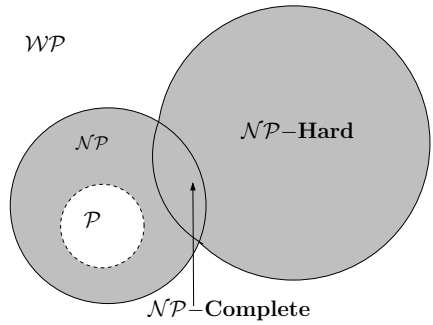


Fig. 4. Wicked problem tame problem dichotomy

problem framework is useful for characterizing the FDP, since the characteristics of FDPs as shall be argued are similar. Not all IEs or manufacturing engineers might agree with the equivalence statement, but the equivalence framework, as we shall argue, will become the basis for the new paradigm.

Very often, IEs utilize algorithmic approaches to solve FDPs, so they become integral parts of the solution process of factory design problems, but a key question here is: *Can we utilize systematic procedures to solve FDPs?*

While no formal classification of WPs has been developed so far, other than what is depicted in Figure 4, it appears that the distinction between one type of wicked problem and another can be based on the following three measurable dimensions:

- x:= # Stakeholders (# persons concerned, involved and affected by the problem)
- y:= # Objectives in the problem $\{f_1, f_2, \ldots f_p\}$
- z:= Time frame or planning horizon (in years)

The degree of "wickedness" is correlated with the cardinality of the dimensions. For example, establishing the solution for the disposal of nuclear waste is one of the most difficult WPs, since the time frame is thousands of years, and the consequences affect millions of people. The reason for selecting these problem dimensions should become clearer as the paper unfolds.

Project management is a classic example of a WP. We know that minimizing the number of dummy activities in a PERT/CPM diagram is actually $\mathcal{NP}$-Complete [12], however, the complexity of balancing time, cost, and quality tradeoffs in scheduling the construction and launching for example of the space shuttle is a very wicked problem. Tame Problems and their solutions are often subsets of WPs and they have their usefulness especially in providing arguments to convince people one way or another on resolving a planning issue, but the TPs are in another class compared to WPs.

Many other researchers have begun to realize the importance and extent of wicked problems in other professions besides factory design. Some of the literature on wicked problems is related to public service facility planning [22], government resource planning within developing countries [19] software engineering design projects [3], planning and project scheduling[20].

Unlike TPs, the first characteristic of a wicked problem is that:

Δ_1:There is no definitive problem formulation.

The dilemma argues that factory design problems cannot be written down on a sheet of paper (like a quadratic equation), given to someone, where they then can go off into a corner and work out the solution. Students are continually drilled with textbook problems (the author is guilty of this himself), but these are not the real problems. Recent research on the modularization of design problems has shown that modularization avoids trade-offs in decision making and often ignores important interactions between decision choices [5].

If someone states the problem as: "build a new plant" or "remodel the existing facility", or "add another storey", then, *i.e.* the solution and problem are one and the same! This is antithetical to the scientific paradigm. In fact, the entire edifice of NP-Completeness problems (*i.e.* Tame Problems) is critically structured around the precise problem definition *e.g.* 3-satisfiability.

For FDPs, it is important whom you talk with and their worldview because in the ensuing dialog the solution to the problem and the problem definition will emerge.

In the case of the polymer recycling plant, when the facility was first examined, their receiving and shipping areas were co-located in the same area of the plant, see the lower left hand corner of Figure 3 which resulted in severe material handling conflicts with forklift truck movements, accidents, and space utilization problems. It was obvious that separate receiving and shipping areas were desirable– thus, the problem was the same as the solution: "re-layout the plant and separate receiving and shipping."

Thus, we have the first formal paradox: $\chi_1 :=$ *Every formulation of a problem corresponds to a statement of its solution and vice versa[14].*

This first dilemma of factory design is a most difficult one. One cannot know *a priori* the problems inherent in factory design, independent of the client and the context around which the problem occurs. In essence, the factory design process is essentially information deficient.

Many "experts" in manufacturing and IE purport to know the answers, yet one must talk with the owners, the plant manager, the line staff, and many others involved with the facility, before the problems and their solutions can be identified. As the paper proceeds, we will postulate the underlying principles of the new paradigm as Propositions. In fact, the principle underlying the paradigm associated with this first dilemma and paradox is:

Proposition 1. *The FDP design system* $\equiv$ *Knowledge/Information System.*

What is meant here by an knowledge/information system? The knowledge/information system here is a special type of information system, not just a sophisticated data base system, where one collects data for the sake of collecting data, but data is collected to resolve the planning issues. The planning issues are the fundamental units within the information system [13]. A related information system approach based on the first proposition is that of Peter Checkland's work [1], however, the information system and resulting paradigm discussed in this paper is based upon different concepts and is directly related to the FDP.

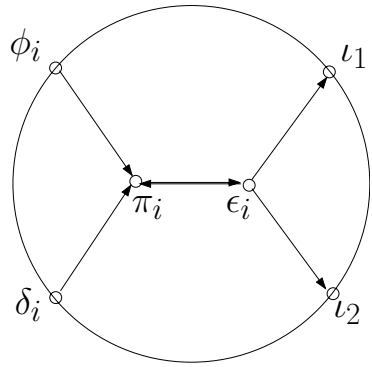


Fig. 5. Planning issue π_i

What are the building blocks of this knowledge/information system? There are essentially four categories of knowledge (issues) needed to help formulate the problem. These fundamental categories of issues are basic to the IBIS[13]:

- **Factual issue** (ϕ_i)**:=** *Knowledge of what is, was, or will be the case.*
- **Deontic issue** (δ_i)**:=** *Knowledge of what ought to be or should be the case.*
- **Explanatory issue** (ϵ_i)**:=** *Knowledge of why something is the case.*
- **Instrumental issue** (ι_i)**:=** *Knowledge of the conditions and methods under which the problem can be resolved.*

Proposition 2. *A planning issue π_i is a discrepancy between what is the case ϕ_i and what ought to be the case δ_i [15].*

The conflict between ϕ_i and δ_i gives rise to π_i. Deontic knowledge is critical to the problem formulation and might be considered as factory planning principles, or "golden rules."

The explanatory issues ϵ_i describe why the problem occurs and the instrumental issues ι_1, ι_2 describe alternative ways of resolving the π_i. At least two alternative ways of resolving an issue are felt to be important for the problem structure and its completeness. Figure 5 illustrates the relationship between a factual issue, a deontic issue, the explanatory and instrumental issues. Each planning issue should be comprised of these component parts.

The planning issue structure is a useful paradigm itself of the elements of problem formulation. It becomes clear how the component parts of a problem should be defined. It also provides an unambiguous method for defining a problem. Each planning issue is dynamic but also bounded. A brief example of a planning issue is derived from the polymer recycling plant.

- *Factual Issue (ϕ_i):=* The number of accidents and potential conflicts with personnel in the plant at the receiving and shipping areas is excessive.
- *Deontic Issue (δ_i):=* The number of conflicts between plant personnel and forklift trucks should be minimized.
- **Planning Issue** (π_i)**:=** How should congestion between forklift trucks and plant personnel be avoided at the receiving and shipping area?

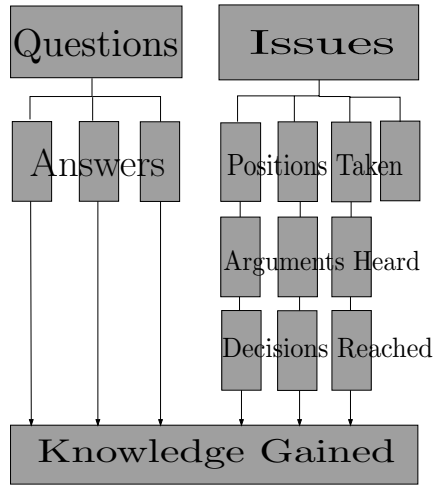


Fig. 6. Planning issues resolution process

– *Explanatory Issue (ϵ_i):=* There is not clear separation between the forklift trucks and the plant personnel within the receiving and shipping area.

  Instrumental Issue (ι_1):= If space is available, separate receiving and shipping and design the material handling systems in the plant in a *U*-shape layout.

  Instrumental Issue (ι_2):= If space is unavailable, clearly demarcate the receiving and shipping areas and the paths of the vehicles and pedestrians.

The reason the above are stated as issues is that evidence for their support must be brought forth to support or refute each issue. People must be convinced of the case being made. Some issues are easily resolved as questions, while others may not be so easily resolved. Not everyone might agree with what we mean by "excessive" traffic in the receiving and shipping area of ϕ_i so some supporting data may be necessary. Likewise, even the instrumental issues will likely need supporting evidence such as is possible with sophisticated simulation and queueing models to estimate expected (maximum) volume of forklift traffic, # number of expected gaylords in the shipping and receiving areas, etc. Why a *U*-shape layout? is certainly arguable. Figure 6 is suggestive of the issue resolution process.

While this approach to problem formulation through the planning issues paradigm can be seen as well-structured, there can be many planning issues in factory design, which, unfortunately, leads to the next dilemma.

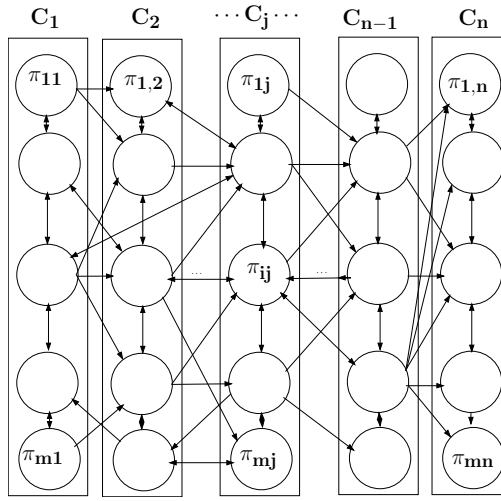


Fig. 7. IBIS dynamic programming paradigm

Δ_2 : Every factory design problem is symptomatic of every other factory design problem.

The second dilemma underscores the fact that there are many problems nested together, there is not simply one isolated problem to be solved. The paradox surrounding the second dilemma is that: χ_2 :=*Tackling the problem as formulated may lead to curing the symptoms of the problem rather than the real problem-you are never sure you are tackling the right problem at the right level.*

One needs to tackle the problems on as high a level as possible. In the polymer recycling project, issues of scheduling, resource configuration and utilization, quality control, and many others became functionally related to the plant layout problem. As will be shown, these other issues emerged as critical to the plant layout. The principle needed in the paradigm in response to the paradox of dilemma #2 is:

Proposition 3. *Construct a network of planning issues, an Issue-Based Information System (IBIS).*

An (IBIS) is needed in order to identify, interrelate, and quantify (weights of importance) the different planning issues within the FDP. Figure 7 illustrates one realization of an IBIS through a dynamic programming (DP) paradigm.

An IBIS has a number of **stages** $C_1, \ldots C_n$ which serve as useful ways of organizing the planning issues as they are defined and emerge in the planning process. Each node within a stage j represents a planning issue π_{ij}. The planning issues represent the **states** of the DP framework. Within each stage C_j all $\pi's$ are inter-connected cliques. There can be many links from one π_{ij} to another π_{ik} so it makes the most sense that the data organization would be some type of relational data base. However, depending upon the problem, other ways of organizing the issues would be possible, such as a simple matrix.

Each C_j represents a stage of the DP paradigm and each state has a set of alternative ways of resolving each planning issue π_{ij} labelled as alternative k within each planning issue x_{ijk} Transitions between states in adjacent stages would have an associated cost for transitioning or linking adjacent states. One possible recursive cost function for an additive or separable resource constrained problem could be [8]:

$$f_j(\pi_i, x_{ijk}) = c_{\pi x_k} + f^*_{j+1}(x_{ijk})$$

In general, the recursive cost function need not be additive, yet the additive situation would be quite appropriate in many resource constrained IBIS scenarios. The general recursive cost function relationship would more likely be:

$$f_j(\pi) = \max_{x_{ijk}} / \min_{x_{ijk}} \{f_j(\pi, x_{ijk})\}$$

One can consider the overall cost of resolving a set of planning issues as a path/tree through the stages and states of the IBIS problem. Each such path represents a morphological plan solution.

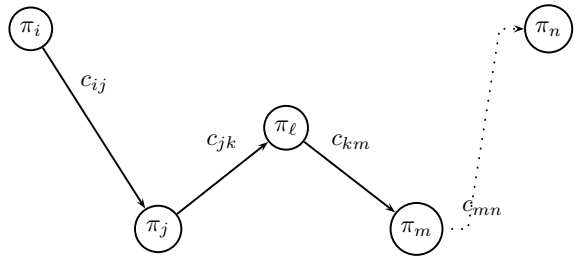


Figure 8 illustrates another IBIS network that was utilized by the author to approach a resource planning problem at the University of Massachusetts [20]. In this study there were five categories (stages) of planning issues (22 issues total):

- C_1 : Client Communication/ Ownership
- I_2 : Information Tracking of Projects
- S_3 : Scheduling and Control of Projects
- G_4 : General Project Management
- O_5 : Outreach to Clients

The IBIS provided a viable framework which resulted in a successful resolution of the management process of small-scale construction projects. In fact, as we speak, this management struggle is still on-going at the University. The planning issues will simply not go away.

The obvious implications for the manufacturing and IE professionals and their education is that the design and analysis of information systems are crucial to the profession. This is in response to dilemmas Δ_1 and Δ_2.

Well, let's argue that these notions of planning issues and information systems are reasonable, what next?

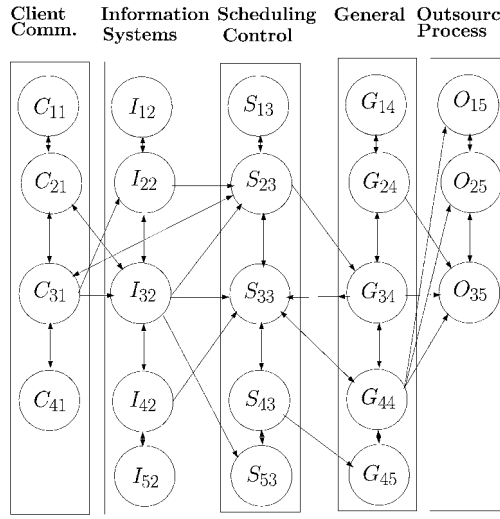


Fig. 8. University of Massachusetts IBIS project

Δ_3 There is no list of permissable operations.

When one plays chess, there are only a finite number of moves to start the game. In linear programming, one needs a starting feasible solution to begin the process. In factory design, there is no one single place to start the problem formulation and solution process.

For the polymer recycling project, we could have visited other polymer processing plants, travelled to other locations besides Western Massachusetts, read all the literature on polymer re-processing, carried out a mail survey, talked with all the employees, and so on. We should have done all the above, but alas, it was not practical nor cost-effective. This dilemma is founded on the following paradox: χ_3 :=*If one is rational, one should consider the consequences of their actions; however, one should also consider the consequences of considering the consequences,* i.e. *if there is nowhere to start to be rational, one should somehow start earlier [15].*

The paradox indicates that a great deal of knowledge about the system under study is needed to assist the client and the engineers in making decisions about the FDP. Of course, a logical response to this paradox is the following principle:

Proposition 4. *Construct a system representation Σ (analytical or simulation) of the manufacturing system within which the FDP is situated.*

This principle is very useful one but obviously can be expensive in time to construct. It makes eminent sense in the supply-chain business environment currently popular, so the more one understands the logistics and the manufacturing systems and processes, the better. At this point, the system model Σ becomes an integral part of the new paradigm.

A discrete-event digital simulation model of the polymer recycling plant was constructed in order to better understand the manufacturing processes and the system as well as the logistics of the product shipments to and from the plant. This

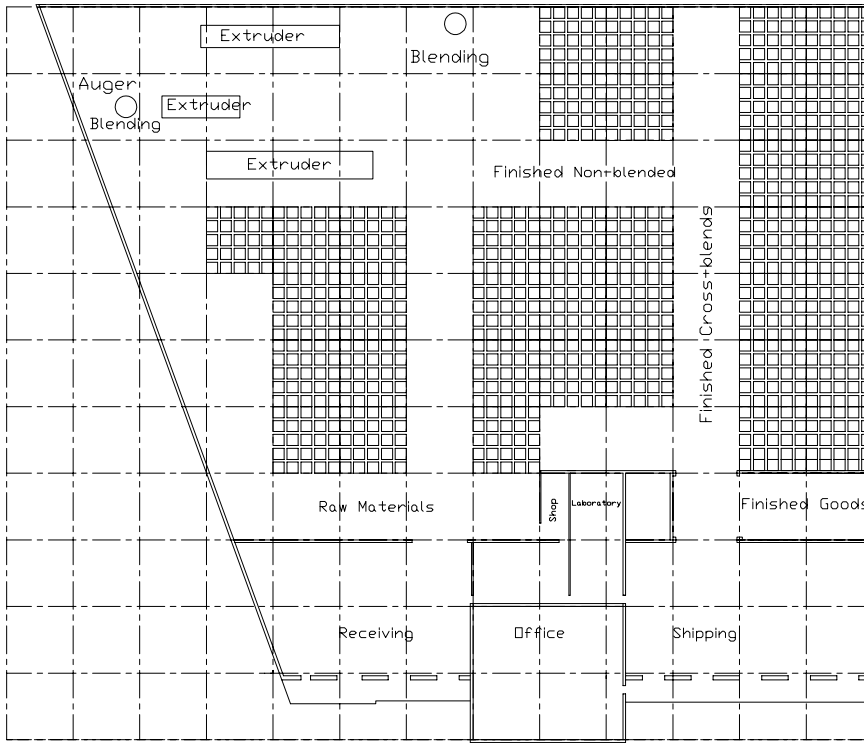


Fig. 9. Final plan for polymer re-processing plant

was felt to be crucial before simply re-laying out the plant and will be shown to be an extremely fortunate decision.

Figure 9 illustrates the layout plan arrived at with a u-shaped circulation flow to eliminate the forklift conflicts from the previous scheme (Fig. 3). Unfortunately, this was not the end of the story.

Thus, for the Manufacturing and IE professional, system models such as supply-chain networks, simulation and queueing network models are critically important to frame the context of the problem. The "systems approach" is still sage advice. Related to Δ_3 is:

Δ_4 : **There is no stopping rule.**

In chess, you either win, lose, or draw– game over! In linear programming, either you find the optimal solution, an unbounded one, or find out that the problem is infeasible. In factory design, you can always make improvements to the system. As we saw above, simply arriving at the layout design in not enough. Thus, we have the following paradox: χ_4 :=*If one is rational, one should also know that every consequence has a consequence, so once one starts to be rational, one cannot stop-one can always do better [15].*

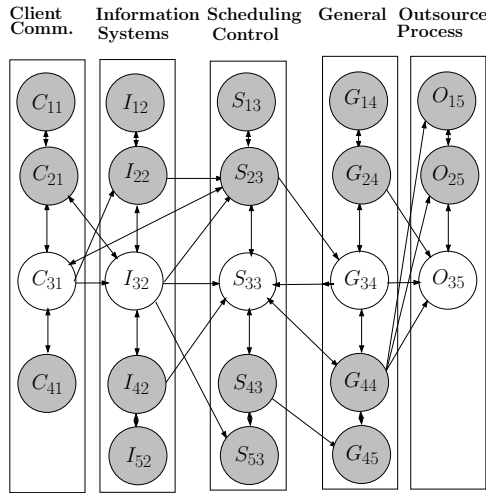


Fig. 10. Path through IBIS network

In fact, another paradox which interrelates Δ_3 and Δ_4 is: χ_5 :=*One cannot start to be rational and, consequently, one cannot stop [15].*

The final step in generating plans for FDPs here is that in factory design and in most wicked problems, time, resources, and the finances involved indicate that one must terminate the design process and arrive at a final plan.

In the context of the IBIS network (see Fig. 10), the highlighted circles illustrate the selected path/plan through the IBIS issues which is actually the path that was taken for the University of Massachusetts project. This path included the following prescient issues which was used to formulate the ultimate strategy (and problem!) for solution:

- C_{31} : There is no customer feedback loop.
- I_{32} : Small construction projects are not as well managed as larger capital projects.
- S_{33} : Cycle times for small construction projects are not satisfactory.
- G_{34} : There is no dedicated professional staff assigned to small projects.
- O_{35} : Outside private contractors (rather than University personnel) do not have access to as-built drawings of University facilities.

Given the resources, time, and financial constraints, this selected path through the IBIS represented a reasonable morphological plan solution.

Also, the remaining issue network does not disappear once the final plan is agreed upon. This is a realistic assessment of the planning process and is also related to the next dilemma.

Δ_5: There are many alternative explanations for a planning issue

As one can argue, there are many explanations for each planning issue, and thus, there are many potential solutions, not just one. Refer to Figure 11 for an illustration of this process.

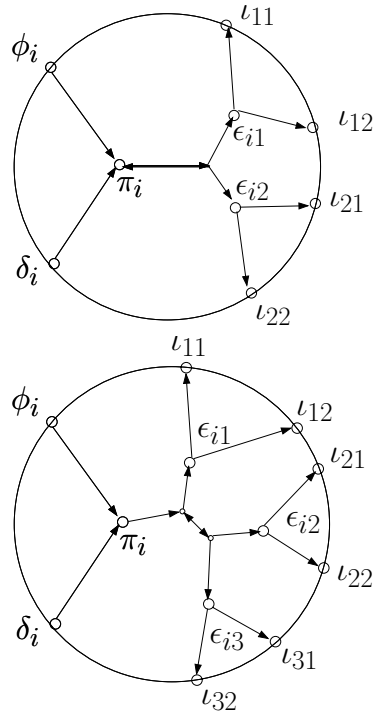


Fig. 11. Many explanations possible for π_i

The paradox surrounding this dilemma: $\chi_{\mathbf{5}}$:=*People need to choose one solution as a "best" solution; but, unfortunately, there are many potential solutions, with correspondingly difficult tradeoffs.*

In response to this situation, one needs much help to generate innovative solutions to the underlying FDP problems. Layout planning algorithms were used in the polymer processing plant to help come to a solution to the layout problem and also were seen as vehicles to resolve issues in the layout problem, not as ends in themselves. Besides using combinatorial optimization algorithms, one needs to generate a special set of solutions, in fact, the paradigmatic principle which $\Delta_{\mathbf{5}}$ is based upon is closely related to the next dilemma both in spirit and in practice.

$\Delta_{\mathbf{6}}$: There is no single criterion for correctness.

In most TPs, there are objective functions which clearly demarcate feasible from optimal solutions. The gap between linear and nonlinear programming TPs can be quite huge. In wicked problems, there are a multiple number of objective functions, not only linear and nonlinear ones. Paradoxically, in factory design we have: $\chi_{\mathbf{6}}$:=*Solutions are either good or bad not right or wrong (true or false). There are multiple criteria embedded within each planning issue.*

Δ_5 and Δ_6 are closely related since one of the reasons why there are so many solutions is that there are multiple objectives in FDP. Thus, we need to generate a Non-Inferior(NI) set of solutions, and the notion of optimality becomes spurious because it only makes sense in a single objective environment. It is very rare that an FDP has only one objective. In another project we worked on, the project manager gave out the following daunting list of objectives before we started our project:

Optimize product flow in order to:

- Reduce project costs;
- Reduce WIP investment;
- Increase Inventory turns;
- Reduce scrap and rework;
- Quicker response to customer needs;
- Improve response time to quality problems;
- Improve housekeeping;
- Better utilize floor space;
- Improve safety;
- Eliminate fork lift trucks.

Thus, we have the following:

Proposition 5. *Generate a Non-inferior (NI) set of Solutions based upon the multiple objectives/criteria* $\{f_1(x), f_2(x), \ldots, f_p(x)\}$ *involved in the FDP.*

The implications of Δ_5 and Δ_6 for the manufacturing and IE profession and curriculum are that multi-criteria and multi-objective programming are essential methodological concepts and algorithmic tools in manufacturing systems and IE. MCDM concepts and methodologies have slowly been introduced into IE curriculums which is a very positive sign. Related to the last dilemma is the fact that:

Δ_7: There is no immediate or ultimate test of a solution

Mathematical programming models, analytical stochastic tools, and simulation models become very important for arguing why resolving a certain issue in a certain way should be carried out. Thus, the systems model suggested in Δ_3 are critical for resolving Δ_7. The paradox here is: χ_7 :=*Unlike chess or solving an equation system, there is no immediate or ultimate test of a solution, because there are dynamic consequences over time, i.e. a great deal of uncertainty.*

Δ_7 is closely related to Δ_4. Both simulation and analytical stochastic and dynamic models are necessary .

Δ_8: Every factory design problem is a one-shot operation.

In factory design problems, one doesn't get a second chance. One can play chess or solitaire many times over. Solving mathematical programming programs on one computer or a distributed computer network is routine. Markovian queueing networks can be run forwards or backwards in time and this affords their decomposability. The paradox is that χ_8 :=*FDPs are not time reversible. There is no*

trial and error with factory design problems, no experimentation you cannot build a plant, tear it down, and rebuild it without significant consequences.

This dilemma and paradox are very troubling because once the factory design project goes to the construction phase, there is no turning back. In many scientific disciplines, repeated experimentation to test an hypothesis is routine and accepted practice because the costs and consequences are justified. The principle relating Δ_7 and Δ_8 is:

Proposition 6. *Dynamic Models* $\Sigma(\mathbf{t})$ *are needed for FDPs.*

For the manufacturing and IE profession, simulation modelling is accepted practice and with good reason. Analytical system models with queueing networks are also becoming more important and many of these analytical tools are often used in addition to simulation.

The polymer recycling project is most appropriate as an illustration of these dilemmas at this stage. In order to test our final factory design layout, we ran the simulation model and calculated the number of gaylords in the warehouse as a function of variations in the input demand $\lambda_i, \forall\ i$, from $0\% - 20\%$. Figure 12 illustrates the results of the simulation runs for the total number of raw material gaylords possible on the y-axis vs. the input demand on the x-axis.

The first 3 columns of Figure 12 illustrate the number of raw material gaylords as a function of input demand. Thus, as one can see the initial design of the plant was fairly robust.

However, Figure 13 revealed that as the input volume ramped up in the plant to 120% (3rd column), a serious problem arose with one of the key resources because at 120% of input demand the minimum raw material input volume went negative by 670 gaylords. Essentially, the plant input-processing of raw materials basically shut-down. We needed to find out which resource was the bottleneck.

After a detailed analysis of the simulation model outputs, it was revealed in the third column of Figure 14 where it is shown that the auger blender was operating at 100% capacity and could not handle any more input. The auger blender was the bottleneck. Thus, if the input demand was to be greater than 20% of the current demand, it became obvious that a minimum of 2 auger blenders were needed as opposed to only one.

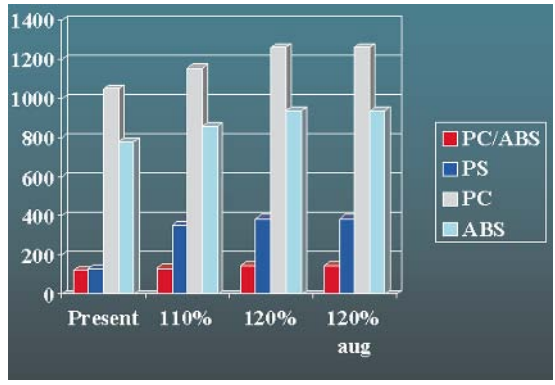


Fig. 12. Total number of raw material gaylords

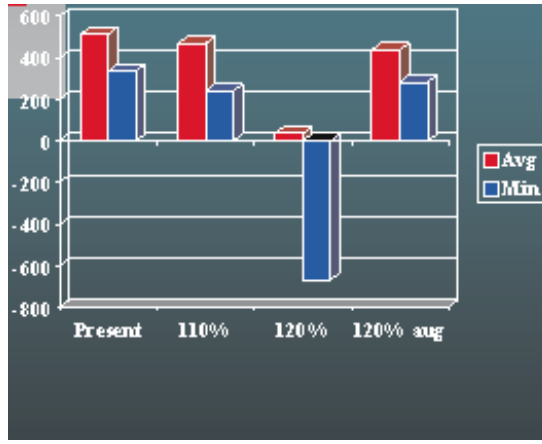


Fig. 13. Average and minimum raw material capacity

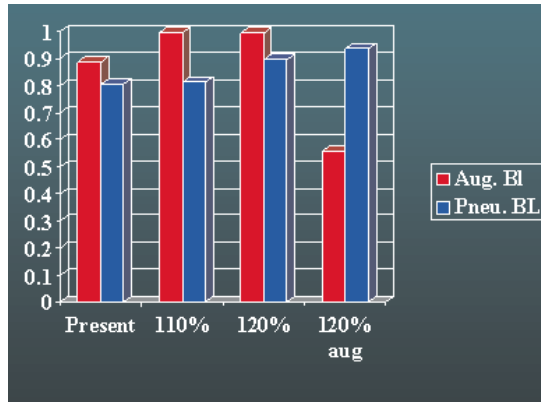


Fig. 14. Blender utilizations vs. input demand

In subsequent runs of the simulation model, 2 auger blenders were utilized, so that in viewing the fourth column in Figures 12,13, and 14, the output statistics include 2 auger blenders operating within the plant. Finally, Figure 15 illustrates that with 2 auger blenders, the total capacity of the plant (# of gaylords including raw materials and finished goods in the revised layout) is acceptable for the given input levels of input demand.

Additional runs of the simulation model revealed that if future input demand were to increase beyond 20%, four extruders rather the current three would be needed to handle the demand. Thus, the simulation model became an invaluable tool to identify the shifting bottlenecks and forecast the configuration of resources needed within the plant as demand increased over time. The next dilemma is very troubling for academics, because it argues that:

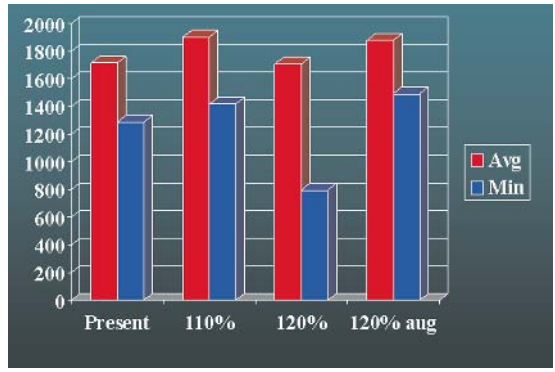


Fig. 15. Final total # of gaylords warehouse capacity

Δ_9: Every factory design problem is unique.

In academia, one learns general principles (deontic knowledge); however, in practice, these general principles must be tempered with the surrounding context, the client, the ever-changing problem requirements, and uncertainty in modelling. With every new FDP, one must start over again. The paradox is that: χ_9 :=*General knowledge and rules are very limited. You cannot learn for the next time. One cannot easily use strategies that have worked in the past and expect that they will work in the future [15].*

Even with all the detailed simulation models and understanding of the plant painstakingly done, when it came to examining the relocation of the polymer processing plant two years after the study, everything had to be re-done all over again, because the site was different, the existing buildings were not the same, the input volume had changed, etc.

Certainly one might argue that experienced people have special knowledge of the issues surrounding a FDP, but there is no guarantee, even if one knows the issues, that the solutions used in the past to resolve them will work in the future.

Proposition 7. *You should never decide too early the nature of the solution and whether or not an old solution can be used in a new context [15].*

Finally, we have the last dilemma:

Δ_{10} := We have no right to be wrong.

This is also very challenging for professionals as well as academics, because the principles of scientific research can be compromised. Science can accept or refute an hypothesis, mathematicians can disprove conjectures, but running a business cannot accept failure. Compromise is essential. The cynical remarks by George Bernard Shaw [21] mentioned at the beginning of this paper underly the moral dilemma captured by this last dilemma. The paradox surrounding this last dilemma is that: χ_{10} :=*Design cannot be carried out in solitary confinement, the FDP design process is democratic.* The final principle summarizes our overall approach to FDP:

Proposition 8. *Solving FDPs is an argumentative and dynamic process concerned with identifying, explaining, and resolving of the planning issues.*

This last principle links back to Δ_1, since the problem formulation process must start with inquiries and issues, and thus an argumentative, dynamic process through an IBIS is critical to the entire FDP process.

5 Implications for the profession and the curriculum

To briefly summarize and emphasize the importance of the preceding discussion, the ten different dilemmas are re-presented below:

Δ_1 : **There is no definitive problem formulation.**
Δ_2 : **Every problem is symptomatic of every other problem.**
Δ_3 : **There is no list of permissable operations.**
Δ_4 : **There is no stopping rule.**
Δ_5 : **There are many alternative solutions for a planning issue.**
Δ_6 : **There is no single criterion for correctness.**
Δ_7 : **There is no immediate or ultimate test of a solution.**
Δ_8 : **Every factory design problem is a one-shot operation.**
Δ_9 : **Every factory design problem is unique.**
Δ_{10} : **We have no right to be wrong.**

The elemental implications for the manufacturing and IE profession are probably best described in a summary implication diagram centered around the dilemmas

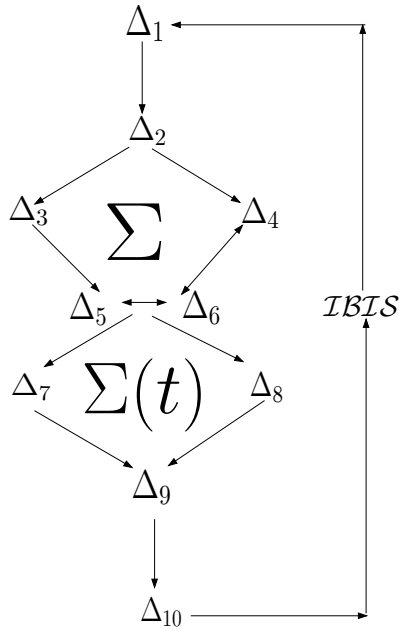


Fig. 16. Final IBIS paradigm

and the IBIS which must integrate them and the models necessary to resolve the issues (see Fig. 16).

The IBIS is necessary to frame Δ_1 and to interrelate the different issues and problems spawned by Δ_2. A systems model Σ is necessary for Δ_3, Δ_4. Generating ideas and evaluating as captured by Δ_5 and Δ_6 must rely on effective algorithmic tools but these must be tempered with a cognizance of the multiple objectives and criteria involved so that effective tradeoffs can be made. A Stochastic/Dynamic model Σ_t is necessary to address the variability, prediction, and control issues surrounding $\Delta_7, \Delta_8, \Delta_9$. Indeed, the degree of uncertainty in most FDPs makes this last stage very challenging. Finally, the IBIS needs to be an open and democratic system that links all aspects of the FDP process.

Perhaps the weakest element in most manufacturing and IE curriculums, at least from the perspectives argued in this paper, is adequate exposure to FDPs as Wicked Problems.

Design problems within academia with real clients are most desirable, whereas, if this is not possible, projects derived from a real world setting with realistic constraints and expectations should be pursued. In a very positive sense, many schools have semester or year-long senior design projects which can capture this aspect of the FDP problem. An interesting development in Engineering education is the Conceiving-Designing-Implementing-Operating real-world systems and products (CDIO) collaborative http://www.cdio.org/ which underscores much of what has been argued here in the is paper. It is oriented to all of Engineering rather than just Industrial and Manufacturing Engineering, but its philosophy is similar. However, it does not appear to rely on an IBIS approach, which as argued for in this paper, is very critical to success in resolving real-world problems.

Problem formulation and structuring for WPs are very difficult topics to treat and teach, but the IBIS framework is something which has clear paradigmatic and teachable elements. Of course, how these elements are put together into the curriculum remains the real wicked problem.

6 Summary and conclusions

The underlying dilemmas, paradoxes, and possible paradigms of factory design have been expounded upon. All these concepts are closely intertwined and it is hoped that illuminating the relationship between these elements will shed some light on possible approaches to FDPs. An IBIS is proposed to be the vehicle for structuring the design process for FDPs. Also, as a side benefit, possible changes to the manufacturing and IE curriculums have been discussed, since FDPs pose a microcosm and synthesis of many of the activities manufacturing and IEs profess.

References

1. Checkland P (1984) Rethinking a systems approach. In: Tomlnson R, Kiss I (eds) Rethinking the process of operational research and systems analysis, pp 43–65. Pergamon Press, New York
2. Cook SA (1971) The complexity of theorem-proving procedures. Proc. of the Third ACM Symposium on Theory of Computing, pp 4–18
3. DeGrace P, Stahl L (1990) Wicked problems, righteous solutions. Yourden Press Computing Series, Upper Saddle River, NJ
4. Dixon JR, Poli C (1995) Engineering design and design for manufacturing. Field Stone, Conway, MA
5. Ethiraj SK, Levinthal D (2004) Modularity and innovation in complex systems. Management Science 50(2): 159–173
6. Funk, Wagnalis (1968) Standard college dictionary. Harcourt, Brace and World, New York
7. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, San Francisco
8. Hillier F, Lieberman G (2001) Introduction to operations research. McGraw-Hill, New York
9. Hopp W, Spearman M (1996) Factory physics. McGraw-Hill, New York
10. Karp RM (1972) Reducibility among combinatorial problems. In: Miller RE, Thatcher JW (eds) Complexity of computer computations. Plenum Press, New York
11. Karp RM (1975) On the computational complexity of combinatorial problems. Networks 5: 45–68
12. Krishnamoorthy MS, Deo N (1979) Complexity of minimum-dummy-activities problem in a PERT network. Networks 9: 189–194
13. Kunz W, Rittel H (1970) Issues as elements of information systems. Institute for Urban and Regional Development, Univeristy of California, Berkeley, Working Paper No. 131
14. Rittel H (1968) Lecture notes for arch, vol 130. Unpublished, Department of Architecture, University of California, Berkeley, CA
15. Rittel H (1972a) On the planning crisis: systems analysis of the 'first and second generations'. Bedrifts Okonomen 8: 390–396
16. Rittel H (1972b) Structure and usefulness of planning information systems. Bedrifts Okonomen (8): 398–401
17. Rittel H, Webber M (1973) Dilemmas in a general theory of planning. Policy Sciences 4: 155–167
18. Rittel H (1975) On the planning crisis: systems analysis of the first and second generations. Reprinted from: Bedrifts ØKonmen, No. 8, October 1972; Reprint No. 107, Berkeley, Institute of Urban and Regional Development
19. Roberts N (2001) Coping with wicked problems: the case of Afghanistan. Learning from International Public Management Forum, vol 11B, pp 353–375. Elsevier, Amsterdam
20. Robinson D, et al. (1999) A change proposal for small-scale renovation and construction projects. Project Team Report. Internal University of Massachusetts Report. Campus Committee for Organization Restructuring, University of Massachusetts, Amherst, MA
21. Shaw GB (1946) The doctors dilemma. Penguin, New York
22. Smith J MacGregor, Larson RJ, MacGilvary DF (1976) Trial court facility. National Clearinghouse for Criminal Justice Planning and Architecture, Monograph B5. In: Guidelines for the planning and design of state court programs and facilities. University of Illinois, Champaign, IL

Section II: Unreliable Production Lines

Lean buffering in serial production lines with non-exponential machines

Emre Enginarlar[1], **Jingshan Li**[2], **and Semyon M. Meerkov**[3]

[1] Decision Applications Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA
[2] Manufacturing Systems Research Laboratory, GM Research and Development Center, Warren, MI 48090-9055, USA
[3] Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122, USA (e-mail: smm@eecs.umich.edu)

Abstract. In this paper, lean buffering (i.e., the smallest level of buffering necessary and sufficient to ensure the desired production rate of a manufacturing system) is analyzed for the case of serial lines with machines having Weibull, gamma, and log-normal distributions of up- and downtime. The results obtained show that: (1) the lean level of buffering is not very sensitive to the type of up- and downtime distributions and depends mainly on their coefficients of variation, CV_{up} and CV_{down}; (2) the lean level of buffering is more sensitive to CV_{down} than to CV_{up} but the difference in sensitivities is not too large (typically, within 20%). Based on these observations, an empirical law for calculating the lean level of buffering as a function of machine efficiency, line efficiency, the number of machines in the system, and CV_{up} and CV_{down} is introduced. It leads to a reduction of lean buffering by a factor of up to 4, as compared with that calculated using the exponential assumption. It is conjectured that this empirical law holds for any unimodal distribution of up- and downtime, provided that CV_{up} and CV_{down} are less than 1.

Keywords: Lean production systems – Serial lines – Non-exponential machine reliability model – Coefficients of variation – Empirical law

1 Introduction

1.1 Goal of the study

The smallest buffer capacity, which is necessary and sufficient to achieve the desired throughput of a production system, is referred to as lean buffering. In (Enginarlar et al., 2002, 2003a), the problem of lean buffering was analyzed for the case of

Correspondence to: S.M. Meerkov

serial production lines with exponential machines, i.e., the machines having up- and downtime distributed exponentially. The development was carried out in terms of normalized buffer capacity and production system efficiency. The normalized buffer capacity was introduced as

$$k = \frac{N}{T_{down}}, \tag{1}$$

where N denoted the capacity of each buffer and T_{down} the average downtime of each machine in units of cycle time (i.e., the time necessary to process one part by a machine). Parameter k was referred to as the *Level of Buffering* (LB). The production line efficiency was quantified as

$$E = \frac{PR_k}{PR_\infty}, \tag{2}$$

where PR_k and PR_∞ represented the production rate of the line (i.e., the average number of parts produced by the last machine per cycle time) with LB equal to k and infinity, respectively. The smallest k, which ensured the desired E, was denoted as k_E and referred to as the *Lean Level of Buffering* (LLB).

Using parameterizations (1) and (2), Enginarlar et al., (2002, 2003a) derived *closed* formulas for k_E as a function of system characteristics. For instance, in the case of two-machines lines, it was shown that (Enginarlar et al., 2002)

$$k_E^{exp} = \begin{cases} \dfrac{2e(E-e)}{1-E}, \text{ if } e < E, \\ \\ 0, \qquad\quad \text{otherwise.} \end{cases} \tag{3}$$

Here the superscript exp indicates that the machines have exponentially distributed up- and downtime, and e denotes machine efficiency in isolation, i.e.,

$$e = \frac{T_{up}}{T_{up} + T_{down}}, \tag{4}$$

where T_{up} is the average uptime in units of cycle time. For the case of $M > 2$-machine serial lines, the following formula had been derived (Enginarlar et al., 2003a):

$$k_E^{exp}(M{\geq}3) = \begin{cases} \dfrac{e(1-Q)(eQ+1-e)(eQ+2-2e)(2-Q)}{Q(2e-2eQ+eQ^2+Q-2)} \times \\ \ln\left(\dfrac{E-eE+eEQ-1+e-2eQ+eQ^2+Q}{(1-e-Q+eQ)(E-1)}\right), \quad \text{if } e<E^{\frac{1}{M-1}}, \\ \\ 0, \qquad\qquad\qquad\qquad\qquad\qquad \text{otherwise,} \end{cases} \tag{5}$$

where

$$Q = 1 - E^{\frac{1}{2}\left[1+\left(\frac{M-3}{M-1}\right)^{M/4}\right]} + \left(E^{\frac{1}{2}\left[1+\left(\frac{M-3}{M-1}\right)^{M/4}\right]} - E^{\frac{M-2}{M-1}}\right) \times \exp\left\{-\left(\frac{E^{\frac{1}{M-1}} - e}{1-E}\right)\right\}. \tag{6}$$

This formula is exact for $M = 3$ and approximate for $M > 3$.

Initial results on lean buffering for non-exponential machines have been reported in (Enginarlar et al., 2002). Two distributions of up- and downtime have been considered (Rayleigh and Erlang). It has been shown that LLB for these cases is smaller than that for the exponential case. However, (Enginarlar et al., 2002) did not provide a sufficiently complete characterization of lean buffering in non-exponential production systems. In particular, it did not quantify how different types of up- and downtime distributions affect LLB and did not investigate relative effects of uptime vs. downtime on LLB.

The goal of this paper is to provide a method for selecting LLB in serial lines with non-exponential machines. We consider Weibull, gamma, and log-normal reliability models under various assumptions on their parameters. This allows us to place their coefficients of variations at will and study LLB as a function of up- and downtime variability. Moreover, since each of these distributions is defined by two parameters, selecting them appropriately allows us to analyze the lean buffering for 26 various shapes of density functions, ranging from almost delta-function to almost uniform. This analysis leads to the quantification of both influences of distribution shapes on LLB and effects of up- and downtime on LLB. Based of these results, we develop a method for selecting LLB in serial lines with Weibull, gamma, and log-normal reliability characteristics and conjecture that the same method can be used for selecting LLB in serial lines with arbitrary unimodal distributions of up- and downtime.

1.2 Motivation for considering non-exponential machines

The case of non-exponential machines is important for at least two reasons:

First, in practice the machines often have up- and downtime distributed non-exponentially. As the empirical evidence (Inman, 1999) indicates, the coefficients of variation, CV_{up} and CV_{down} of these random variables are often less than 1; thus, the distributions cannot be exponential. Therefore, an analytical characterization of k_E for non-exponential machines is of theoretical importance.

Second, such a characterization is of practical importance as well. Indeed, it can be expected that k_E^{exp} is the upper bound of k_E for $CV < 1$ and, moreover, k_E might be substantially smaller than k_E^{exp}. This implies that a smaller buffer capacity is necessary to achieve the desired line efficiency E when the machines are non-exponential. Thus, selecting LLB based on realistic, non-exponential reliability characteristics would lead to increased leanness of production systems.

1.3 Difficulties in studying the non-exponential case

Analysis of lean buffering in serial production lines with non-exponential machines is complicated, as compared with the exponential case, by the reasons outlined in Table 1 . Especially damaging is the first one, which practically precludes analytical investigation. The other reasons lead to a combinatorially increasing number of cases to be investigated. In this work, we partially overcome these difficulties by

Table 1. Difficulties of the non-exponential case as compared with the exponential one

Exponential case	Non-exponential case
Analytical methods for evaluating PR are available	No analytical methods for evaluating PR are available
Machine up- and downtimes are distributed identically (i.e., exponentially).	Machine up- and downtimes may have different distributions.
Coefficients of variation of machine up- and downtimes are identical and equal to 1.	Coefficients of variation of machine up- and downtimes may take arbitrary positive values and may be non-identical.
All machines in the system have the same type of up- and downtime distributions (i.e., exponential).	Each machine in the system may have different types of up- and downtime distributions.

using numerical simulations and by restricting the number of distributions and coefficients of variation analyzed.

1.4 Related literature

The majority of quantitative results on buffer capacity allocation in serial production lines address the case of exponential or geometric machines (Buzacott, 1967; Caramanis, 1987; Conway et al., 1988; Smith and Daskalaki, 1988; Jafari and Shanthikumar, 1989; Park, 1993; Seong et al., 1995; Gershwin and Schor, 2000). Just a few numerical/empirical studies are devoted to the non-exponential case. Specifically, two-stage coaxian type completion time distributions are considered by Altiok and Stidham (1983), Chow (1987), Hillier and So (1991a,b), and the effects of log-normal processing times are analyzed by Powell (1994), Powell and Pyke (1998), Harris and Powell (1999). These papers consider lines with reliable machines having random processing time. Another approach is to develop methods to extend the results obtained for such cases to unreliable machines with deterministic processing time (Tempelmeier, 2003). Phase-type distributions to model random processing time and reliability characteristics are analyzed by Altiok (1985, 1989), Altiok and Ranjan (1989), Yamashita and Altiok (1998), but the resulting methods are computationally intensive and can be used only for short lines with small buffers (e.g., two-machine lines with buffers of capacity less than six). Finally, as it was mentioned in the Introduction, initial results on lean level of buffering in serial lines with Rayleigh and Erlang machines have been reported in (Enginarlar et al., 2002).

1.5 Contributions of this paper

The main results derived in this paper are as follows:

- LLB is not very sensitive to the type of up- and downtime distributions and depends mostly on their coefficients of variation (CV_{up} and CV_{down}).
- LLB is more sensitive to CV_{down} than to CV_{up}, but this difference in sensitivities is not too large (typically, within 20%).
- In serial lines with M machines having Weibull, gamma, and log-normal distributions of up- and downtime with CV_{up} and CV_{down} less than 1, LLB can be selected using the following upper bound:

$$k_E(M, E, e, CV_{up}, CV_{down}) \leq \frac{\max\{0.25, CV_{up}\} + \max\{0.25, CV_{down}\}}{2} \; k_E^{exp}(M, E, e), \tag{7}$$

  where k_E^{exp} is given by (5), (6). This bound is referred to as the *empirical law*. It is conjectured that this bound holds for all unimodal up- and downtime distributions with $CV_{up} < 1$ and $CV_{down} < 1$.
- Although for some values of CV_{up} and CV_{down}, bound (7) may not be too tight, it still leads to a reduction of lean buffering by a factor of up to 4, as compared to LLB based on the exponential assumption.

1.6 Paper organization

In Section 2, the model of the production system under consideration is introduced and the problems addressed are formulated. Section 3 describes the approach of this study. Sections 4 and 5 present the main results pertaining, respectively, to systems with machines having identical and non-identical coefficients of variation of up- and downtime. In Section 6, serial lines with machines having arbitrary, i.e., general, reliability models are discussed. Finally, in Section 7, the conclusions are formulated.

2 Model and problem formulation

2.1 Model

The block diagram of the production system considered in this work is shown in Figure 1, where the circles represent the machines and the rectangles are the buffers. Assumptions on the machines and buffers, described below, are similar to those of (Enginarlar et al., 2003a) with the only difference that up- and downtime distributions are not exponential. Specifically, these assumptions are:

(i) Each machine m_i, $i = 1, \ldots, M$, has two states: up and down. When up, the machine is capable of processing one part per cycle time; when down, no production takes place. The cycle times of all machines are the same.

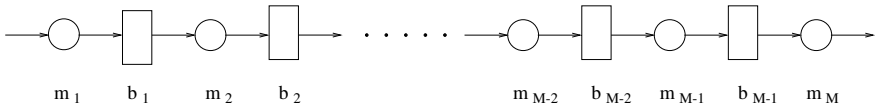


Fig. 1. Serial production line

(ii) The up- and downtime of each machine are random variables measured in units of the cycle time. In other words, uptime (respectively, downtime) of length $t \geq 0$ implies that the machine is up (respectively, down) during t cycle times. The up- and downtime are distributed according to one of the following probability density functions, referred to as *reliability models:*

(a) Weibull, i.e.,

$$f_{up}^{W}(t) = p^P e^{-(pt)^P} P t^{P-1},$$
$$f_{down}^{W}(t) = r^R e^{-(rt)^R} R t^{R-1}, \tag{8}$$

where $f_{up}^{W}(t)$ and $f_{down}^{W}(t)$ are the probability density functions of up- and downtime, respectively and (p, P) and (r, R) are their parameters. (Here, and in the subsequent distributions, the parameters are positive real numbers). These distributions are denoted as $W(p, P)$ and $W(r, R)$, respectively.

(b) Gamma, i.e.,

$$f_{up}^{g}(t) = pe^{-pt}\frac{(pt)^{P-1}}{\Gamma(P)},$$
$$f_{down}^{g}(t) = re^{-rt}\frac{(rt)^{R-1}}{\Gamma(R)}, \tag{9}$$

where $\Gamma(x)$ is the gamma function, $\Gamma(x) = \int_0^\infty s^{x-1}e^{-s}ds$. These distributions are denoted as $g(p, P)$ and $g(r, R)$, respectively.

(c) Log-normal, i.e.,

$$f_{up}^{LN}(t) = \frac{1}{\sqrt{2\pi}Pt}e^{-\frac{(ln(t)-p)^2}{2P^2}},$$
$$f_{down}^{LN}(t) = \frac{1}{\sqrt{2\pi}Rt}e^{-\frac{(ln(t)-r)^2}{2R^2}}. \tag{10}$$

We denote these distributions as $LN(p, P)$ and $LN(r, R)$, respectively.

The expected values, variances, and coefficients of variation of distributions (8)–(10) are given in Table 2.

(iii) The parameters of distributions (8)–(10) are selected so that the machine efficiencies, i.e.,

$$e = \frac{T_{up}}{T_{up} + T_{down}}, \tag{11}$$

and, moreover, T_{up}, T_{down}, CV_{up}, and CV_{down} of all machines are identical for all reliability models, i.e.,

$$T_{up} = p^{-1}\Gamma\left(1 + \frac{1}{P}\right) \qquad \text{(Weibull)}$$

Table 2. Expected value, variance, and coefficient of variation of up- and downtime distributions considered

	Gamma	Weibull	Log-normal
T_{up}	P/p	$p^{-1}\Gamma(1+1/P)$	$e^{p+P^2/2}$
T_{down}	R/r	$r^{-1}\Gamma(1+1/R)$	$e^{r+R^2/2}$
σ^2_{up}	P/p^2	$p^{-2}[\Gamma(1+2/P)-\Gamma^2(1+1/P)]$	$e^{2p+P^2(e^{P^2}-1)}$
σ^2_{down}	R/r^2	$r^{-2}[\Gamma(1+2/R)-\Gamma^2(1+1/R)]$	$e^{2r+R^2(e^{R^2}-1)}$
CV_{up}	$1/\sqrt{P}$	$\sqrt{\Gamma(1+2/P)-\Gamma^2(1+1/P)}\big/\Gamma(1+1/P)$	$\sqrt{e^{P^2}-1}$
CV_{down}	$1/\sqrt{R}$	$\sqrt{\Gamma(1+2/R)-\Gamma^2(1+1/R)}\big/\Gamma(1+1/R)$	$\sqrt{e^{R^2}-1}$

$$
\begin{aligned}
&= \frac{P}{p} \quad \text{(gamma)}\\
&= e^{p+P^2/2} \quad \text{(log-normal);}\\
T_{down} &= r^{-1}\Gamma\left(1+1/R\right) \quad \text{(Weibull)}\\
&= \frac{R}{r} \quad \text{(gamma)}\\
&= e^{r+R^2/2} \quad \text{(log-normal);}
\end{aligned}
$$

$$
\begin{aligned}
CV_{up} &= \frac{\sqrt{\Gamma(1+2/P)-\Gamma^2(1+1/P)}}{\Gamma(1+1/P)} \quad \text{(Weibull)}\\
&= \frac{1}{\sqrt{P}} \quad \text{(gamma)}\\
&= \sqrt{e^{P^2}-1} \quad \text{(log-normal);}\\
CV_{down} &= \frac{\sqrt{\Gamma(1+2/R)-\Gamma^2(1+1/R)}}{\Gamma(1+1/R)} \quad \text{(Weibull)}\\
&= \frac{1}{\sqrt{R}} \quad \text{(gamma)}\\
&= \sqrt{e^{R^2}-1} \quad \text{(log-normal).}
\end{aligned}
$$

(iv) Buffer b_i, $i = 1, \ldots, M-1$ is of capacity $0 \le N \le \infty$.

(v) Machine m_i, $i = 2, \ldots, M$, is starved at time t if it is up at time t, buffer b_{i-1} is empty at time t and m_{i-1} does not place any work in this buffer at time t. Machine m_1 cannot be starved.

(vi) Machine m_i, $i = 1, \ldots, M-1$, is blocked at time t if it is up at time t, buffer b_i is full at time t and m_{i+1} fails to take any work from this buffer at time t. Machine m_M cannot be blocked.

Remark 1.

- Assumptions (i)–(iii) imply that all machines are identical from all points of view except, perhaps, for the nature of up- and downtime distributions. The buffers are also assumed to be of equal capacity (see (iv)). We make these assumptions in order to provide a compact characterization of lean buffering.
- Assumption (ii) implies, in particular, that time-dependent, rather than operation-dependent failures, are considered. This failure mode simplifies the analysis and results in just a small difference in comparison with operation-dependent failures.

2.2 Notations

Each machine considered in this paper is denoted by a pair

$$[D_{up}(p, P), D_{down}(r, R)]_i, \qquad i = 1, \ldots, M, \tag{12}$$

where $D_{up}(p, P)$ and $D_{down}(r, R)$ represent, respectively, the distributions of up- and downtime of the i-th machine in the system, D_{up} and $D_{down} \in \{W, g, LN\}$. The serial line with M machines is denoted as

$$\{[D_{up}, D_{down}]_1, \ldots, [D_{up}, D_{down}]_M\}. \tag{13}$$

If all machines have identical distribution of uptimes and downtimes, the line is denoted as

$$\{[D_{up}(p, P), D_{down}(r, R)]_i, i = 1, \ldots, M\}. \tag{14}$$

If, in addition, the types of up- and downtime distributions are the same, the notation for the line is

$$\{[D(p, P), D(r, R)]_i, i = 1, \ldots, M\}. \tag{15}$$

Finally, if up- and downtime distributions of the machines are not necessarily W, g, or LN but are general in nature, however, unimodal, the line is denoted as

$$\{[G_{up}, G_{down}]_1, \ldots, [G_{up}, G_{down}]_M\}. \tag{16}$$

2.3 Problems addressed

Using the parameterizations (1), (2), the model (i)–(vi), and the notations (12)–(16), this paper is intended to

- develop a method for calculating *Lean Level of Buffering* in production lines (13)–(15) under the assumption that the coefficients of variation of up- and downtime, CV_{up} and CV_{down}, are identical, i.e., $CV_{up} = CV_{down} = CV$;
- develop a method of calculating LLB in production lines (13)–(15) for the case of $CV_{up} \neq CV_{down}$;
- extend the results obtained to production lines (16).

Solutions of these problems are presented in Sections 4–6 while Section 3 describes the approach used in this work.

3 Approach

3.1 General considerations

Since LLB depends on line efficiency E, the calculation of k_E requires the knowledge of the production rate, PR, of the system. Unfortunately, as it was mentioned earlier, no analytical methods exist for evaluating PR in serial lines with either Weibull, or gamma, or log-normal reliability characteristics. Approximation methods are also hardly applicable since, in our experiences, even 1%-2% errors in the production rate evaluation (due to the approximate nature of the techniques) often lead to much larger errors (up to 20%) in lean buffering characterization. Therefore, the only method available is the Monte Carlo approach based on numerical simulations. To implement this approach, a MATLAB code was constructed, which simulated the operation of the production line defined by assumptions (i)–(vi) of Section 2. Then, a set of representative distributions of up- and downtime was selected and, finally, for each member of this set, PR and LLB were evaluated with guaranteed statistical characteristics. Each of these steps is described below in more detail.

3.2 Up- and downtime distributions analyzed

The set of 26 downtime distributions analyzed in this work is shown in Table 3, where the notations introduced in Section 2.1 are used. These distributions are classified according to their coefficients of variation, CV_{down}, which take values from the set $\{0.1, 0.25, 0.5, 0.75, 1.0\}$. The analysis of LLB for this set is intended to reveal the behavior of k_E as a function of CV_{down}.

To investigate the effect of the average downtime, the distributions of Table 3 have been classified according to T_{down}, which takes values 20 and 100.

An illustration of a few of the downtime distributions included in Table 3 is given in Figure 2 for $CV_{down} = 0.5$. As one can see, the shapes of the distributions included in Table 3 range from "almost" uniform to "almost" δ-function.

Table 3. Downtime distributions considered

CV_{down}	$T_{down} = 20$	$T_{down} = 100$
0.1	$g(5, 100)$,	$g(1, 100)$,
	$W(0.048,\ 12.15)$, $LN(2.99, 0.1)$	$W(0.01,\ 12.15)$, $LN(4.602, 0.1)$
0.25	$g(0.8, 16)$,	$g(0.16, 16)$,
	$W(0.046,\ 4.54)$, $LN(2.97, 0.25)$	$W(0.009,\ 4.54)$, $LN(4.57, 0.25)$
0.5	$g(0.2, 4)$,	$g(0.04, 4)$,
	$W(0.044,\ 2.1)$, $LN(2.88, 0.49)$	$W(0.009,\ 2.1)$, $LN(4.49, 0.49)$
0.75	$g(0.09, 1.8)$,	$g(0.018, 1.8)$,
	$W(0.046,\ 1.35)$, $LN(2.77, 0.66)$	$W(0.009,\ 1.35)$, $LN(4.38, 0.66)$
1.00	$LN(2.65, 0.83)$	$LN(4.26, 0.83)$

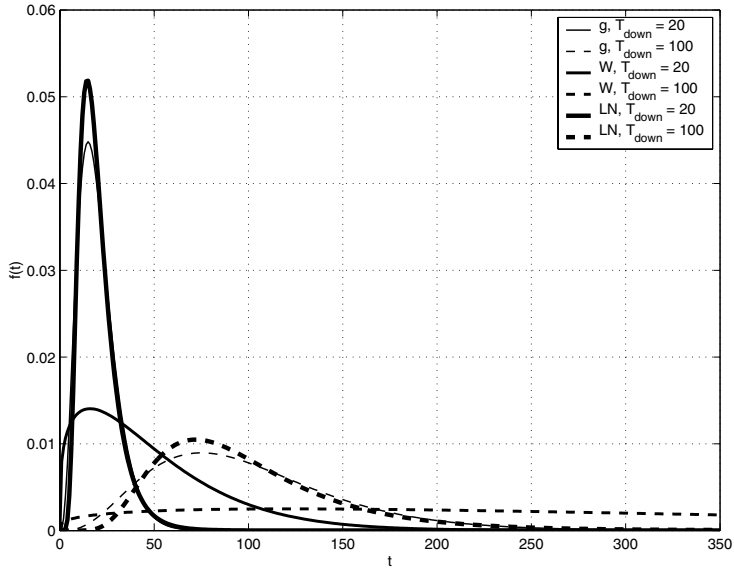


Fig. 2. Different distributions with identical coefficients of variation ($CV_{down} = 0.5$)

The uptime distributions, corresponding to the downtime distributions of Table 3, have been selected as follows: For a given machine efficiency, e, the average uptime was chosen as

$$T_{up} = \frac{e}{1-e} T_{down}.$$

Next, CV_{up} was selected as $CV_{up} = CV_{down}$, when the case of identical coefficients of variation of up- and downtime was considered; otherwise CV_{up} was selected as a constant independent of CV_{down}. Finally, using these T_{up} and CV_{up}, the distribution of uptime was selected to be the same as that of the downtime, if the case of identical distributions was analyzed; otherwise it was selected as any other distribution from the set $\{W, g, LN\}$. For instance, if the downtime was distributed according to $D_{down}(r, R) = g(0.018, 1.8)$ and e was 0.9, the uptime distribution was selected as

$$D_{up}(p, P) = \begin{cases} g(0.002, 1.8) & \text{for} \quad CV_{up} = CV_{down}, \\ g(0.0044, 4) & \text{for} \quad CV_{up} = 0.5, \end{cases}$$

or

$$D_{up}(p, P) = \begin{cases} LN(6.69, 0.47) & \text{for} \quad CV_{up} = CV_{down}, \\ LN(2.88, 0.49) & \text{for} \quad CV_{up} = 0.5. \end{cases}$$

Remark 2. Both CV_{up} and CV_{down} considered are less than 1 because, according to the empirical evidence of (Inman, 1999), the equipment on the factory floor often satisfies this condition. In addition, it has been shown by Li and Meerkov (2005) that CV_{up} and CV_{down} are less than 1 if the breakdown and repair rates of the machines are increasing functions of time, which often takes place in reality.

3.3 Parameters selected

In all systems analyzed, particular values of M, E, and e have been selected as follows:

(a) The number of machines in the system, M: Since, as it was shown in (Enginarlar et al., 2002), k_E^{exp} is not very sensitive to M if $M \geq 10$, the number of machines in the system was selected to be 10. For verification purposes, we analyzed also serial lines with $M = 5$.

(b) Line efficiency, E: In practice, production lines are often operated close to their maximum capacity. Therefore, for the purposes of simulation, E was selected to belong to the set $\{0.85, 0.9, 0.95\}$. For the purposes of verification, additional values of E analyzed were $\{0.7, 0.8\}$.

(c) Machine efficiency, e: Although in practice e may have widely different values (e.g., smaller in machining operations and much larger in assembly), to obtain a manageable set of systems for simulation, e was selected from the set $\{0.85, 0.9, 0.95\}$. For verification purposes, we considered $e \in \{0.6, 0.7, 0.8\}$.

3.4 Systems analyzed

Specific systems of the form (15) considered in this work are:

$$\begin{gathered}\{[W(p,P), W(r,R)]_i, i = 1, \ldots, 10\}, \\ \{[g(p,P), g(r,R)]_i, i = 1, \ldots, 10\}, \\ \{[LN(p,P), LN(r,R)]_i, i = 1, \ldots, 10\}.\end{gathered} \tag{17}$$

Systems of the form (13) have been formed as follows: For each machine m_i, $i = 1, \ldots, 10$, the up- and downtime distributions were chosen from the set $\{W, g, LN\}$ equiprobably and independently of each other and all other machines in the system. As a result, the following two lines were selected:

$$\begin{aligned}\text{Line 1:}\quad & \{(g,W), (LN,LN), (W,g), (g,LN), (g,W), \\ & (LN,g), (W,W), (g,g), (LN,W), (g,LN)\}, \\ \text{Line 2:}\quad & \{(W,LN), (g,W), (LN,W), (W,g), (g,LN), \\ & (g,W), (W,W), (LN,g), (g,W), (LN,LN)\}.\end{aligned} \tag{18}$$

We will use notations $A \in \{(17)\}$, $A \in \{(18)\}$ or $A \in \{(17), (18)\}$ to indicate that line A is one of (17), or one of (18), and one of (17) and (18), respectively.

Lines (17) and (18) are analyzed in Sections 4 and 5 for the cases of $CV_{up} = CV_{down}$ and $CV_{up} \neq CV_{down}$, respectively.

3.5 Evaluation of the production rate

To evaluate the production rate in systems (17) and (18), using the MATLAB code and the up- and downtime distributions discussed in Sections 3.1–3.3, zero initial

conditions of all buffers have been assumed and the states of all machines at the initial time moment have been selected "up". The first 100,000 cycle times were considered as warm-up period. The subsequent 1,000,000 cycle times were used for statistical evaluation of PR. Each simulation was repeated 10 times, which resulted in 95% confidence intervals of less than 0.0005.

3.6 Evaluation of LLB

The lean buffering, k_E, necessary and sufficient to ensure line efficiency E, was evaluated using the following procedure:

For each model of serial line (13)–(15), the production rate was evaluated first for $N = 0$, then for $N = 1$, and so on, until the production rate $PR = E \cdot PR_\infty$ was achieved. Then k_E was determined by dividing the resulting N_E by the machine average downtime (in units of the cycle time).

Remark 3. Although, as it is well known (Hillier and So, 1991b), the optimal allocation of a fixed total buffer capacity is non-uniform, to simplify the analysis we consider only uniform allocations. Since the optimal (i.e., inverted bowl) allocation typically results in just $1 - 2\%$ throughput improvement in comparison with the uniform allocation, for the sake of simplicity we consider only the latter case.

4 LLB in serial lines with $CV_{up} = CV_{down} = CV$

4.1 System $\{[D(p, P), D(r, R)]_i, i = 1, \ldots, 10\}$

Figures 3 and 5 present the simulation results for production lines (17) for all distributions of Table 3. These figures are arranged as matrices where the rows and columns correspond to $e \in \{0.85, 0.9, 0.95\}$ and $E \in \{0.85, 0.9, 0.95\}$, respectively. Since, due to space considerations, the graphs in Figures 3 and 5 are congested and may be difficult to read, one of them is shown in Figure 4 in a larger scale. (The dashed lines in Figs. 3–5 will be discussed in Sect. 4.3.) Examining these data, the following may be concluded:

- As expected, k_E for non-exponential machines is smaller than k_E^{exp}. Moreover, k_E is a monotonically increasing function of CV. In addition, $k_E(CV)$ is convex, which implies that reducing larger CV's leads to larger reduction of k_E than reducing smaller CV's.
- Function $k_E(CV)$ seems to be polynomial in nature. In fact, each curve of Figures 3 and 5 can be approximated by a polynomial of an appropriate order. However, since these approximations are "parameter-dependent" (i.e., different polynomials must be used for different e and E), they are of small practical importance, and, therefore, are not reported here.
- Since for every pair (E, e), corresponding curves of Figures 3 and 5 are identical, it is concluded that k_E is not dependent of T_{up} and T_{down} explicitly but only through the ratio e. In other words, the situation here is the same as in lines with exponential machines (see (5), (6)).

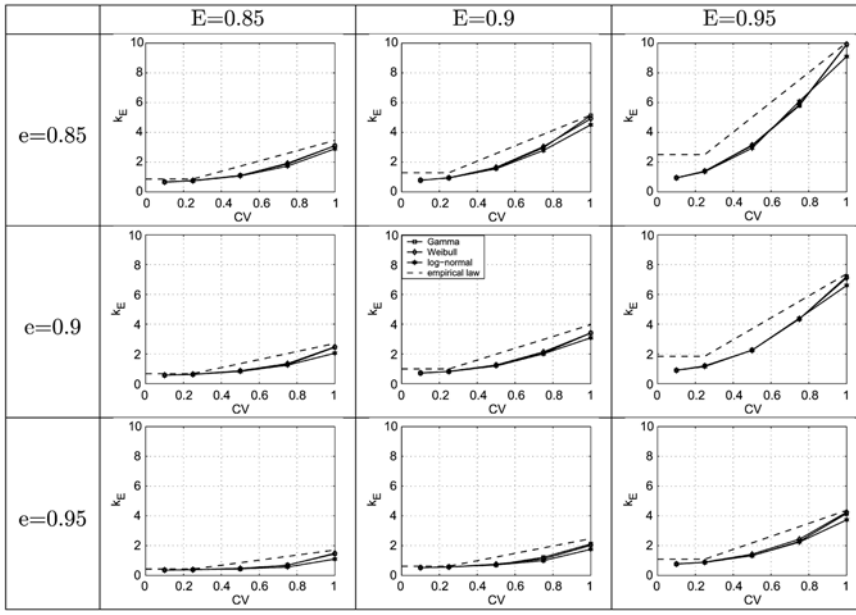


Fig. 3. LLB versus CV for systems (17) with $T_{down} = 20$

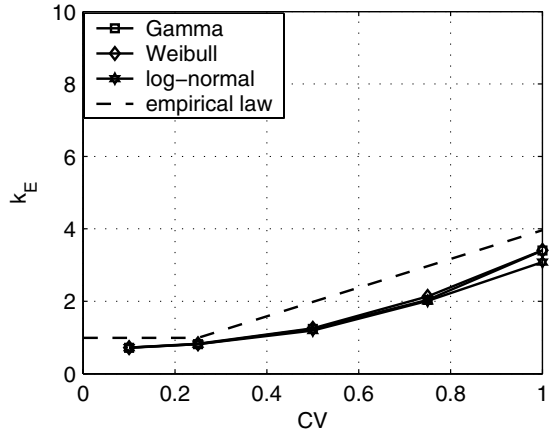


Fig. 4. LLB versus CV for system $\{(D(p,P), D(r,R))_i, i = 1, \ldots, 10\}$ with $T_{down} = 20$, $e = 0.9$, $E = 0.9$

– Finally, and perhaps most importantly, the behavior of k_E as a function of CV is almost independent of the type of up- and downtime distributions considered. Indeed, let $k_E^A(CV)$ denote LLB for line $A \in \{(17)\}$ with $CV \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$. Then the sensitivity of k_E to up- and downtime distributions may be characterized by

$$\epsilon_1(CV) = \max_{A,B \in \{(17)\}} \left| \frac{k_E^A(CV) - k_E^B(CV)}{k_E^A(CV)} \right| \cdot 100\%. \tag{19}$$

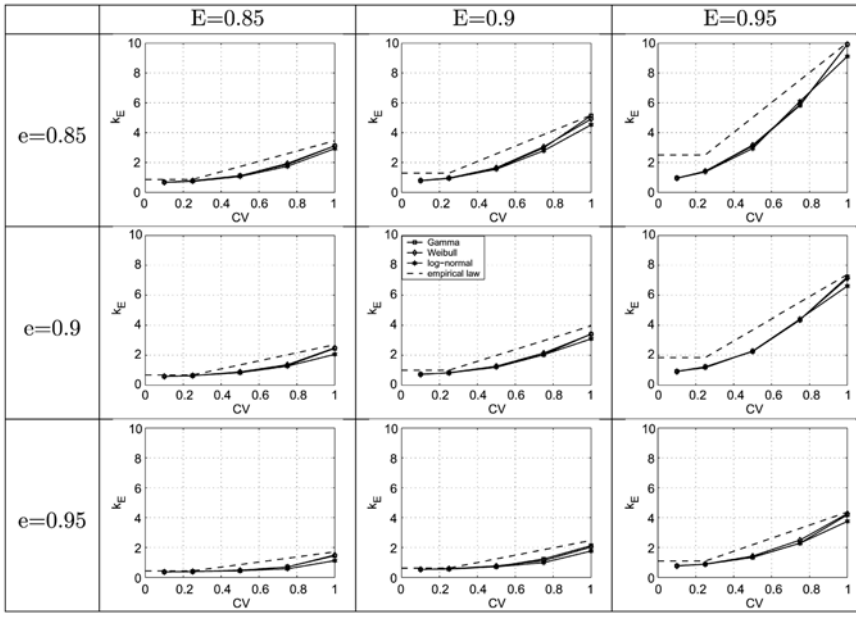


Fig. 5. LLB versus CV for systems (17) with $T_{down} = 100$

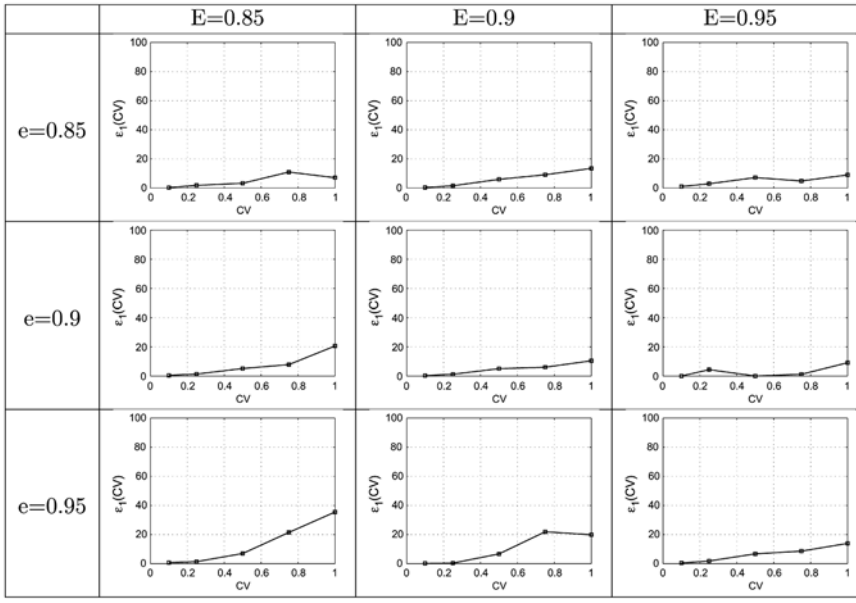


Fig. 6. Sensitivity of LLB to the nature of up- and downtime distributions for systems (17)

Function $\epsilon_1(CV)$ is illustrated in Figure 6. As one can see, in most cases it takes values within 10%. Thus, it is possible to conclude that for all practical purposes k_E depends on the coefficients of variation of up- and downtime, rather than on actual distribution of these random variables.

4.2 System $\{[D(p,P),D(r,R)]_1,\ldots,[D(p,P),D(r,R)]_{10}\}$

Figures 7 and 8 present the simulation results for lines (18), while Figure 9 characterizes the sensitivity of k_E to up- and downtime distributions. This sensitivity is calculated according to (19) with the only difference that the max is taken over $A,B\in\{(18)\}$. Based on these data, we affirm that the conclusions formulated in Section 4.1 hold for production lines of the type (13) as well.

4.3 Empirical law

4.3.1 Analytical expression

Simulation results reported above provide a characterization of k_E for $M=10$ and E and $e\in\{0.85,0.9,0.95\}$. How can k_E be determined for other values of M, E, and e? Obviously, simulations for all values of these variables are impossible. Even for particular values of M, E, and e, simulations take a very long time: Figures 3 and 5 required approximately one week of calculations using 25 Sun workstations working in parallel. Therefore, an analytical method for evaluating k_E for all values of M, E, e, and CV is desirable. Although an exact characterization of the function $k_E=k_E(M,E,e,CV)$ is all but impossible, results of Sections 4.1 and 4.2 provide an opportunity for introducing an upper bound of k_E as a function of all four variables. This upper bound is based on the expression of $k_E^{exp}=k_E^{exp}(M,E,e)$, given by (5), (6), and the fact that all curves of Figures 3, 5 and 7, 8 are below the linear function of CV with the slope k_E^{exp}, if $0.25<CV\le 1$. For $0<CV\le 0.25$, all curves are below the constant $0.25k_E^{exp}$. Thus, the following piece-wise linear upper bound for k_E may be introduced:

$$k_E(M,E,e,CV)\le\max\{0.25,CV\}k_E^{exp}(M,E,e),\qquad CV\le 1. \tag{20}$$

This expression, referred to as *the empirical law*, is illustrated in Figures 3-5 and 7, 8 by the broken lines.

The tightness of this bound can be characterized by the function

$$\epsilon_2(CV)=\max_{A\in\{(17),(18)\}}\frac{k_E^{\text{upper bound}}-k_E^A}{k_E^A}\cdot 100\%,\qquad CV\le 1, \tag{21}$$

where $k_E^{\text{upper bound}}$ is the right-hand-side of (20). Function $\epsilon_2(CV)$ is illustrated in Figure 10. Although, as one can see, the empirical law is quite conservative, its usage still leads to up to 400% reduction of buffering, as compared with that based on the exponential assumption (see Figs. 3, 5 and 7, 8).

Remark 4. As it was pointed out above, the curves of Figures 3, 5 and 7, 8 are polynomial in nature. This, along with the quadratic dependence of performance

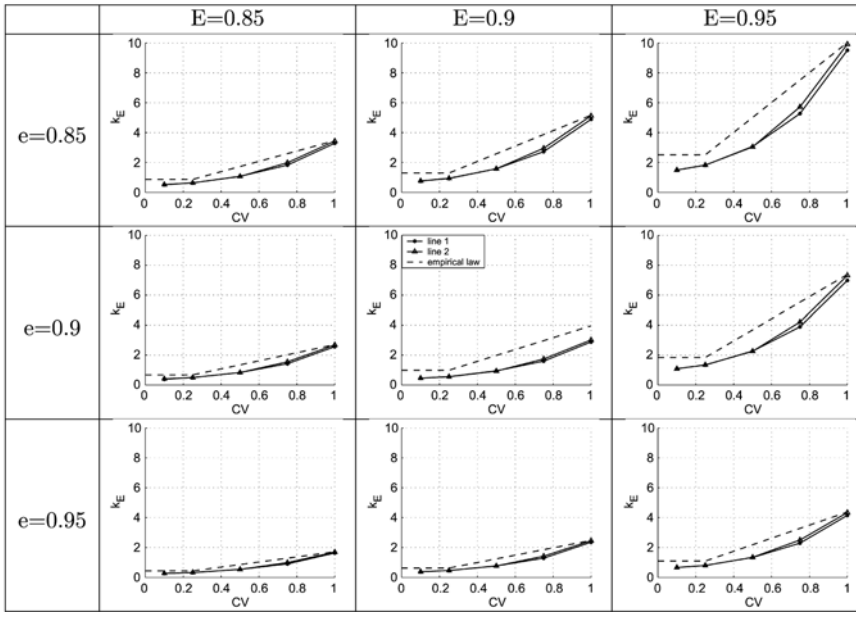


Fig. 7. LLB versus CV for systems (18) with $T_{down} = 20$

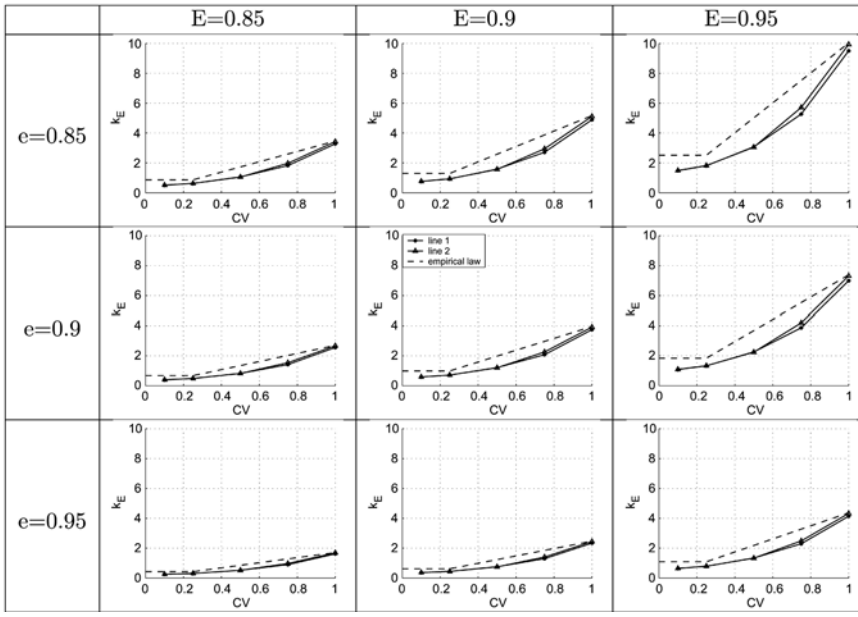


Fig. 8. LLB versus CV for systems (18) with $T_{down} = 100$

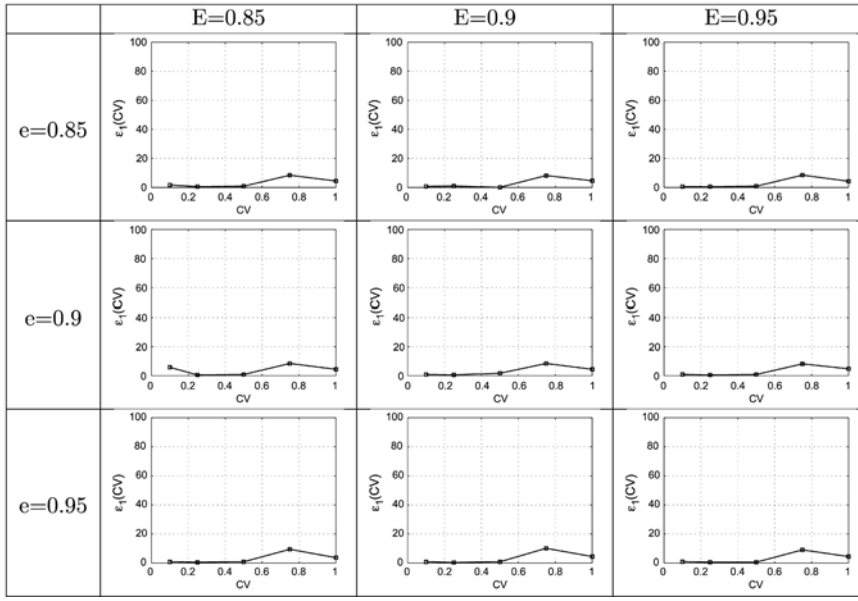


Fig. 9. Sensitivity of LLB to the nature of up- and downtime distributions for systems (18)

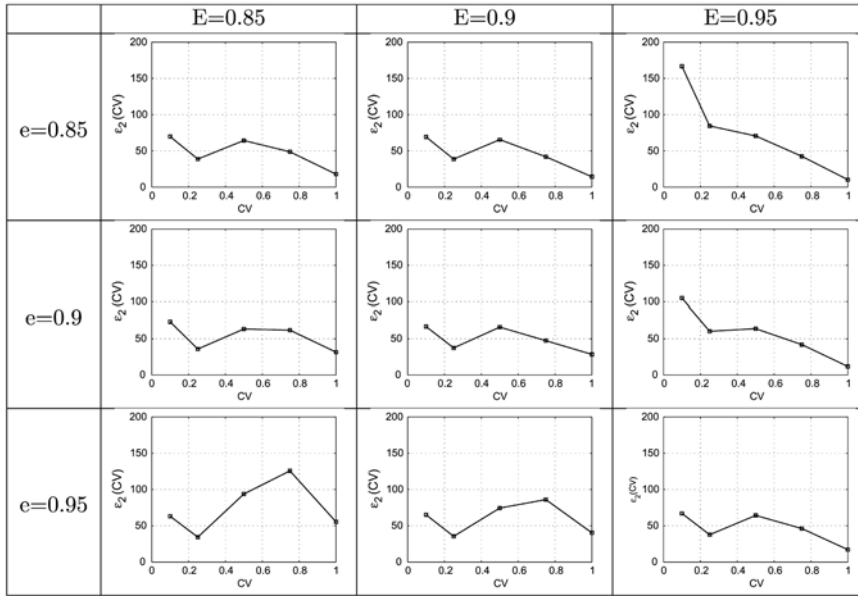


Fig. 10. The tightness of the empirical law (20)

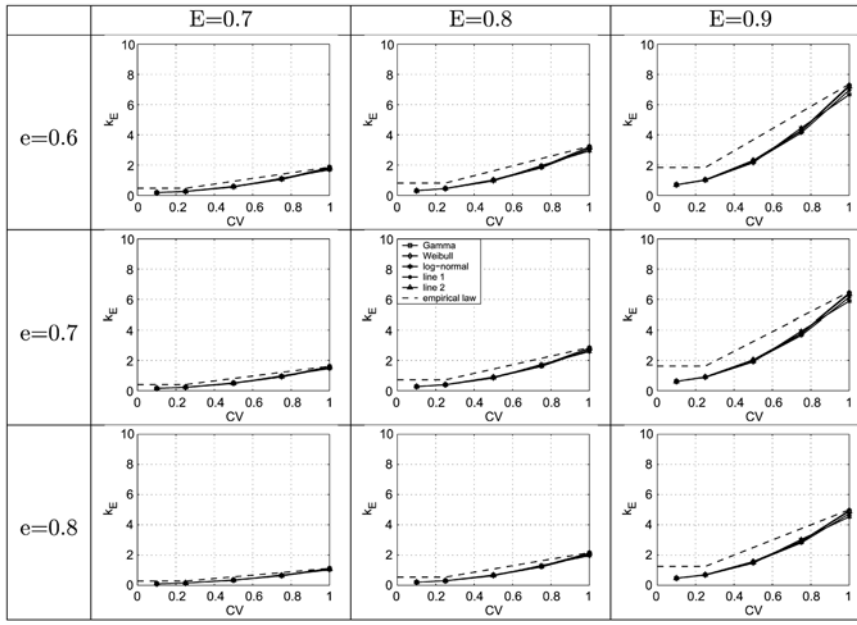


Fig. 11. Verification: LLB versus CV for system $\{(D(p,P), D(r,R))_i, i = 1,\ldots,5\}$ with $T_{down} = 10$

measures on CV in $G/G/1$ queues, might lead to a temptation to approximate these curves by polynomials. This, however, proved to be practically impossible, since for various values of M, E, and e, the order and the coefficients of the polynomials would have to be selected differently. This, together with the fact that only one point is known analytically (i.e., k_E^{exp}), leads to the selection of the piece-wise linear approximation (20).

4.3.2 Verification

To verify the empirical law (20), production lines (17) and (18) were simulated with parameters M, E, and e other than those considered in Sections 4.1 and 4.2. Specifically, the following parameters have been selected: $M = 5$, $E \in \{0.7, 0.8, 0.9\}$, $e \in \{0.6, 0.7, 0.8\}$, $T_{down} = 10$. (In lines (18), the first 5 machines were selected.) The results are shown in Figure 11. As one can see, the upper bound given by (20) still holds.

5 LLB in serial lines with $CV_{up} \neq CV_{down}$

5.1 Effect of CV_{up} and CV_{down}

The case of $CV_{up} \neq CV_{down}$ is complicated by the fact that CV_{up} and CV_{down} may have different effects on k_E. If this difference is significant, it would be difficult

to expect that the empirical law (20) could be extended to the case of unequal coefficients of variation. On the other hand, if CV_{up} and CV_{down} affect k_E in a somewhat similar manner, it would seem likely that (20) might be extended to the case under consideration. Therefore, analysis of effects of CV_{up} and CV_{down} on k_E is of importance. This section is devoted to such an analysis.

To investigate this issue, introduce two functions:

$$k_E(CV_{up}|CV_{down} = \alpha) \tag{22}$$

and

$$k_E(CV_{down}|CV_{up} = \alpha), \tag{23}$$

where

$$\alpha \in \{0.1, 0.25, 0.5, 0.75, 1.0\}. \tag{24}$$

Function (22) describes k_E as a function of CV_{up} given that $CV_{down} = \alpha$, while (23) describes k_E as a function of CV_{down} given that $CV_{up} = \alpha$. If for all α and $\beta \in \{0.1, 0.25, 0.5, 0.75, 1.0\}$,

$$k_E(CV_{down} = \beta|CV_{up} = \alpha) < k_E(CV_{up} = \beta|CV_{down} = \alpha) \tag{25}$$

when $\alpha > \beta$, it must be concluded that CV_{down} has a larger effect on k_E than CV_{up}. If the inequality is reversed, CV_{up} has a stronger effect. Finally, if (25) holds for some α and β from (24) and does not hold for others, the conclusion would be that, in general, neither has a dominant effect.

To investigate which of these situations takes place, we evaluated functions (22) and (23) using the approach described in Section 3. Some of the results for Weibull distribution are shown in Figure 12 (where the broken lines and CV_{eff} will be defined in Sect. 5.2). Similar results were obtained for gamma and log-normal distributions as well (see Enginarlar et al., 2003b for details). From these results, the following can be concluded:

- For all α and β, such that $\alpha > \beta$, inequality (25) takes place. Thus, CV_{down} has a larger effect on k_E than CV_{up}.
- However, since each pair of curves (22), (23) corresponding to the same α are close to each other, the difference in the effects of CV_{up} and CV_{down} is not too dramatic. To analyze this difference, introduce the function

$$\epsilon_3^A(CV|CV_{up} = CV_{down} = \alpha) = \frac{k_E^A(CV_{up}{=}CV|CV_{down} = \alpha) - k_E^A(CV_{down}{=}CV|CV_{up}{=}\alpha)}{k_E^A(CV_{up}{=}CV|CV_{down}{=}\alpha)} \cdot 100\,, \tag{26}$$

  where $A \in \{W, g, LN\}$. The behavior of this function for Weibull distribution is shown in Figure 13 (see Enginarlar et al., 2003b for gamma and log-normal distributions). Thus, the effects of CV_{up} and CV_{down} on k_E are not dramatically different (typically within 20% and no more than 40%).

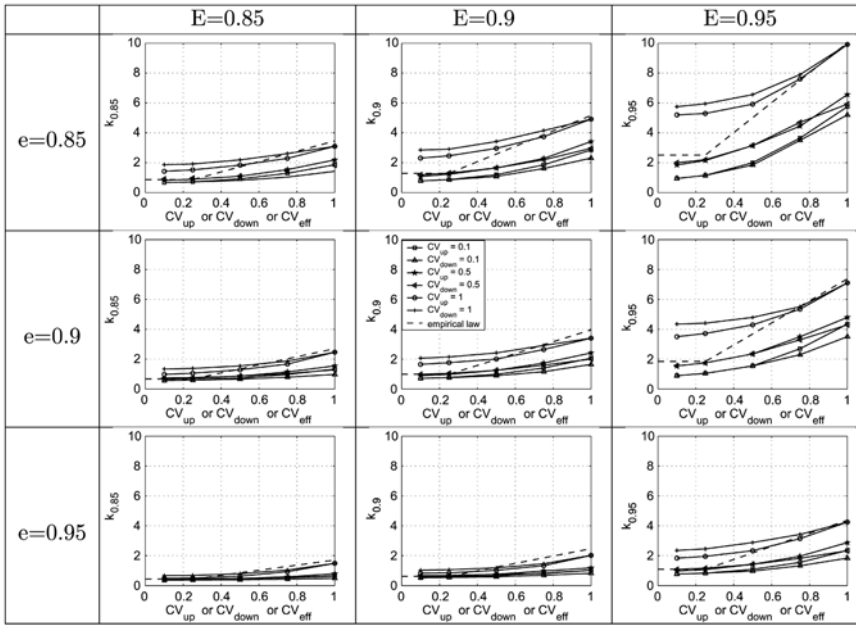


Fig. 12. LLB versus CV for $M = 10$ Weibull machines

5.2 Empirical law

5.2.1 Analytical expression

Since the upper bound (20) is not too tight (and, hence, may accommodate additional uncertainties) and the effects of CV_{up} and CV_{down} on k_E are not dramatically different, the following extension of the empirical law is suggested:

$$k_E(M, E, e, CV_{up}, CV_{down}) \leq \frac{\max\{0.25, CV_{up}\} + \max\{0.25, CV_{down}\}}{2} k_E^{exp}(M, E, e), \quad CV_{up} \leq 1, \quad CV_{down} \leq 1, \tag{27}$$

where, as before, k_E^{exp}, is defined by (5), (6). If $CV_{up} = CV_{down}$, (27) reduces to (20); otherwise, it takes into account different values of CV_{up} and CV_{down}.

The first factor in the right-hand-side of (27) is denoted as CV_{eff}:

$$CV_{eff} = \frac{\max\{0.25, CV_{up}\} + \max\{0.25, CV_{down}\}}{2}. \tag{28}$$

Thus, (27) can be rewritten as

$$k_E \leq CV_{eff} k_E^{exp}(M, E, e). \tag{29}$$

The right-hand-side of (29) is shown in Figure 12 by the broken lines.

The utilization of this law can be illustrated as follows: Suppose $CV_{up} = 0.1$ and $CV_{down} = 1$. Then $CV_{eff} = 0.625$ and, according to (27),

$$k_E \leq 0.625 k_E^{exp}(M, E, e).$$

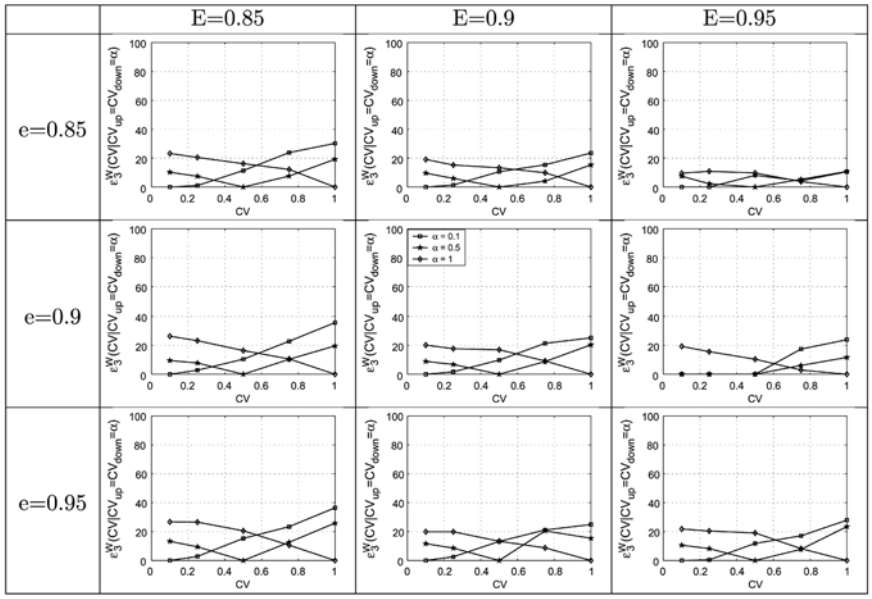


Fig. 13. Function $\epsilon_3^W(CV|CV_{up} = CV_{down} = \alpha)$

Table 4. $\Delta(10, E, e)$ for all $CV_{up} \neq CV_{down}$ cases considered

	E=0.85	E=0.9	E=0.95
$e = 0.85$	0.1016	0.0386	0.0687
$e = 0.9$	0.0425	0.1647	0.1625
$e = 0.95$	0.0402	0.0488	0.1200

To investigate the validity of the empirical law (27), consider the following function:

$$\Delta(M, E, e) = \min_{A \in \{(17)\}} \min_{CV_{up}, CV_{down} \in \{(24)\}} \left[k_E^{\text{upper bound}}(M, E, e, CV_{eff}) - k_E^A(M, E, e, CV_{up}, CV_{down}) \right], \tag{30}$$

where $k_E^{\text{upper bound}}$ is the right-hand-side of (29), i.e.,

$$k_E^{\text{upper bound}}(M, E, e, CV_{eff}) = CV_{eff} k_E^{exp}(M, E, e).$$

If for all values of its arguments, function $\Delta(M, E, e)$ is positive, the right-hand-side of inequality (27) is an upper bound. The values of $\Delta(10, E, e)$ for $E \in \{0.85, 0.9, 0.95\}$ and $e \in \{0.85, 0.9, 0.95\}$ are shown in Table 4. As one can see, function $\Delta(10, E, e)$ indeed takes positive values. Thus, the empirical law (27) takes place for all distributions and parameters analyzed.

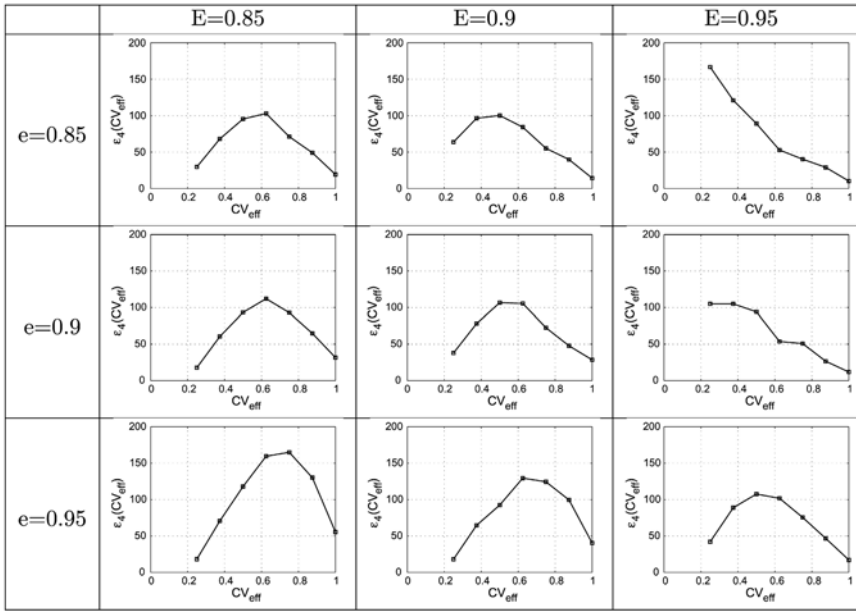


Fig. 14. The tightness of the empirical law (27)

To investigate the tightness of the bound (27), consider the function

$$\epsilon_4(CV_{eff}) = \max_{A\in\{(17)\}} \max_{CV_{up},CV_{down}\in\{(24)\}} \frac{k_E^{\text{upperbound}}(M,E,e,CV_{eff}) - k_E^A(M,E,e,CV_{up},CV_{down})}{k_E^A(M,E,e,CV_{up},CV_{down})} \cdot 100\,. \tag{31}$$

Figure 14 illustrates the behavior of this function. Comparing this with Figure 10, we conclude that the tightness of bound (27) appears to be similar to that of (20).

5.2.2 Verification

To evaluate the validity of the upper bound (27), serial production lines with $M = 5$, $E \in \{0.7, 0.8, 0.9\}$, $e \in \{0.6, 0.7, 0.8\}$, and $T_{up} = 10$ were simulated. For each of these parameters, systems (17) and (18) have been considered. (For system (18), the first 5 machines were selected.) Typical results are shown in Figure 15 (see Enginarlar et al., 2003b for more details). The validity of empirical law (27) for these cases is analyzed using function $\Delta(M, E, e)$, defined in (30) with the only difference that the first min is taken over $A \in \{(17), (18)\}$. Since the values of this function, shown in Table 5, are positive, we conclude that empirical law (27) is indeed verified for all values of M, E, e, and all distributions of up- and downtime considered.

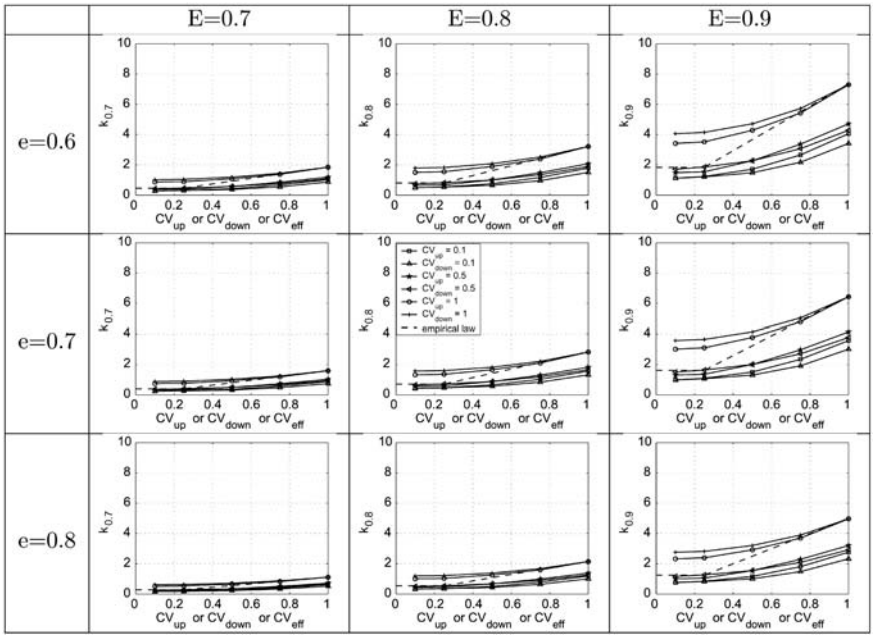


Fig. 15. Verification: LLB versus CV for $M = 5$ Weibull machines

Table 5. Verification: $\Delta(5, E, e)$ for all $CV_{up} \neq CV_{down}$ cases considered

	E=0.7	E=0.8	E=0.9
$e = 0.6$	0.0039	0.0242	0.0547
$e = 0.7$	0.0102	0.0213	0.0481
$e = 0.8$	0.0084	0.0162	0.0355

6 SYSTEM $\{[G_{up}, G_{down}]_1, \ldots, [G_{up}, G_{down}]_M\}$

So far, serial production lines with Weibull, gamma, and log-normal reliability models have been analyzed. It is of interests to extend this analysis to general probability density functions. Based on the results obtained above, the following conjecture is formulated:

The empirical laws (20) and (27) hold for serial production lines satisfying assumptions (i), (iii)–(vi) with up- and downtime having arbitrary unimodal probability density functions.

The verification of this conjecture is a topic for future research.

7 Conclusions

Results described in this paper suggest the following procedure for designing lean buffering in serial production lines defined by assumptions (i)–(vi):

1. Identify the average value and the variance of the up- and downtime, T_{up}, T_{down}, σ_{up}^2, and σ_{down}^2, for all machines in the system (in units of machine cycle time). This may be accomplished by measuring the duration of the up- and downtimes of each machine during a shift or a week of operation (depending on the frequency of occurrence). If the production line is at the design stage, this information may be obtained from the equipment manufacturer (however, typically with a lower level of certainty).
2. Using (5), (6), and T_{up}, T_{down}, determine the level of buffering, necessary and sufficient to obtain the desired efficiency, E, of the production line, if the downtime of all machines were distributed exponentially, i.e., k_E^{exp}.
3. Finally, if $CV_{up} = \frac{\sigma_{up}}{T_{up}} \leq 1$ and $CV_{down} = \frac{\sigma_{down}}{T_{down}} \leq 1$, evaluate the level of buffering for the line with machines under consideration using the empirical law

$$k_E \leq \frac{\max\{0.25, CV_{up}\} + \max\{0.25, CV_{down}\}}{2} \cdot k_E^{exp}.$$

As it is shown in this paper, this procedure leads to a reduction of lean buffering by a factor of up to 4, as compared with that based on the exponential assumption.

References

Altiok T (1985) Production lines with phase-type operation and repair times and finite buffers. International Journal of Production Research 23: 489–498

Altiok T (1989) Approximate analysis of queues. In: Series with phase-type service times and blocking. Operations Research 37: 601–610

Altiok T, Stidham SS (1983) The allocation of interstage buffer capacities in production lines. IIE Transactions 15: 292–299

Altiok T, Ranjan R (1989) Analysis of production lines with general service times and finite buffers: a two-node decomposition approach. Engineering Costs and Production Economics 17: 155–165

Buzacott JA (1967) Automatic transfer lines with buffer stocks. International Journal of Production Research 5: 183–200

Caramanis M (1987) Production line design: a discrete event dynamic system and generalized benders decomposition approach. International Journal of Production Research 25: 1223–1234

Chow W-M (1987) Buffer capacity analysis for sequential production lines with variable processing times. International Journal of Production Research 25: 1183–1196

Conway R, Maxwell W, McClain JO, Thomas LJ (1988) The role of work-in-process inventory in serial production lines. Operations Research 36: 229–241

Enginarlar E, Li J, Meerkov SM, Zhang RQ (2002) Buffer capacity to accommodating machine downtime in serial production lines. International Journal of Production Research 40: 601–624

Enginarlar E, Li J, Meerkov SM (2003a) How lean can lean buffers be? Control Group Report CGR 03-10, Deptartment of EECS, University of Michigan, Ann Arbor, MI; accepted for publication in IIE Transactions on Design and Manufacturing (2005)

Enginarlar E, Li J, Meerkov SM (2003b) Lean buffering in serial production lines with non-exponential machines. Control Group Report CGR 03-13, Deptartment of EECS, University of Michigan, Ann Arbor, MI

Gershwin SB, Schor JE (2000) Efficient algorithms for buffer space allocation. Annals of Operations Research 93: 117–144

Harris JH, Powell SG (1999) An algorithm for optimal buffer placement in reliable serial lines. IIE Transactions 31: 287–302

Hillier FS, So KC (1991a) The effect of the coefficient of variation of operation times on the allocation of storage space in production line systems. IIE Transactions 23: 198–206

Hillier FS, So KC (1991b) The effect of machine breakdowns and internal storage on the performance of production line systems. International Journal of Production Research 29: 2043–2055

Inman RR (1999) Empirical evaluation of exponential and independence assumptions in queueing models of manufacturing systems. Production and Operation Management 8: 409–432

Jafari MA, Shanthikumar JG (1989) Determination of optimal buffer storage capacity and optimal allocation in multistage automatic transfer lines. IIE Transactions 21: 130–134

Li J, Meerkov SM (2005) On the coefficients of variation of up- and downtime in manufacturing equipment. Mathematical Problems in Engineering 2005: 1–6

Park T (1993) A two-phase heuristic algorithm for determining buffer sizes in production lines. International Journal of Production Research 31: 613–631

Powell SG (1994) Buffer allocation in unbalanced three-station lines. International Journal of Production Research 32: 2201–2217

Powell SG, Pyke DF (1998) Buffering unbalanced assembly systems. IIE Transactions 30: 55–65

Seong D, Change SY, Hong Y (1995) Heuristic algorithm for buffer allocation in a production line with unreliable machines. International Journal of Production Research 33: 1989–2005

Smith JM, Daskalaki S (1988) Buffer space allocation in automated assembly lines. Operations Research 36: 343–357

Tempelmeier H (2003) Practical considerations in the optimization of flow production systems. International Journal of Production Research 41: 149–170

Yamashita H, Altiok T (1988) Buffer capacity allocation for a desired throughput of production lines. IIE Transactions 30: 883–891

Analysis of flow lines with Cox-2-distributed processing times and limited buffer capacity*

Stefan Helber

University of Hannover, Department for Production Management, Königsworther Platz 1, 30167 Hannover, Germany (e-mail: stefan.helber@prod.uni-hannover.de)

Abstract. We describe a flow line model consisting of machines with Cox-2-distributed processing times and limited buffer capacities. A two-machine subsystem is analyzed exactly and a larger flow lines are evaluated through a decomposition into a set of coupled two-machine lines. Our results are compared to those given by Buzacott, Liu and Shantikumar for their "Stopped Arrival Queue Modell".

Keywords: Flow line – Performance evaluation – Decomposition – General processing times – Cox-2-distribution

1 Introduction

We describe an approximate approach to determine the production rate and inventory level of a flow line consisting of more than two machines where adjacent machines are decoupled through buffers of limited capacity. We assume that machines are reliable and that processing times are Cox-2-distributed. This allows us to model processing times with any squared coefficient of variation $c^2 \geq 0.5$. These processing times can include the random delay of workpieces which is due to random failures and repairs of the machines if we use the completion time concept proposed by Gaver [17].

Several researchers have studied transfer lines or assembly/disassembly (A/D) systems with limited buffer capacity. A comprehensive survey is given by Dallery and Gershwin [15]. This 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 [7, 11], and [28].

Transfer lines and A/D systems are often studied using Markov chain or process models to allow for an analytic solution or an accurate approximation. Many of these

* The author thanks the anonymous referees for their helpful comments and suggestions.

Table 1. Two-machine models and approximation approaches

Type of process	Analysis of two-machine models	Approximate decomposition approaches
Discrete state/ discrete time	[2,5,7,24,26,29,34]	[13,18,20,25]
Discrete state/ continuous time	[6,22,31]	[12,19,25]
Continuous state/ continuous time	[21,23,32,33,35]	[4,14,16]

approximations are based on a decomposition of the complete system into a set of single server queues [27] or two-machine transfer lines [18,32,35] which can be evaluated analytically. The main advantage of analytical approaches as opposed to simulation models is that the analytical techniques are much faster. This is crucial if a large 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 [15]

- 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. An additional possible reason to have discrete states is that machines can be either operational or under repair. 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. In this paper we describe a discrete-state, continuous-time model where the discrete states reflect discrete buffer levels.

Table 1 gives an overview of two-machine models and decomposition approaches for the case of limited buffer capacity. In many of these papers machines are assumed to be unreliable. Textbooks covering these and similar techniques in detail are [1,10,30] as well as [21] which gives a thorough introduction into how to derive these models. In this paper, we develop a two-machine transfer line decomposition of the discrete state-continuous time type. We assume, however, that machines are reliable and that processing times may exhibit variability with any squared coefficient of variation larger than 0.5. The two papers most closely related to this one are an older paper by Buzacott and Kostelski [8] on the analysis of a specific two-machine line and by Buzacott et al. [9] on particular decomposition techniques for longer lines with limited buffer capacity.

The paper is structured as follows: In Section 2 we formally describe the type of flow line to be analyzed. Section 3 outlines the exact analysis of the two-machine, one-buffer subsystem that serves as the building block of a decomposition and

which has already been analyzed by Buzacott and Kostelski [8] using the Matrix geometric method. Our analysis of the two-machine system, however, follows the approach for two-machine systems which is thoroughly explained by Gershwin [21]. The decomposition algorithm is briefly described in Section 4. In Section 5 we present some preliminary numerical results by comparing our results to those obtained from the multistage flow line analysis with the stopped arrival queue model as proposed by Buzacott et al. [9].

2 The model

We assume that the flow line consists of M machines or stages. The processing times at machine M_i follow a Cox-2 distribution. Each buffer B_i between machines M_i and M_{i+1} has the capacity to hold up to C_i workpieces which flow from the leftmost to the rightmost machine. An example of such a flow line is depicted in Figure 1.

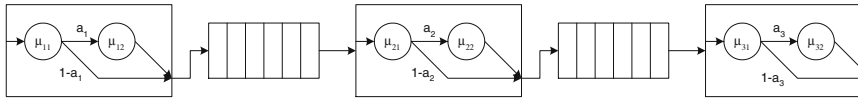


Fig. 1. Flow line with three machines

The rates of the two phases of stage i are μ_{i1} and μ_{i2} respectively. The second phase of stage i will be required after completion of phase one with probability a_i. Therefore, a workpiece completes its service at stage i with probability $1 - a_i$ after the completion of the first phase and with probability a_i after the completion of the second phase. Note that these states of a machine or stage do not represent servers: No more than one workpiece can be at a machine at any moment in time, and if it is there, it is in one out of the two phases of the respective machine. Each machine M_i except for the first and the last can be either idle (starved) or blocked or it can be processing a part in phase one or two. The state of machine M_i is denoted as $\alpha_i(t)$. The possible machine states are $\alpha_i(t) \in \{1, 2, B, S\}$, representing phase one, phase two, blocking and starvation. The buffer level $n(i)$ is defined such that it includes the parts between machines M_i and M_{i+1}, the one part residing at machine M_{i+1} (if this machine is not starved) and the one part waiting at machine M_i if this machine is blocked because the buffer between M_i and M_{i+1} is full.

3 The two-maschine subsystem

3.1 State space and transition equations

In order to analyze larger systems with more than two machines, we first study a two-machine line. The state of this two-machine line is given by the state of the first machine, the state of the second machine, and the buffer level. In the analysis to follow, we define the buffer level include all parts that are currently being processed at the second machine, that are waiting in the physical buffer between the first and

the second machine, and those parts that have been processed at the first machine but cannot leave it because the physical buffer between the machines is full so that the first machine is blocked. That is, we follow the blockage convention which is described in [21, p. 95]. The (total or extended) buffer capacity is therefore $N_i = C_i + 2$. In order to describe the state space, we use use the triple (n, α_1, α_2) where n denotes the buffer level. The probability of the system being in this state is $\mathbf{p}(n, \alpha_1, \alpha_2)$. Machine M_1 can either be in the first phase ($\alpha_1 = 1$), in the second phase ($\alpha_1 = 2$) or it can be blocked ($\alpha_1 = B$). The downstream machine M_2 can either be in the first phase ($\alpha_2 = 1$), in the second phase ($\alpha_2 = 2$) or it can be starved ($\alpha_2 = S$).

This leads to the following transition equations which differ for states with an empty or almost empty buffer, states for a full a almost full buffer, and the states with a buffer level that is in between:

Lower boundary states:

$$\mu_{11}\mathbf{p}(0,1,S) = (1-a_2)\mu_{21}\mathbf{p}(1,1,1) + \mu_{22}\mathbf{p}(1,1,2) \tag{1}$$

$$\mu_{12}\mathbf{p}(0,2,S) = a_1\mu_{11}\mathbf{p}(0,1,S) + (1-a_2)\mu_{21}\mathbf{p}(1,2,1) + \mu_{22}\mathbf{p}(1,2,2) \tag{2}$$

$$(\mu_{11}+\mu_{21})\mathbf{p}(1,1,1) = (1-a_1)\mu_{11}\mathbf{p}(0,1,S) + \mu_{12}\mathbf{p}(0,2,S) + (1-a_2)\mu_{21}\mathbf{p}(2,1,1) + \mu_{22}\mathbf{p}(2,1,2) \tag{3}$$

$$(\mu_{11}+\mu_{22})\mathbf{p}(1,1,2) = a_2\mu_{21}\mathbf{p}(1,1,1) \tag{4}$$

Intermediate stages:

$$(\mu_{11}+\mu_{21})\mathbf{p}(n,1,1) = (1-a_1)\mu_{11}\mathbf{p}(n-1,1,1) + \mu_{12}\mathbf{p}(n-1,2,1) + (1-a_2)\mu_{21}\mathbf{p}(n+1,1,1) + \mu_{22}\mathbf{p}(n+1,1,2) \quad (\text{for } 2 \le n \le N-2) \tag{5}$$

$$(\mu_{11}+\mu_{22})\mathbf{p}(n,1,2) = (1-a_1)\mu_{11}\mathbf{p}(n-1,1,2) + \mu_{12}\mathbf{p}(n-1,2,2) + a_2\mu_{21}\mathbf{p}(n,1,1) \quad (\text{for } 2 \le n \le N-1) \tag{6}$$

$$(\mu_{12}+\mu_{21})\mathbf{p}(n,2,1) = a_1\mu_{11}\mathbf{p}(n,1,1) + \mu_{22}\mathbf{p}(n+1,2,2) + (1-a_2)\mu_{21}\mathbf{p}(n+1,2,1) \quad (\text{for } 1 \le n \le N-2) \tag{7}$$

$$(\mu_{12}+\mu_{22})\mathbf{p}(n,2,2) = a_1\mu_{11}\mathbf{p}(n,1,2) + a_2\mu_{21}\mathbf{p}(n,2,1) \quad (\text{for } 2 \le n \le N-1) \tag{8}$$

Upper boundary states:

$$\mu_{21}\mathbf{p}(N,B,1) = (1-a_1)\mu_{11}\mathbf{p}(N-1,1,1) + \mu_{12}\mathbf{p}(N-1,2,1) \tag{9}$$

$$\mu_{22}\mathbf{p}(N,B,2) = a_2\mu_{21}\mathbf{p}(N,B,1) + (1-a_1)\mu_{11}\mathbf{p}(N-1,1,2) + \mu_{12}\mathbf{p}(N-1,2,2) \tag{10}$$

$$(\mu_{11}+\mu_{21})\mathbf{p}(N-1,1,1) = (1-a_1)\mu_{11}\mathbf{p}(N-2,1,1) + \mu_{12}\mathbf{p}(N-2,2,1) + (1-a_2)\mu_{21}\mathbf{p}(N,B,1) + \mu_{22}\mathbf{p}(N,B,2) \quad (11)$$

$$(\mu_{12}+\mu_{21})\mathbf{p}(N-1,2,1) = a_1\mu_{11}\mathbf{p}(N-1,1,1) \quad (12)$$

Together with the normalization equation

$$\mathbf{p}(0,1,S) + \mathbf{p}(0,2,S) + \left(\sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}\sum_{\alpha_2=1}^{2}\mathbf{p}(n,\alpha_1,\alpha_2)\right) + \mathbf{p}(N,B,1) + \mathbf{p}(N,B,2) = 1 \quad (13)$$

this leads to a linear system of equations which can be solved in several ways. An almost identical system of equations has been formulated in [8] and solved via the matrix geometric method and a recursive algorithm. Since their methods suffered from numerical instabilities, we developed a solution technique using the ideas for the analysis of two-machine models presented in [21, pp.105]. It leads to a numerically stable algorithm providing the exact values of all the system states as well as the performance measures such as the production rate and the inventory level.

3.2 Identities

Conservation of flow. The rate at which parts leave machine M_1 is the product of the steady-state probabilities of all states where M_1 is not blocked times the respective rate for this state:

$$PR_1 = \mu_{11}(1-a_1)\mathbf{p}(0,1,S) + \mu_{12}\mathbf{p}(0,2,S) + \sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}(\mu_{11}(1-a_1)\mathbf{p}(n,1,\alpha_2) + \mu_{12}\mathbf{p}(n,2,\alpha_2)) \quad (14)$$

The reasoning for machine M_2 (which may not be starved) is similar:

$$PR_2 = \sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}(\mu_{21}(1-a_2)\mathbf{p}(n,\alpha_1,1) + \mu_{22}\mathbf{p}(n,\alpha_1,2)) + \mu_{21}(1-a_2)\mathbf{p}(N,B,1) + \mu_{22}\mathbf{p}(N,B,2) \quad (15)$$

The Conservation-of-Flow-identity (COF) states that the rates of parts passing through machines M_1 and M_2 are equal:

$$PR_1 = PR_2 \quad (16)$$

The reason is that the flow of material is linear and parts are neither created nor destroyed at either machine.

Rate of changes from phase one to two equals rate of changes from phase two to one. For each change of machine M_1 from phase one to phase two there must be a change from phase two to phase one

$$\begin{aligned} & a_1\mu_{11}\left(\mathbf{p}(0,1,S)+\sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}\mathbf{p}(n,1,\alpha_2)\right) \\ = \;& \mu_{12}\left(\mathbf{p}(0,2,S)+\sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}\mathbf{p}(n,2,\alpha_2)\right) \end{aligned} \tag{17}$$

and the same holds true for machine M_2:

$$\begin{aligned} & a_2\mu_{21}\left(\mathbf{p}(N,B,1)+\sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}\mathbf{p}(n,\alpha_1,1)\right) \\ = \;& \mu_{22}\left(\mathbf{p}(N,B,2)+\sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}\mathbf{p}(n,\alpha_1,2)\right) \end{aligned} \tag{18}$$

Flow-Rate-Idle-Time-Equations. The Flow-Rate-Idle-Time-Equations (FRIT-Equations) relate the flow or production rates of the up- and downstream machines to the probability of the respective machine being blocked or starved.

The expected processing time $\mathbf{E}[T_1]$ at the upstream machine M_1 of a two-machine-line is the weighted sum of the expected processing time $\frac{1}{\mu_{11}}$ if a workpiece only goes through phase one (which happens with probability $(1-a_1)$) and the expected processing time $\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}$ if it undergoes both phases (with probability a_1):

$$\mathbf{E}[T_1]=(1-a_1)\frac{1}{\mu_{11}}+a_1\left(\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}\right) \tag{19}$$

The reasoning for the expected processing time $\mathbf{E}[T_2]$ at at the second (downstream) machine of a two-machine-line leads to an analogous result:

$$\mathbf{E}[T_2]=(1-a_2)\frac{1}{\mu_{21}}+a_2\left(\frac{1}{\mu_{21}}+\frac{1}{\mu_{22}}\right) \tag{20}$$

Now the production rate PR_1 of machine M_1 is the multiplicative inverse of the average processing time of this machine times the probability $1-p_B = 1-(\mathbf{p}(N,B,1)+\mathbf{p}(N,B,2))$ of not being blocked:

$$PR_1=\frac{1-p_B}{\mathbf{E}[T_1]}=\frac{1-p_B}{(1-a_1)\frac{1}{\mu_{11}}+a_1\left(\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}\right)} \tag{21}$$

This leads to an equation for the probability of the machine being blocked:

$$p_B=1-PR_1\left((1-a_1)\frac{1}{\mu_{11}}+a_1\left(\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}\right)\right) \tag{22}$$

For the downstream machine the FRIT-equation

$$PR_2 = \frac{1-p_S}{\mathbf{E}[T_2]} = \frac{1-p_S}{(1-a_2)\frac{1}{\mu_{21}} + a_2\left(\frac{1}{\mu_{21}} + \frac{1}{\mu_{22}}\right)} \tag{23}$$

is similar and it also leads to a similar equation for the probability of the downstream machine not being starved:

$$p_S = 1 - PR_2\left((1-a_2)\frac{1}{\mu_{21}} + a_2\left(\frac{1}{\mu_{21}} + \frac{1}{\mu_{22}}\right)\right) \tag{24}$$

While equations (21) and (23) can be used to determine the production rate of a two-machine system, the equations (22) and (24) will later be used in a decomposition approach to analyze larger flow lines with more than two machines.

3.3 Derivation of the solution

In this section, we derive a specialized solution procedure similar to the one given in [22].

3.3.1 Analysis of internal states

Following the basic approach in Gershwin and Berman, we assume that the internal equations (5)–(8) have a solution of the form

$$\mathbf{p}[n,\alpha_1,\alpha_2] = \sum_{j=1}^{J} c_j \xi_j(n,\alpha_1,\alpha_2) = \sum_{j=1}^{J} c_j X_j^n Y_{1j}^{\alpha_1-1} Y_{2j}^{\alpha_2-1} \tag{25}$$

where c_j, X_j, Y_{1j}, and Y_{2j} are parameters to be determined. The analysis below is very similar to the one in [22] and [26, Sect. 3.2.4]. Replacing $\mathbf{p}(n,\alpha_1,\alpha_2)$ by $X_j^n Y_{1j}^{\alpha_1-1} Y_{2j}^{\alpha_2-1}$ in Equations (6), (7) and (8), we derive the following non-linear set of equations:

$$(\mu_{11}+\mu_{22})XY_2 = a_2\mu_{21}X + (1-a_1)\mu_{11}Y_2 + \mu_{12}Y_1Y_2 \tag{26}$$

$$(\mu_{12}+\mu_{21})Y_1 = a_1\mu_{11} + (1-a_2)\mu_{21}XY_1 + \mu_{22}XY_1Y_2 \tag{27}$$

$$(\mu_{12}+\mu_{22})Y_1Y_2 = a_2\mu_{21}Y_1 + a_1\mu_{11}Y_2 \tag{28}$$

Equations (26) and (27) are used to eliminate X. From the resulting equation and (28) we can next eliminate Y_2. A considerable algebraic effort leads to the following fourth degree equation in Y_1

$$a_2\mu_{21}(\mu_{12}Y_1 - a_1\mu_{11})(Y_1^3 + sY_1^2 + tY_1 + v) = 0 \tag{29}$$

with auxiliary variables s, t, v, and w defined as follows:

$$w = \mu_{21}(a_2\mu_{12} - \mu_{12} - \mu_{22}) \tag{30}$$

$$s = \frac{1}{w}\left(\mu_{11}\mu_{12} - \mu_{12}^2 + a_1\mu_{11}\mu_{21} + a_2\mu_{11}\mu_{21} - a_1a_2\mu_{11}\mu_{21} - \mu_{12}\mu_{21} + \mu_{11}\mu_{22} - \mu_{12}\mu_{22} - \mu_{21}\mu_{22}\right) \tag{31}$$

$$t = \frac{1}{w}\left(a_1\mu_{11}(-\mu_{11} + 2\mu_{12} + \mu_{21} + \mu_{22})\right) \tag{32}$$

$$v = \frac{1}{w} -(a_1^2\mu_{11}^2) \tag{33}$$

From the first term on the left side of Equation (29) we see that one solution to (29) is

$$Y_{11} = \frac{a_1\mu_{11}}{\mu_{12}} \tag{34}$$

Applying this result to Equation (28), we find

$$Y_{21} = \frac{a_2\mu_{21}}{\mu_{22}} \tag{35}$$

and from (34) and (35) in (26) or (27) we see that

$$X_1 = 1. \tag{36}$$

The remaining three solutions to (29) are[1]

$$Y_{12} = 2\sqrt{-\frac{a}{3}}\cos\left(\frac{\phi}{3}\right) - \frac{s}{3} \tag{37}$$

$$Y_{13} = 2\sqrt{-\frac{a}{3}}\cos\left(\frac{\phi}{3} + \frac{2\phi}{3}\right) - \frac{s}{3} \tag{38}$$

$$Y_{14} = 2\sqrt{-\frac{a}{3}}\cos\left(\frac{\phi}{3} + \frac{4\phi}{3}\right) - \frac{s}{3} \tag{39}$$

with auxiliary variables

$$a = \frac{1}{3}\left(3t - s^2\right) \tag{40}$$

$$b = \frac{1}{27}\left(2s^3 - 9st + 27v)\right) \tag{41}$$

$$\phi = \arccos\left(-\frac{b}{2\sqrt{\frac{-a^3}{27}}}\right) \tag{42}$$

The corresponding values of Y_{22}, Y_{22}, and Y_{24} are again determined via (28). The values of X_2, X_3, and X_4 are next computed from (26) or (27).

Since we have found four solutions to equations (26), (27), and (28), the general expression for the steady-state probabilities of the internal states is as follows

$$\mathbf{p}(n, \alpha_1, \alpha_2) = \sum_{j=1}^{4} c_j\xi_j(n, \alpha_1, \alpha_2) = \sum_{j=1}^{4} c_j X_j^n Y_{1j}^{\alpha_1 - 1} Y_{2j}^{\alpha_2 - 1} \tag{43}$$

where we still have to determine the parameters c_j.

[1] See [3, Sect. 2.4.2.3, p. 131]

3.3.2 Analysis of boundary states

There is a total of 12 boundary states in the model. The transition equations of four of them ($(1,2,1)$, $(1,2,2)$, $(N-1,1,2)$, and $(N-1,2,2)$) are of internal form (6) - (8), i.e. their steady-state probabilities can be computed from equation (43) even though they are boundary states.

Since $\mathbf{p}(1,2,1)$ and $\mathbf{p}(1,2,2)$ are given from (43), the corresponding equations (7) and (8) related to states $(1,2,1)$ and $(1,2,2)$ constitute a linear system of two equations in two unknowns $\mathbf{p}(1,1,1)$ and $\mathbf{p}(1,1,2)$ with the following solution:

$$\mathbf{p}(1,1,1) = \frac{(\mu_{12}+\mu_{21})\mathbf{p}(1,2,1) - (\mu_{21}-a_2\mu_{21})\mathbf{p}(2,2,1)}{a_1\mu_{11}} - \frac{\mu_{22}\mathbf{p}(2,2,2)}{a_1\mu_{11}} \tag{44}$$

$$\mathbf{p}(1,1,2) = \frac{(\mu_{12}+\mu_{22})\mathbf{p}(1,2,2) - a_2\mu_{21}\mathbf{p}(1,2,1)}{a_1\mu_{11}} \tag{45}$$

Given $\mathbf{p}(1,1,1)$, $\mathbf{p}(1,1,2)$, $\mathbf{p}(1,2,1)$, and $\mathbf{p}(1,2,2)$, Equations (1) and (2) can immediately be used to determine first $\mathbf{p}(0,1,S)$ and next $\mathbf{p}(0,2,S)$ (in this order).

The upper boundary steady-state probabilities are determined in exactly the same way as now states $(N-1,1,2)$ and $(N-1,2,2)$ are of internal form and we may compute $\mathbf{p}(N-1,1,2)$ and $\mathbf{p}(N-1,2,2)$ from (43), then solve (6) and (8) for $\mathbf{p}(N-1,1,1)$ and $\mathbf{p}(N-1,2,1)$ to find

$$\mathbf{p}(N-1,1,1) = \frac{(\mu_{11}+\mu_{22})\mathbf{p}(N-1,1,2) - (\mu_{11}-a_1\mu_{11})\mathbf{p}(N-2,1,2)}{a_2\mu_{21}} - \frac{\mu_{12}\mathbf{p}(N-2,2,2)}{a_2\mu_{21}} \tag{46}$$

$$\mathbf{p}(N-1,2,1) = \frac{(\mu_{12}+\mu_{22})\mathbf{p}(N-1,2,2) - a_1\mu_{11}\mathbf{p}(1,1,2)}{a_2\mu_{21}}. \tag{47}$$

Given $\mathbf{p}(N-1,1,1)$ and $\mathbf{p}(N-1,2,1)$, we can now (in this order) compute $\mathbf{p}(N,B,1)$ from equation (9) and finally $\mathbf{p}(N,B,2)$ from equation (10). Consider again the symmetry of upper and lower boundary values.

Since boundary states are now expressed in terms of internal states, and since internal states are of the form

$$\mathbf{p}(n,\alpha_1,\alpha_2) = \sum_{j=1}^{4} c_j \xi_j(n,\alpha_1,\alpha_2), \tag{48}$$

the equations for boundary states hold for each solution $\xi_j(n,\alpha_1,\alpha_2)$ of the equations for internal states. The equation (45) corresponding to state $(1,1,2)$, for example, leads to

$$\sum_{j=1}^{4} c_j \xi_j(1,1,2) = \frac{(\mu_{12}+\mu_{22})\sum_{j=1}^{4} c_j\xi_j(1,2,2)}{a_1\mu_{11}} - \frac{a_2\mu_{21}\sum_{j=1}^{4} c_j\xi_j(1,2,1)}{a_1\mu_{11}} \tag{49}$$

Similar equations can be found to determine the terms $\xi_j(n, \alpha_1, \alpha_2)$ for the other boundary state probabilities. The terms $\xi_j(n, \alpha_1, \alpha_2)$ corresponding to transient states are all zero.

Now all steady-state probabilities have been related to equation (43). What remains to be done is to find appropriate values of the coefficients c_j in (43).

3.3.3 Determination of coefficients c_j

To determine four coefficients $c_j, j = 1, ..., 4$, a linear system of four equations in the four unknowns c_j can be solved. The following four equations can be derived by inserting (43) into the conservation of flow equation (16), the two equations stating that for every transition from phase one to phase two there is one from phase two to phase one ((17) and (18)), and the condition (13) that all probabilities sum up to one:

Conservation of flow

$$\begin{aligned}
&\mu_{11}(1-a_1)\sum_{j=1}^{4} c_j\xi_j(0,1,S) + \mu_{12}\sum_{j=1}^{4} c_j\xi_j(0,2,S) + \\
&\sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}(\mu_{11}(1-a_1)\sum_{j=1}^{4} c_j\xi_j(n,1,\alpha_2) + \mu_{12}\sum_{j=1}^{4} c_j\xi_j(n,2,\alpha_2)) - \\
&\sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}(\mu_{21}(1-a_2)\sum_{j=1}^{4} c_j\xi_j(n,\alpha_1,1) - \mu_{22}\sum_{j=1}^{4} c_j\xi_j(n,\alpha_1,2)) - \\
&\mu_{21}(1-a_2)\sum_{j=1}^{4} c_j\xi_j(N,B,1) - \mu_{22}\sum_{j=1}^{4} c_j\xi_j(N,B,2) = 0
\end{aligned} \tag{50}$$

Rate of changes from phase one to 2 equals rate of changes from phase two to 1 at Maschine M_1

$$\begin{aligned}
&a_1\mu_{11}\left(\sum_{j=1}^{4} c_j\xi_j(0,1,S) + \sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}\sum_{j=1}^{4} c_j\xi_j(n,1,\alpha_2)\right) \\
&-\mu_{12}\left(\sum_{j=1}^{4} c_j\xi_j(0,2,S) + \sum_{n=1}^{N-1}\sum_{\alpha_2=1}^{2}\sum_{j=1}^{4} c_j\xi_j(n,2,\alpha_2)\right) = 0
\end{aligned} \tag{51}$$

Rate of changes from phase one to 2 equals rate of changes from phase two to 1 at Maschine M_2

$$\begin{aligned}
&a_2\mu_{21}\left(\sum_{j=1}^{4} c_j\xi_j(N,B,1) + \sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}\sum_{j=1}^{4} c_j\xi_j(n,\alpha_1,1)\right) \\
&-\mu_{22}\left(\sum_{j=1}^{4} c_j\xi_j(N,B,2) + \sum_{n=1}^{N-1}\sum_{\alpha_1=1}^{2}\sum_{j=1}^{4} c_j\xi_j(n,\alpha_1,2)\right) = 0
\end{aligned} \tag{52}$$

Probabilities sum up to one

$$\sum_{j=1}^{4} c_j \xi_j(0,1,S) + \sum_{j=1}^{4} c_j \xi_j(0,2,S) + \left(\sum_{n=1}^{N-1} \sum_{\alpha_1=1}^{2} \sum_{\alpha_2=1}^{2} \sum_{j=1}^{4} c_j \xi_j(n,\alpha_1,\alpha_2) \right) +$$

$$\sum_{j=1}^{4} c_j \xi_j(N,B,1) + \sum_{j=1}^{4} c_j \xi_j(N,B,2) = 1 \tag{53}$$

Note that the right hand side of the three of the four equations is zero. For this reason, it is relatively painless to solve this linear system of equations in the four unknowns $c_j, j = 1..4$ numerically.

3.4 The algorithm to determine steady-state probabilities and performance measures

The algorithm to compute the required steady-state probabilities $\mathbf{p}[n, \alpha_1, \alpha_2]$ and performance measures PR and $\overline{n}$ consists of the following steps:

1. Compute auxiliary variables w, s, t, v, a, b, and ϕ from (30)-(33) and (40)-(42). Compute Y_{11} from (34) and $Y_{12}...Y_{14}$ from (37)-(39). Compute $Y_{21}...Y_{24}$ from (28) and $X_1...X_4$ from (26) or (27).
2. Determine the coefficients $c_j, j = 1, ..., 4$ in Equation (43) by solving the linear system of equations given by (50)-(53).
3. Use the c_j from Step 2 to compute the required steady-state probabilities $\mathbf{p}(n, \alpha_1, \alpha_2)$ of states of internal form via (43) and those of the remaining boundary states as described in Section 3.3.2.
4. Determine performance measures. Determine the production rate from (14) or (15), the in-process inventory via

$$\bar{n} = \sum_{n=1}^{N-1} \sum_{\alpha_1=1}^{2} \sum_{\alpha_2=1}^{2} n\mathbf{p}(n,\alpha_1,\alpha_2) + N(\mathbf{p}(N,B,1) + \mathbf{p}(N,B,2) \tag{54}$$

and blocking and starvation probabilities p_B and p_S via

$$p_B = \mathbf{p}(N,B,1) + \mathbf{p}(N,B,2) \tag{55}$$

$$p_S = \mathbf{p}(0,1,S) + \mathbf{p}(0,2,S). \tag{56}$$

This algorithm proved to be numerically stable and it was used as a building block within the decomposition approach employed to analyze flow lines with more than two machines.

4 The decomposition approach

4.1 Derivation of decomposition equations

While it is possible to analyze a two-machine system exactly, the exact analysis of larger systems is practically impossible as the state space of the system explodes

very quickly. For this reason decomposition approaches are frequently used to analyze larger systems. The basic idea is to decompose a system with K machines and $K-1$ buffers into $K-1$ two-machine systems with virtual machines that mimic to an observer in the buffer the flow of material in and out of this buffer as it would be seen in the corresponding buffer of the real system. We followed the ideas presented in great detail in [21] to develop an iterative decomposition algorithm to analyze flow lines with more than two machines. However, some modifications were necessary which we will now briefly outline. While the models analyzed in [21] assumed unreliable machines and consequently lead to so-called *interruption-of-flow*- and *resumption-of-flow*-equations, we are studying a flow line with reliable machines which cannot fail. The machines in our system, however, change their phases of operation as described in Section 2. For this reason, we derived the following three types of decomposition equations:

- **Phase-One-to-Two(P1t2)-Equation:** This type of equation deals with the probability of the transition of the virtual machine from operating in its first phase to its second.
- **Phase-Two-to-One(P2t1)-Equation:** This type of equation deals with the probability of the transition of the virtual machine from operating in its second phase to its first.
- **Flow-Rate-Idle-Time(FRIT)-Equation:** This is a type of equation which relates the flow of material through a machine to its isolated production rate and its probability of being blocked and starved. This type of equation has also been used by Gershwin et al.

In the following we will briefly discuss the derivation of the parameters of the virtual machines.

The key to the derivation of the P1t2- and P2t1-equations is the definition of virtual machine states. We study a virtual two-machine line $L(i)$ which is related to the buffer between machines M_i and M_{i+1}. The virtual machines of line $L(i)$ are $M_u(i)$ (upstream of the buffer) and $M_d(i)$ (downstream of the buffer). We want to determine the parameters $a_u(i)$, $\mu_{u_1}(i)$, and $\mu_{u_2}(i)$ of the virtual machine $M_u(i)$ as well as the parameters $a_d(i)$, $\mu_{d_1}(i)$, and $\mu_{d_2}(i)$ of the virtual machine $M_d(i)$ in order to be able to use our two-machine model in Section 3 to determine performance measures for the flow line.

The upstream machine of a two-machine line is never starved (and the downstream machine is never blocked). We therefore assume that the virtual machine $M_u(i)$ is in phase one if the real machine M_i is processing a workpiece in phase one or when it is *waiting* for the next workpiece:

$$\{\alpha_u(i,t)=1\} \text{ iff } \big\{\{\alpha_i(t)=1\} \text{ or } \{\alpha_i(t)=S\}\big\} \tag{57}$$

Machine $M_u(i)$ is in phase two if M_i is in phase two

$$\{\alpha_u(i,t)=2\} \text{ iff } \{\alpha_i(t)=2\} \tag{58}$$

and it is blocked if M_i is blocked:

$$\{\alpha_u(i,t)=B\} \text{ iff } \{\alpha_i(t)=B\} \tag{59}$$

The definition of virtual machine states for machine $M_d(i)$ is symmetric: Machine $M_d(i)$ is in phase one if the machine M_{i+1} downstream of the buffer number i is in phase one or blocked:

$$\{\alpha_d(i,t)=1\} \text{ iff } \big\{\{\alpha_{i+1}(t)=1\} \text{ or } \{\alpha_{i+1}(t)=B\}\big\} \tag{60}$$

It is in phase two if machine M_{i+1} in the real system is in phase two

$$\{\alpha_d(i,t)=2\} \text{ iff } \{\alpha_{i+1}(t)=2\} \tag{61}$$

and starved if M_{i+1} is starved:

$$\{\alpha_d(i,t)=S\} \text{ iff } \{\alpha_{i+1}(t)=S\} \tag{62}$$

Phase-One-to-Two (P1t2)-Equation: To derive the P1t2-equation for machine $M_u(i)$, we ask for the probability of observing a transition of the virtual machine $M_u(i)$ from phase one to phase two. For this to happen, we have to observe a completion of phase one (with probability $\mu_u(i)\delta t$) *and* the process must enter the second phase, which happens with probability $a_u(i)$.

$$a_u(i)\mu_{u_1}(i)\delta t = \text{Prob}[\{\alpha_u(i,t+\delta t)=2\}|\{\alpha_u(i,t)=1\}] \tag{63}$$

The joint probability $a_u(i)\mu_{u_1}(i)\delta t$ can be related to a change in the machine states defined above if we insert the definitions of the virtual machine states given in (57) and (58):

$$\begin{aligned}
a_u(i)\mu_{u_1}(i)\delta t &= \text{Prob}[\{\alpha_u(i,t+\delta t)=2\}|\{\alpha_u(i,t)=1\}] \\
&= \text{Prob}[\{\alpha_i(t+\delta t)=2\}|\{\alpha_i(t)=1\} \text{ or } \{\alpha_i(t)=S\}] \\
&= \text{Prob}[\{\alpha_i(t+\delta t)=2\}|\{\alpha_i(t)=1\}] \cdot \\
&\quad \text{Prob}[\{\alpha_i(t)=1\}|\{\alpha_i(t)=1\} \text{ or } \{\alpha_i(t)=S\}] \\
&+ \text{Prob}[\{\alpha_i(t+\delta t)=2\}|\{\alpha_i(t)=S\}] \cdot \\
&\quad \text{Prob}[\{\alpha_i(t)=S\}|\{\alpha_i(t)=1\} \text{ or } \{\alpha_i(t)=S\}]
\end{aligned}$$

$$a_u(i)\mu_{u_1}(i) \approx a_i\mu_{i1}\text{Prob}[n(i-1,t)>0] \tag{64}$$

In the above derivation, the probability of machine M_i being in phase two at time $t+\delta t$, given that it was starved at time t, is zero. However, the rest of this derivation is still only a (possibly crude) approximation since the conditional probability $\text{Prob}[\{\alpha_i(t)=1\}|\{\alpha_i(t)=1\} \text{ or } \{\alpha_i(t)=S\}]$ of machine M_i being in phase one given that it is either in phase one or starved is simply approximated by the probability $\text{Prob}[n(i-1,t)>0]$ of machine M_i not being starved. This is crude since if it is not starved, in can still be in phase two or blocked. The reasoning behind this crude approximation is that if machine M_i is in phase one, we at least know that it cannot be starved and the probability of this state is related to the probability of machine $M_d(i-i)$ not being starved. While there is no stronger analytical justification for this substitution, it appears to work well in the numerical algorithm to be described below.

The basic approach to derive the probability of a transition from phase one to two at the virtual machine $M_d(i)$ is similar:

$$a_d(i)\mu_{d_1}(i)\delta t = \text{Prob}[\{\alpha_d(i,t+\delta t)=2\}|\{\alpha_d(i,t)=1\}] \tag{65}$$

We again insert the definition of virtual machine states and find

$$\begin{aligned} a_d(i)\mu_{d_1}(i)\delta t &= \text{Prob}[\{\alpha_d(i,t+\delta t)=2\}|\{\alpha_d(i,t)=1\}] \\ &= \text{Prob}[\{\alpha_{i+1}(t+\delta t)=2\}|\{\alpha_{i+1}(t)=1\} \text{ or } \{\alpha_{i+1}(t)=B\}] \\ &= \text{Prob}[\{\alpha_{i+1}(t+\delta t)=2\}|\{\alpha_{i+1}(t)=1\}] \cdot \\ &\quad \text{Prob}[\{\alpha_{i+1}(t)=1\}|\{\alpha_{i+1}(t)=1\} \text{ or } \{\alpha_{i+1}(t)=B\}] \\ &+ \text{Prob}[\{\alpha_{i+1}(t+\delta t)=2\}|\{\alpha_{i+1}(t)=B\}] \cdot \\ &\quad \text{Prob}[\{\alpha_{i+1}(t)=B\}|\{\alpha_{i+1}(t)=1\} \text{ or } \{\alpha_{i+1}(t)=B\}] \end{aligned}$$

$$a_d(i)\mu_{d_1}(i) \approx a_{i+1}\mu_{i+1,1}\text{Prob}[n(i+1,t)<N] \tag{66}$$

where we again substitute in an admittedly crude way the conditional probability of machine M_{i+1} being in phase one given that it is either in phase one or blocked by the probability $\text{Prob}[n(i+1,t)<N-1]$ of not being blocked.

Phase-Two-to-One (P2t1)-Equation: A transition of the virtual machine $M_u(i)$ from state two to state one or to being blocked can only occur if the real machine M_i completes the second phase of operation on a workpiece:

$$\begin{aligned} \mu_{u_2}(i)\delta t = \text{Prob}[&\{\alpha_u(i,t+\delta t)=1\} \text{ or } \{\alpha_u(i,t+\delta t)=B\}| \\ &\{\alpha_u(i,t)=2\}] \end{aligned} \tag{67}$$

If we insert the definition of the virtual machine states given in (57), (58) and (59), we get

$$\begin{aligned} \mu_{u_2}(i)\delta t &= \text{Prob}[\{\alpha_u(i,t+\delta t)=1\} \text{ or } \{\alpha_u(i,t+\delta t)=B\}|\{\alpha_u(i,t)=2\}] \\ &= \text{Prob}[\{\alpha_i(t+\delta t)=1\} \text{ or } \{\alpha_i(t+\delta t)=S\} \text{ or} \\ &\qquad \{\alpha_i(t+\delta t)=B\}|\{\alpha_i(t)=2\}] = \mu_{i2}\delta t \end{aligned} \tag{68}$$

and eventually

$$\mu_{u_2}(i) = \mu_{i2}. \tag{69}$$

In this derivation, the equation (68) holds because a part must complete its phase two (which happens with probability $\mu_{i2}\delta t$) in order for machine M_i to reach states 1, S, or B.

The reasoning for machine $M_d(i)$ is analogous:

$$\begin{aligned} \mu_{d_2}(i)\delta t = \text{Prob}[&\{\alpha_d(i,t+\delta t)=1\} \text{ or } \{\alpha_d(i,t+\delta t)=S\}| \\ &\{\alpha_d(i,t)=2\}] \end{aligned} \tag{70}$$

This leads to the following result:

$$\mu_{d_2}(i) = \mu_{i+1,2} \tag{71}$$

FRIT-Equation: The Flow-Rate-Idle-Time-Equation is the third type of decomposition equation. It states that the production rate PR_i of machine M_i in the real system is the probability that this machine is neither blocked nor starved divided by the average processing time of this machine:

$$PR_i = \frac{\text{prob}[\{n_i(t) > 0\} \text{ and } \{n_{i+1}(t) < N_{i+1}\}]}{(1-a_i)\frac{1}{\mu_{i1}} + a_i\left(\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}\right)} \tag{72}$$

The probability of machine M_i in the real system not being blocked or starved is approximated as follows:

$$\begin{aligned}&\text{prob}[\{n_i(t) > 0\} \text{ and } \{n_{i+1}(t) < N_{i+1}\}] \approx \\ &1 - \text{prob}[\{n_i(t) = 0\}] - \text{prob}[\{n_{i+1}(t) = N_{i+1}\}]\end{aligned} \tag{73}$$

This is only an approximation since M_i can both blocked and starved. Now these probabilities are unknown. However, we can use equations (22) and (24) from the two-machine model to approximate these quantities if we decompose the real system into a set of two-machine lines. This leads to the following equation:

$$\begin{aligned}PR_i &\approx \frac{1 - p_S(i) - p_B(i+1)}{(1-a_i)\frac{1}{\mu_{i1}} + a_i\left(\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}\right)} \\ &= \frac{PR_2(i)\left((1-a_d(i))\frac{1}{\mu_{d_1}(i)} + a_d(i)\left(\frac{1}{\mu_{d_1}(i)} + \frac{1}{\mu_{d_2}(i)}\right)\right)}{(1-a_i)\frac{1}{\mu_{i1}} + a_i\left(\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}\right)} \\ &+ \frac{PR_1(i+1)\left((1-a_u(i+1))\frac{1}{\mu_{u_1}(i+1)} + a_u(i+1)\left(\frac{1}{\mu_{u_1}(i)} + \frac{1}{\mu_{u_2}(i)}\right)\right)}{(1-a_i)\frac{1}{\mu_{i1}} + a_i\left(\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}\right)} \\ &- \frac{1}{(1-a_i)\frac{1}{\mu_{i1}} + a_i\left(\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}\right)}\end{aligned} \tag{74}$$

Because of the conservation-of-flow-equation, the following condition should be met by any decomposition of the original flow line into a set of two-machine lines:

$$PR_i = PR(i) = PR(i+1) \tag{75}$$

These equations will be used when we solve the decomposition equations.

4.2 Simultaneous solution of the decomposition equations

Equations (64), (69) and (74) can be solved simultaneously for the parameters $\mu_{u_1}(i)$, $\mu_{u_2}(i)$, and $a_u(i)$ of the upstream machine $M_u(i)$ related to line $L(i)$ of the decomposition to find:

$$a_u(i) = \Big(a_i\big(\mu_{d_1}(i-1)\mu_{i1}\mu_{d_2}(i-1)\mu_{i2} +$$

$$
\begin{aligned}
&a_i\mu_{d_1}(i-1)\mu_{i1}\mu_{d_2}(i-1)PR(i-1) + \\
&\mu_{d_1}(i-1)\mu_{d_2}(i-1)\mu_{i2}PR(i-1) - \\
&a_d(i-1)\mu_{d_1}(i-1)\mu_{i1}\mu_{i2}PR(i-1) - \\
&\mu_{i1}\mu_{d_2}(i-1)\mu_{i2}PR(i-1))(1-p_S(i-1))\Big) \cdot \\
&\frac{1}{\mu_{d_1}(i-1)\mu_{d_2}(i-1)PR(i-1)(\mu_{i2}+a_i\mu_{i1}(1-p_S(i-1)))}
\end{aligned} \tag{76}
$$

$$
\begin{aligned}
\mu_{u_1}(i) = &\Big(\mu_{d_1}(i-1)\mu_{i1}\mu_{d_2}(i-1)PR(i-1)(\mu_{i2}+a_i\mu_{i1}(1-p_S(i-1)))\Big) : \\
&\Big(\mu_{d_1}(i-1)\mu_{i1}\mu_{d_2}(i-1)\mu_{i2} + \\
&a_i\mu_{d_1}(i-1)\mu_{i1}\mu_{d_2}(i-1)PR(i-1) + \\
&\mu_{d_1}(i-1)\mu_{d_2}(i-1)\mu_{i2}PR(i-1) - \\
&a_d(i-1)\mu_{d_1}(i-1)\mu_{i1}\mu_{i2}PR(i-1) - \\
&\mu_{i1}\mu_{d_2}(i-1)\mu_{i2}PR(i-1)\Big)
\end{aligned} \tag{77}
$$

$$
\mu_{u_2}(i) = \mu_{i2} \tag{78}
$$

In Eqs. (76) and (77), the expressions PR_i and $PR(i)$ have been replaced by $PR(i-1)$ which is allowed because of conservation of flow. Now all three parameters of the virtual upstream machine $M_u(i)$ are expressed in terms of parameters of the real machine M_i or parameters or performance measures of line $L(i-1)$.

In exactly the same way the parameters for the downstream machine $M_d(i)$ can be determined:

$$
\begin{aligned}
a_d(i) = &\Big(a_{i+1}(\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{i+1,2}\mu_{u_2}(i+1) + \\
&a_{i+1}\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{u_2}(i+1)PR(i+1) + \\
&\mu_{u_1}(i+1)\mu_{i+1,2}\mu_{u_2}(i+1)PR(i+1) - \\
&a_u(i+1)\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{i+1,2}PR(i+1) - \\
&\mu_{i+1,1}\mu_{i+1,2}\mu_{u_2}(i+1)PR(i+1))(1-p_B(i+1))\Big) \cdot \\
&\frac{1}{\mu_{u_1}(i+1)\mu_{u_2}(i+1)PR(i+1)(\mu_{i+1,2}+a_{i+1}\mu_{i+1,1}(1-p_B(i+1)))}
\end{aligned} \tag{79}
$$

$$
\begin{aligned}
\mu_{d_1}(i) = &\Big(\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{u_2}(i+1)PR(i+1)\cdot \\
&(\mu_{i+1,2}+a_{i+1}\mu_{i+1,1}(1-p_B(i+1)))\Big) : \\
&\Big(\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{i+1,2}\mu_{u_2}(i+1) + \\
&a_{i+1}\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{u_2}(i+1)PR(i+1) + \\
&\mu_{u_1}(i+1)\mu_{i+1,2}\mu_{u_2}(i+1)PR(i+1) - \\
&a_u(i+1)\mu_{i+1,1}\mu_{u_1}(i+1)\mu_{i+1,2}PR(i+1) - \\
&\mu_{i+1,1}\mu_{i+1,2}\mu_{u_2}(i+1)PR(i+1)\Big)
\end{aligned} \tag{80}
$$

$$
\mu_{d_2}(i) = \mu_{i+1,2} \tag{81}
$$

Note that again all three parameters of the virtual downstream machine $M_d(i)$ are expressed in terms of parameters of the real machine M_{i+1} or parameters or performance measures of line $L(i+1)$.

4.3 Decomposition algorithm

We used an iterative algorithm to solve the decomposition equations numerically. No proof of convergence or accuracy can be given for this algorithm, as for many similar algorithms for flow line decomposition. It consists of the following steps:

1. **Initialization:** The initial parameters for the $M-1$ two-machine lines arising in the decomposition of a flow line with M machines are given as follows:

$$\begin{aligned}
a_u(i) &:= a_i, && i = 1, \ldots, M-1 \\
\mu_{u_1}(i) &:= \mu_{i1}, && i = 1, \ldots, M-1 \\
\mu_{u_2}(i) &:= \mu_{i2}, && i = 1, \ldots, M-1 \\
a_d(i) &:= a_{i+1}, && i = 1, \ldots, M-1 \\
\mu_{d_1}(i) &:= \mu_{i+1,1}, && i = 1, \ldots, M-1 \\
\mu_{d_2}(i) &:= \mu_{i+1,2}, && i = 1, \ldots, M-1
\end{aligned}$$

   If C_i is the number of buffer spaces between machines M_i and M_{i+1}, the extended buffer size $N(i)$ is

$$N(i) := C_i + 2. \tag{82}$$

   It includes the workspace at machine M_{i+1} (and also the one at M_i if this machine should be blocked). Given these parameters for the $M-1$ virtual two-machine lines, the production rate $PR(i)$, the average inventory level $\bar{n}(i)$ and the probabilities of blocking $p_B(i)$ and starvation $p_S(i)$ can be computed using the algorithm in Section 3.4.

2. **Iteration:**
   (a) *Downstream phase:* For line $l = 2, \ldots, M-1$, update parameters $a_u(l)$, $\mu_{u_1}(l)$, and $\mu_{u_2}(l)$ via equations (76), (77) and (78). Compute new performance measures $PR(l)$, $\bar{n}(l)$, $p_B(l)$, and $p_S(l)$.
   (b) *Upstream phase:* For line $l = M-2, \ldots, 1$, update parameters $a_d(l)$, $\mu_{d_1}(l)$, and $\mu_{d_2}(l)$ via equations (79), (80) and (81). Compute new performance measures $PR(l)$, $\bar{n}(l)$, $p_B(l)$, and $p_S(l)$.
   (c) *Accuracy check:* If the condition

$$\frac{\mid PR(l) - PR(l+1) \mid}{PR(l)} < 0.000001, \qquad l = 1, \ldots, M-1 \tag{83}$$

   holds, terminate the algorithm because the conservation of flow condition is met by the result of the decomposition. Otherwise, goto step 2a.

(The algorithm also terminates if no convergence should be reached after 50 iterations.)

Table 2. Three-stage system with general service times

Case	2	3	4	7
μ_1	0.5	0.5	0.5	0.5
μ_2	0.5	0.5	0.5	1.0
μ_3	0.5	0.5	0.5	0.5
c_1^2	0.5	0.8	2.0	0.6
c_2^2	0.5	0.8	2.0	0.6
c_3^2	0.5	0.8	2.0	0.6
Sim. PR	0.382	0.351	0.296	0.427
BLS-a (abs.)	0.384	0.347	0.272	0.441
BLS-a (rel.)	0.52%	−1.14%	−8.11%	3.28%
BLS-b (abs.)	0.381	0.349	0.282	0.429
BLS-b (rel.)	−0.26%	−0.57%	−4.73%	0.47%
CoxDC (abs.)	0.380	0.349	0.298	0.443
CoxDC (rel.)	−0.55%	−0.48%	0.61%	3.82%

5 Numerical results and conclusion

Since the analysis of the two-machine model is exact, the only interesting question with respect the the two-machine algorithm is if it is numerically stable. The results reported in [8] indicated that their method tended to have difficulties with larger buffer sizes. We did not observe such instabilities for the buffer sizes they considered (up to 100 buffer spaces).

In order to evaluate the accuracy of the decomposition algorithm, we compared it to results given in [9]. In these cases, the expected value and the squared coefficient of variation of the processing time for each machine was given. The Cox-2-distribution, however, has three parameters, so that one degree of freedom is left. We used the so-called "balanced-mean" two-phase Coxian distribution [10, p. 542] to match the problem data given in [9, p. 450-451].

Table 2 gives parameters and results for a three-stage system with general service times and one buffer space *between* adjacent machines. (In the paper by Buzacott et al. [9], the buffer space includes the workspace at the downstream machine. If there is just *one* buffer space between two adjacent machines, our general approach to determine steady-state probabilities as described in Section 3.4 cannot be applied since there are no "internal states" in terms of our two-machine model. However, such a system has only 10 states which are all either lower or upper boundary states. It is trivial to determine the steady-state probabilities and performance measures for such a tiny system and we therefore simply added the required algorithm to our decomposition approach in order to be able to deal with two-machine lines with just one buffer space between adjacent machines.) The entries "BLS-a" and "BLS-b" are related to two approaches described in [9], "Sim" denotes the simulation results and "CoxDC" the results from our approach. For the system in Table 2 our approach gives comparable results.

Table 3. Four-stage system with exponential service times

Case	1	2	3	4
μ_1	1.0	1.0	1.0	1.0
μ_2	1.1	1.2	1.5	2.0
μ_3	1.2	1.4	2.0	3.0
μ_4	1.3	1.6	2.5	4.0
Exact. PR	0.71	0.765	0.861	0.929
BLS-a (abs.)	0.689	0.746	0.85	0.925
BLS-a (rel.)	−2.96%	−2.48%	−1.28%	−0.43%
BLS-b (abs.)	0.7	0.756	0.855	0.927
BLS-b (rel.)	−1.41%	−1.18%	−0.70%	−0.22%
CoxDC (abs.)	0.712	0.767	0.862	0.930
CoxDC (rel.)	0.29%	0.24%	0.07%	0.09%

Table 4. Three-stage system with general service times

Case	1	2	3
μ_1	0.5	0.5	0.5
μ_2	0.5	0.5	0.5
μ_3	0.5	0.5	0.5
c_1^2	0.75	2.0	2.0
c_2^2	0.75	2.0	2.0
c_3^2	0.75	2.0	2.0
Sim. PR	0.385	0.322	0.360
BLS-a (abs.)	0.385	0.303	0.345
BLS-a (rel.)	0.00%	−5.90%	−4.17%
BLS-b (abs.)	0.385	0.312	0.349
BLS-b (rel.)	−0.00%	−3.11%	−3.06%
Altiok (abs.)	0.368	0.338	0.368
Altiok (rel.)	−4.42%	4.97%	2.22%
CoxDC (abs.)	0.386	0.327	0.360
CoxDC (rel.)	0.26%	1.40%	−0.10%

For the four-stage systems in Table 3 with exponential service times our approach is more accurate than the procedures proposed in [9]. There is again just one buffer space between adjacent machines in these cases.

Table 4 presents results for systems given by Altiok as cited in [9]. In all cases there are two buffer spaces between adjacent machines except for Case 3 with 9 buffer spaces between machines 2 and 3 (and two between machines 1 and 2). For these systems our approach outperforms the other methods.

Table 5. Eight-stage systems with general service times

Case	1	2	3	4	5	6	7	8
μ_1	1.0	1.0	1.0	1.0	1.2	1.2	1.2	1.2
μ_2	1.0	1.0	1.0	1.0	1.3	1.3	1.3	1.3
μ_3	1.0	1.0	1.0	1.0	1.1	1.1	1.1	1.1
μ_4	1.0	1.0	1.0	1.0	0.8	0.8	0.8	0.8
μ_5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
μ_6	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
μ_7	1.0	1.0	1.0	1.0	1.1	1.1	1.1	1.1
μ_8	1.0	1.0	1.0	1.0	0.9	0.9	0.9	0.9
c_1^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_2^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_3^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_4^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_5^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_6^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_7^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
c_8^2	0.5	0.5	2.0	2.0	0.5	0.5	2.0	2.0
Buffer sizes	1	10	1	10	1	10	1	10
Sim. PR	0.683	0.918	0.462	0.760	0.661	0.799	0.461	0.723
CoxDC (abs.)	0.688	0.923	0.494	0.784	0.663	0.799	0.486	0.736
CoxDC (rel.)	0.62%	0.54%	6.85%	3.26%	0.28%	0.03%	5.44%	1.77%

We finally study some eight-stage systems in Table 5. The numerical results indicate that quite often the proposed algorithm is rather accurate. However, in cases with a high degree of variability of the processing times (squared coefficient of variation of 2.0) and small buffer sizes (one buffer space between adjacent machines), the approximation quality deteriorates while the convergence of the algorithms still appears to be quick and reliable.

Given the numerical results we conclude that our decomposition approach can be used to analyze flow lines with general service times as long as these service time exhibit a squared coefficient of variation larger than 0.5.

References

1. Altiok T (1996) Performance analysis of manufacturing systems. Springer, Berlin Heidelberg New York
2. Artamonov G (1977) Productivity of a two-instrument discrete processing line in the presence of failures. Cybernetics 12: 464–468
3. Bronstein IN, Semendjajew KA (1983) Taschenbuch der Mathematik, 21st edn. Teubner, Leipzig
4. Burman MH (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
5. Buzacott JA (1967) Automatic transfer lines with buffer stocks. International Journal of Production Research 5(3): 183–200
6. Buzacott J (1972) The effect of station breakdowns and random processing times on the capacity of flow lines. AIIE Transactions 4: 308–312
7. Buzacott JA, Hanifin LE (1978) Models of automatic transfer lines with inventory banks – a review and comparison. AIIE Transactions 10(2): 197–207
8. Buzacott JA, Kostelski D (1987) Matrix-geometric and recursive algorithm solution of a two-stage unreliable flow line. IIE Transactions 19(4): 429–438
9. Buzacott JA, Liu XG, Shanthikumar JG (1995) Multistage flow line analysis with the stopped arrival queue model. IIE Transactions 27(4): 444–455
10. Buzacott JA, Shanthikumar JG (1993) Stochastic models of manufacturing systems. Prentice Hall, Englewood Cliffs, NJ
11. Buxey G, Slack N, Wild R (1973) Production flow line system design – a review. AIIE Transactions 5: 37–48
12. Choong Y, Gershwin SB (1987) A decomposition method for the approximate evaluation of capacitated transfer lines with unreliable machines and random processing times. IIE Transactions 19: 150–159
13. Dallery Y, David R, Xie XL (1988) An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers. IIE Transactions 20(3): 280–283
14. Dallery Y, David R, Xie XL (1989) Approximate analysis of transfer lines with unreliable machines and finite buffers. IEEE Transactions on Automatic Control 34(9): 943–953
15. Dallery Y, Gershwin SB (1992) Manufacturing flow line systems: a review of models and analytical results. Queuing Systems Theory and Applications 12(1–2): 3–94
16. Di Mascolo M, David R, Dallery Y (1991) Modeling and analysis of assembly systems with unreliable machines and finite buffers. IIE Transactions 23(4): 315–330
17. Gaver DP (1962) A waiting line with interrupted service, including priorities. Journal of the Royal Statistical Society 24: 73–90
18. Gershwin SB (1987) An efficient decomposition algorithm for the approximate evaluation of tandem queues with finite storage space and blocking. Operations Research 35: 291–305
19. Gershwin SB (1989) An efficient decomposition algorithm for unreliable tandem queueing systems with finite buffers. In: Perros G, Altiok T (eds) Queueing networks with blocking, pp 127–146. North Holland, Amsterdam
20. Gershwin SB (1991) Assembly/disassembly systems: An efficient decomposition algorithm for tree-structured networks. IIE Transactions 23(4): 302–314
21. Gershwin SB (1994) Manufacturing systems engineering. Prentice Hall, Englewood Cliffs, NJ

22. Gershwin SB, 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
23. Gershwin SB, 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, MA
24. Gershwin SB, Schick I (1983) Modeling and analysis of three-stage transfer lines with unreliable machines and finite buffers. Operations Research 31(2): 354–380
25. Helber S (1998) Decomposition of unreliable assembly/dissassembly networks with limited buffer capacity and random processing times. European Journal of Operational Research 109(1): 24–42
26. Helber S (1999) Performance analysis of flow lines with non-linear flow of material. Springer, Berlin Heidelberg New York
27. Hillier F, Boling RW (1967) Finite queues in series with exponential or Erlang service times – a numerical approach. Operations Research 16: 286–303
28. Koenigsberg E (1959) Production lines and internal storage – a review. Management Science 5: 410–433
29. Okamura K, Yamashina H (1977) Analysis of the effect of buffer storage capacity in transfer line systems. AIEE Transactions 9: 127–135
30. Papadopoulus HT, Heavey C, Browne J (1993) Queueing theory in manufacturing systems analysis and design. Chapman & Hall, London
31. Sastry BLN, Awate PG (1988) Analysis of a two-station flow line with machine processing subject to inspection and rework. Opsearch 25: 89–97
32. Sevast'yanov BA (1962) Influence of storage bin capacity on the average standstill time of a production line. Theory of Probability and Its Applications 7: 429–438
33. 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
34. Yeralan S, Muth EJ (1987) A general model of a production line with intermediate buffer and station breakdown. IIE Transactions 19(2): 130–139
35. Zimmern B (1956) Etudes de la propagation des arrêts aleatoires dans les chaines de production. Review Statististical Applications 4: 85–104

Performance evaluation of production lines with finite buffer capacity producing two different products

M. Colledani, A. Matta, and T. Tolio

Politecnico di Milano, Dipartimento di Meccanica, via Bonardi 9, 20133 Milano, Italy
(e-mail: tullio.tolio@polimi.it)

Abstract. This paper presents an approximate analytical method for the performance evaluation of a production line with finite buffer capacity, multiple failure modes and multiple part types. This paper presents a solution to a class of problems where flexible machines take different parts to process from distinct dedicated input buffers and deposit produced parts into distinct dedicated output buffers with finite capacity. This paper considers the case of two part types processed on the line, but the method can be extended to the case of n part types. Also, the solution is developed for deterministic processing times of the machines which are all identical and are assumed to be scaled to unity. The approach however is amenable of extension to the case of inhomogeneous deterministic processing times. The proposed method is based on the approximate evaluation of the performance of the k-machine line by the evaluation of 2(k-1) two-machine lines. An algorithm inspired by the DDX algorithm has been developed and the validation of the method has been carried out by means of testing and comparison with simulation.

Keywords: Flow lines – Performance evaluation – Multiple part types

1 Introduction

Given the increasing flexibility of manufacturing machines and assembly station it is rather frequent that more than one part type is produced on a single production line. Also, in automated systems, machines are normally connected by accumulating conveyors which act as finite capacity buffers. Existing analytical techniques do not allow the modelling of such systems; indeed classical analytical techniques allow the modelling of multiclass systems but do not consider finite capacity buffers while approximate analytical techniques developed to model transfer lines do not take into account different part types. This paper presents a solution procedure to a class of problems of this type where flexible machines take different parts to process from

Correspondence to: T. Tolio

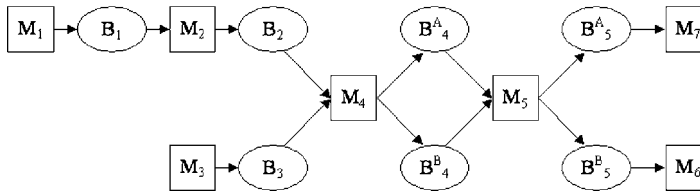


Fig. 1. Example of a system producing two part types

distinct dedicated input buffers and deposit produced parts into distinct dedicated output buffers with finite capacity. By dedicated input and output buffers we mean buffers that can store only one part type. The proposed solution is developed for the case of two part types, however the approach is amenable to extension to the multiple part type case. Also, the solution is developed for deterministic processing times of the machines which are all identical and are assumed to be scaled to unity. The approach however is amenable to extension to the case of inhomogeneous deterministic processing times.

A typical system of the proposed class is represented in Figure 1. In this case machines M_1, M_2, M_3, M_6 and M_7 are dedicated machines i.e. they can produce only one part type. On the contrary, machines M_4 and M_5 are flexible machines and can process both part types. The selection of which type of part to produce depends on the state of the system and on a dispatching rule. If the upstream buffer of one part type is empty or the downstream buffer is full, the machine will produce the other part type. If both the part types are either blocked or starved, the machine will not produce. If both the parts can be produced, than the machine will produce part type A with probability α_i^A and part type B with probability α_i^B. In the paper, systems formed only by flexible machines will be considered, but the proposed method in principle can be extended to the case of systems in which both flexible and dedicated machines are present, such that in Figure 1.

It is important to notice that the proposed system is quite different from assembly/disassembly systems [2, 1]. Indeed in assembly/disassembly systems assembly machines take contemporarily different parts from different input buffers to produce a single subassembly while disassembly machines from one subassembly produce contemporarily different components that are put into different buffers. In either case there is no selection of components to work on but they are all contemporarily involved in the process. On the contrary, in the described system a flexible machine selects a single component to work on.

The proposed system is also different from fork and join networks [3, 4]. Indeed in fork and join networks each machine can either take the input from different buffers to produce an undifferentiated product or take in as input an undifferentiated product and place it after processing in different buffers. On the contrary in the described system flexible machines take in as input different parts from different buffers and produce different products placed in the corresponding buffers. In other words, the identity of the part is not lost within the machine.

The problem presented in this paper has been originally stated by S.B. Gershwin and addressed by Nemec [6]. The original statement however considers a priority rule between the parts and therefore when both part types can be produced, the part

type with the highest priority is selected. This would correspond in our statement to the case of $\alpha^A = 1, \alpha^B = 0$. The solution approach adopted in Nemec is heavily dependent on the original problem statement because parts are treated differently depending on the priority. On the contrary, in the proposed approach all the parts are considered in the same way.

It is interesting to consider the fact that the described problem, which has been inspired by automated production system, is similar to other relevant problems that can be addressed with the same methodology. In particular it is interesting to consider the case of production networks where different enterprize cooperate to produce complex products. In this case each enterprize of the network can be modelled as a flexible machine while input and output storages can be modelled as buffers.

2 Assumptions and notations

In this paper we consider transfer lines composed of K machines in which two distinct part types (type A and type B) are processed in certain ratios. Both part types follow a linear path through the system since they are processed by all the M_i machines (with $i = 1, ..., K$), starting from the first one and finishing with the last machine after which they leave the system. Adjacent machines are separated by two different buffers B_i^A and B_i^B with limited capacities dedicated to temporally store parts of types A and B respectively. Buffer capacities between machines M_i and M_{i+1} are denoted with N_i^A and N_i^B for part types A and B respectively. Machine M_i of the system works part type A and part type B in the ratios α_i^A and α_i^B when it is not blocked or starved.

Machines are multiple failure mode machines, i.e. they are unreliable and can fail in F_i different modes as assumed in [7]; we denote with $p_{i,j}$ the probability of failure of machine M_i in mode j and with $r_{i,j}$ the probability of repair of machine M_i failed in mode j (with $j = 1, ..., F_i$).

A detailed list of the assumptions used in the proposed model is described in the following; assumptions regard the behavior of the machines and describe in particular how failures can occur and how machines select the part type to produce on the basis of blocking and starvation that characterize the part flow in the system.

- In the model, the flow of material through the system is approximated by a discrete time model.
- The first machine is never starved, i.e. there is an infinite number of pieces of both part types waiting to be processed by the system.
- The last machine is never blocked, i.e. there is an infinite space downstream the system where it is always possible to store pieces processed by the system.
- Blocking before service (BBS) is assumed for the machines.
- If buffer B_i^A (B_i^B) is full then machine M_i will process part type B (A) if possible.
- If buffer B_{i-1}^A (B_{i-1}^B) is empty then machine M_i will process part type B (A) if possible.

- If for a given machine both the upstream buffers are not empty and both the downstream buffers are not full the machine will produce a part of type A with probability α_i^A and a part of type B with probability α_i^B ($\alpha_i^A + \alpha_i^B = 1$).
- Operation dependent failures are assumed, i.e. machines can only fail if they are not down, not blocked or starved for both part type at the same time, and not contemporarily starved for part type A(B) and blocked for part type B(A).
- A given machine M_i can fail in F_i different failure modes.
- An operational machine can fail in only one of its failure modes.
- Mean time to failure (MTTF) and mean time to repair (MTTR) are geometrically distributed with average values of $1/p_{i,j}$ and $1/r_{i,j}$ respectively ($i = 1, ..., K$; $j = 1, ..., F_i$).

3 Outline of the method

The method evaluates the performance measures of the systems described in the previous section by using a generalization of the decomposition technique proposed in [7]. The method can also be used in principle with the decomposition technique proposed in [8]. The analyzed system is decomposed into $2(K-1)$ sets of two-machine lines that together represent the behavior of the system. Each two-machine line (building block) models the flow of one of the two part types in the system (Fig. 2). In other words the method creates a two-machine line for each buffer of the original line; each building block is composed of two pseudo machines and one intermediary buffer. The upstream pseudo machine represents the behavior of the portion of the system that precedes, in the original line, the corresponding buffer considered in the building block. In the same way, the downstream pseudo machine represents the behavior of the portion of the system that follows, in the original line, the corresponding buffer considered in the building block. The idea is to analyze simple building blocks, easy to study with existing techniques, instead of the complex original system. In such a way the complexity of the analysis is reduced to study several two-machine lines instead of a long production line. However, the different two-machine lines are not independent and have to be analyzed by means of decomposition equations. To do this, the parameters of the pseudo machines are calculated so that the flow of parts through the buffers of the decomposed systems closely matches the flow through the corresponding buffers of the original line.

Therefore, for buffers B_i^A and B_i^B of the original line, two building blocks (Fig. 2) are created. The first building block models the flow of type A parts and is composed of the upstream pseudo-machine $M^{U(A)}(i)$, the downstream pseudo-machine $M^{D(A)}(i)$ and the buffer $B^A(i)$. These two pseudo-machines together with the buffer form the building block $a(i)$. The second building block models the flow of type B parts and is composed of the upstream pseudo-machine $M^{U(B)}(i)$, the downstream pseudo-machine $M^{D(B)}(i)$ and the buffer $B^B(i)$. These two pseudo-machines together with the buffer form the building block $b(i)$. To model the interruptions of flow through the buffers of the original line, failure probabilities of different modes are associated to each pseudo-machine. In the following we will consider the case of the upstream pseudo-machines. A similar reasoning applies to the downstream pseudo-machines.

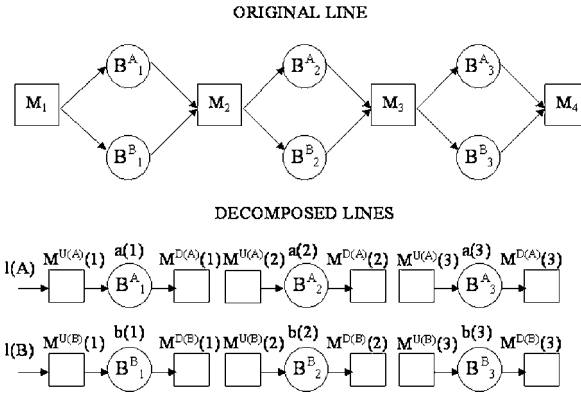


Fig. 2. Decomposition of the original line

Interruptions of flow due to a failure of the machine M_i of the original line are modelled assigning to the upstream pseudo-machines local failure modes with probabilities of failure $p_{i,f_i}^{U(A)}$ and $p_{i,f_i}^{U(B)}$ and probabilities of repair r_{i,f_i} (i.e. the same as the ones in the original line, for both part types), with $f_i = 1...F_i$. It must be noticed that, in multiple product lines, the probabilities of failure in local mode are not the same as the ones in the original flexible machine. They must be increased, considering the probability of the original machine of failing while producing the other part type. In fact, given the presence of two part types, a machine, even if is starved or blocked under the point of view of a given part type, can produce the other part type and can fail while producing that part type. Therefore, the probabilities of local failures must be adjusted to take into account this situation.

To mimic the interruptions of flow due to starvation, remote failure modes are introduced and assigned to the upstream pseudo-machines of the building blocks, namely $M^{U(A)}(i)$ and $M^{U(B)}(i)$. These remote failures have probabilities $p_{j,f_j}^{U(A)}$ and $p_{j,f_j}^{U(B)}$ and probabilities of repair $r_{j,f_j}^{U(A)}$ and $r_{j,f_j}^{U(B)}$ where $j = 1...i-1$, indicates the machines of the original line that actually failed (and are therefore responsible for the starvation) and $f_j = 1...F_j$ indicates the failure modes in which that machines failed. For these remote failure modes, we assume that the repair probabilities are identical to the repair probabilities of the machine of the original line that actually failed. On the other hand, the probability of failure for these remote modes are not known and must be evaluated by using decomposition equations.

The described failure modes follow the approach described in [7] to predict the performance of a transfer line producing only one part type.

To model the interactions between the parts competing for the same machines, in addition to the described failure modes, a new failure mode has been introduced and assigned to each pseudo-machine of the building blocks. This new failure mode has been called *competition failure* and mimics the situation in which a machine does not produce a given part type because it is busy producing the other part type. This new failure mode has probability of failure $p_{i,F_i+1}^{U(A)}$, $p_{i,F_i+1}^{U(B)}$ and probability of repair $r_{i,F_i+1}^{U(A)}$, $r_{i,F_i+1}^{U(B)}$ for part types A and B respectively.

In order to estimate failure and repair probabilities of the competition failure and to adjust local failure probabilities, it is necessary to study in detail all the states of each machine M_i of the original line, producing two part types.

Therefore, the solution approach is based on the analysis of all the states in which the machine M_i of the original line can be and on the solution of the Markov chain representing this machine. In this Markov chain, some transition probabilities are not known, however, the probabilities of starvation and blocking states of this Markov chain can be derived by studying the probabilities of the upstream buffers being empty and the probabilities of the downstream buffer being full. Indeed, by using the decomposition of the original line into building blocks, the flow of material through the buffers in the decomposed lines approximates the flow of material through the buffers in the original line. Therefore, by means of decomposition equations which are a generalization of the ones derived in [7], probabilities of these states can be calculated. Therefore, at the end, it is possible to solve a linear system of equations which allows the evaluation of both the unknown transition probabilities and the probabilities of all the states of the Markov chain.

The probabilities obtained for the various states of the flexible machine M_i are then used to build two separate models, one for each upstream pseudo-machine of the two building blocks ($M^{U(A)}(i)$, $M^{U(B)}(i)$). By studying these two models it is then possible to calculate the local failure parameters $p_{i,f_i}^{U(A)}$ and $p_{i,f_i}^{U(B)}$, considering the possibility for each machine of going down due to a failure occurred while processing the other part. In addition, it is possible to find the probabilities of failure and repair of the competition failures $p_{i,F_i+1}^{U(A)}$ and $p_{i,F_i+1}^{U(B)}$. These parameters completely define the pseudo-machines and allow in turn to evaluate the building blocks.

In Figure 3 the simplified scheme of the proposed method is presented. In particular it represents one single iteration of the algorithm, while studying upstream pseudo-machines.

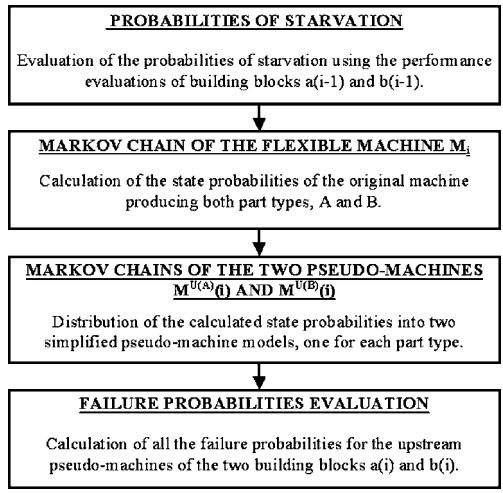


Fig. 3. Outline of the method

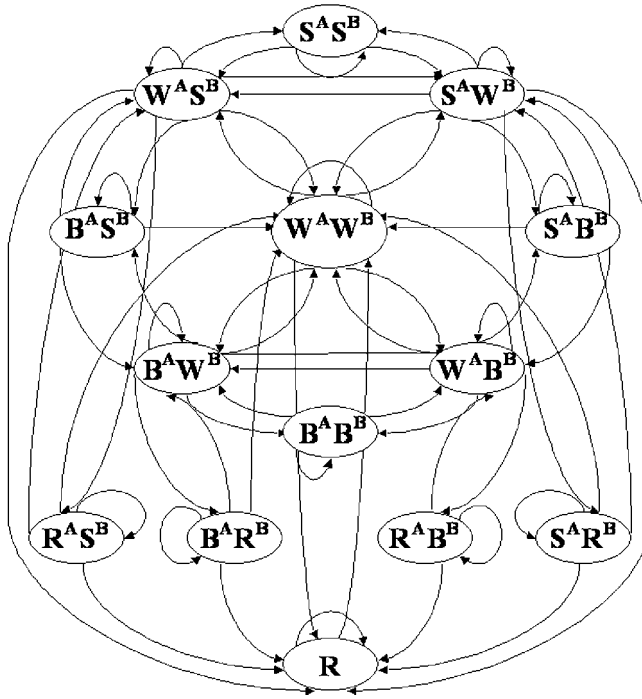


Fig. 4. Markov chain of the flexible machine M_i

4 Detailed description of the method

4.1 Macro states of the original machine

The Markov chain of the machine M_i of the original line is presented in Figure 4. To simplify the picture, all the states of the same type are grouped into a unique aggregate state, without considering different failure modes. Each aggregate state is defined by two state indicators, one referred to part type A and the other referred to part type B. Each state indicator can assume four values that are, for part type A: working (W^A), down in local mode (R^A), starved (S^A), if the upstream buffer dedicated to the storage of product A is empty, and blocked (B^A), if the downstream buffer dedicated to the storage of part type A is full. The same can be written for part type B. In total there are 16 possible aggregate states. For each aggregate state, the probability is obtained by adding up the probabilities of all the states of the same type. For example, the probability of the aggregate state named $W^A R^B$ has been obtained by adding up all the probabilities of the $W^A R_i^B$ states, one for each failure mode of the machine, i.e. $\pi(W^A R^B) = \sum_{i=1}^{F_i} \pi(W^A R_i^B)$.

Obviously, while in the picture we consider the aggregate states, in writing the equations it is important to distinguish all the different failure modes, to correctly evaluate the state probabilities. It must be noticed that machine M_i cannot be both working a part type while being down in local mode for the other part type, therefore

the states of type $W^A R^B$, $R^A W^B$ are not feasible and are not represented in the picture. Also the aggregate state $R^A R^B$ represents a situation where the machine is down and therefore cannot produce either A or B. We call this aggregate state pure local down state and we rename it R. Finally the state $W^A W^B$ represents a state where, for both part types, no local failures, starvation or blocking are present, therefore the original machine can produce either A or B.

In the following, some key characteristics of the Markov chain of the machine M_i (Fig. 4) are discussed:

- If the machine is in a state of type $W^A S^B$ and, while producing part type A (part type B cannot be produced because machine is starved for that part), it fails, it goes in state of type $R^A S^B$. This means that the machine is both down in local mode and starved. From this state it can go either to a pure local down state R if one part is stored in upstream buffer B^B_{i-1} or back to $W^A S^B$ if the local failure is repaired or to $W^A W^B$ if both the local failure is repaired and one part is stored in the upstream buffer B^B_{i-1}. A similar reasoning applies to the states of type $W^A B^B$, $S^A W^B$, $B^A W^B$.
- If the machine is in pure local down state, R, by repairing the local failure it always enters the $W^A W^B$ state.
- When the machine is in state $W^A W^B$ it can process A or B depending on the processing rate α^A_i and α^B_i ($\alpha^A_i + \alpha^B_i = 1$). Therefore from state $W^A W^B$, since only one of the two part types is produced, it is not possible to go to states of type $B^A B^B, B^A S^B, S^A B^B, S^A S^B$ (because if a part type is not produced it is not possible to have blocking or starvation for that part type).
- During a time interval a given machine of the line can at most process one part; therefore it is impossible to move from states of type $S^A S^B$ or $B^A B^B$ to state $W^A W^B$.

As already mentioned, in Figure 4, to simplify the diagram, all the states of the same type are grouped into a unique state without considering different failure modes. The probability of these 14 aggregate states is therefore the sum of the probabilities of the disaggregated states considering all the failure modes. It must be noticed that, in the Markov chain of the original machine, the competition failure is not considered, because this machine is able to produce both part types, as assumed in the previous section.

It must also be noticed that in this Markov chain not all the transition probabilities are known. Indeed the values of $p^{S(A)}_{j,f_j}$, $p^{S(B)}_{j,f_j}$ and $p^{B(A)}_{k,f_k}$, $p^{B(B)}_{k,f_k}$ cannot be derived directly from the original line and therefore they must be found using appropriate equations. In the following the 14 sets of equations required to evaluate the probabilities of the various states, plus the equations required to evaluate the unknown transition probabilities are provided. The approach used to derive the following equations deals with the idea that in the decomposed lines, as they have been defined, each building block mimics the flow of material through one buffer of the original line. Therefore, the macro state probabilities of the machine M_i must be coherent with probabilities of building blocks $a(i-1)$, $b(i-1)$, $a(i)$ and $b(i)$.

In the Markov chain, the sum of all the state probabilities must be equal to unity. Therefore it is possible to write the normalization equation for the model:

$$\sum All\ States\ = 1 \tag{1}$$

Since buffers $B^A(i-1)$ and $B^B(i-1)$ of the decomposed lines are equal to buffers B^A_{i-1} and B^B_{i-1} of the original line, the probability of starvation of machine M_i has to be equal to the one that derives from upstream building blocks $a(i-1)$ and $b(i-1)$.

$$\sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k+1}\pi(S^A_{j,f_j}S^B_{k,f_k}) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1}\pi(S^A_{j,f_j}B^B_{k,f_k}) \tag{2}$$

$$+\pi(W^B S^A_{j,f_j}) + \sum_{f_i=1}^{F_i}\pi(R^B_{i,f_i}S^A_{j,f_j}) = Ps^A_{j,f_j}(i-1)$$

$$j = 1,...,i-1, f_j = 1,...,F_j+1$$

$$\sum_{j=1}^{i-1}\sum_{f_j=1}^{F_j+1}\pi(S^A_{j,f_j}S^B_{k,f_k}) + \sum_{j=i+1}^{K}\sum_{f_j=1}^{F_j+1}\pi(B^A_{j,f_j}S^B_{k,f_k}) \tag{3}$$

$$+\pi(W^A S^B_{k,f_k}) + \sum_{f_i=1}^{F_i}\pi(R^A_{i,f_i}S^B_{k,f_k}) = Ps^B_{k,f_k}(i-1)$$

$$k = 1...i-1, f_k = 1...F_k+1$$

Where $\pi(X)$ is the steady state probability of state X.
Since buffers $B^A(i)$ and $B^B(i)$ of the decomposed lines are equal to buffers B^A_i and B^B_i of the original line, the probability of blocking of machine M_i has to be equal to that of downstream building blocks $a(i)$ and $b(i)$.

$$\sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k+1}\pi(S^B_{k,f_k}B^A_{j,f_j}) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1}\pi(B^A_{j,f_j}B^B_{k,f_k}) \tag{4}$$

$$+\sum_{f_i=1}^{F_i}\pi(R^B_{i,f_i}B^A_{j,f_j}) + \pi(W^B B^A_{j,f_j}) = Pb^A_{j,f_j}(i)$$

$$j = i+1...K; f_j = 1...F_j+1$$

$$\sum_{j=1}^{i-1}\sum_{f_j=1}^{F_j+1}\pi(S^A_{j,f_j}B^B_{k,f_k}) + \sum_{j=i+1}^{K}\sum_{f_j=1}^{F_j+1}\pi(B^A_{j,f_j}B^B_{k,f_k}) \tag{5}$$

$$+\sum_{f_i=1}^{F_i}\pi(R^A_{i,f_i}B^B_{k,f_k}) + \pi(W^A B^B_{k,f_k}) = Pb^B_{k,f_k}(i)$$

$$k = i+1...K; f_k = 1...F_k+1$$

Considering the states of type $R^A S^B$, $R^A B^B$, $R^B S^A$, $R^B B^A$, we can write node equations balancing the probability of entering these states with the probability of exiting the same states.

$$\pi(W^A S^B_{j,f_j})p_{i,f_i}(1 - r^{S(B)}_{j,f_j}) \tag{6}$$

$$= \pi(R^A_{i,f_i} S^B_{j,f_j}) r^{S(B)}_{j,f_j} + \pi(R^A_{i,f_i} S^B_{j,f_j}) r_{i,f_i} (1 - r^{S(B)}_{j,f_j})$$
$$f_i = 1...F_i, \quad j = 1...i-1, \quad f_j = 1...F_j + 1$$

$$\pi(W^B S^A_{j,f_j}) p_{i,f_i} (1 - r^{S(A)}_{j,f_j}) \tag{7}$$
$$= \pi(R^B_{i,f_i} S^A_{j,f_j}) r^{S(A)}_{j,f_j} + \pi(R^B_{i,f_i} S^A_{j,f_j}) r_{i,f_i} (1 - r^{S(A)}_{j,f_j})$$
$$f_i = 1...F_i, \quad j = 1...i-1, \quad f_j = 1...F_j + 1$$

$$\pi(W^A B^B_{k,f_k}) p_{i,f_i} (1 - r^{B(B)}_{k,f_k}) \tag{8}$$
$$= \pi(R^A_{i,f_i} B^B_{k,f_k}) r^{B(B)}_{k,f_k} + \pi(R^A_{i,f_i} B^B_{k,f_k}) r_{i,f_i} (1 - r^{B(B)}_{k,f_k})$$
$$f_i = 1...F_i, \quad k = i+1...K, \quad f_k = 1...F_k + 1$$

$$\pi(W^B B^A_{k,f_k}) p_{i,f_i} (1 - r^{B(A)}_{k,f_k}) \tag{9}$$
$$= \pi(R^B_{i,f_i} B^A_{k,f_k}) r^{B(A)}_{k,f_k} + \pi(R^B_{i,f_i} B^A_{k,f_k}) r_{i,f_i} (1 - r^{B(A)}_{k,f_k})$$
$$f_i = 1...F_i, \quad k = i+1...K, \quad f_k = 1...F_k + 1$$

Considering the states of type $S^A S^B$, $S^A B^B$, $B^A S^B$, $B^A B^B$ we can write node equations balancing the probability of entering these states with the probability of leaving the same states.

$$\pi(W^A S^B_{k,f_k}) p^{S(A)}_{j,f_j} (1 - r^{S(B)}_{k,f_k}) + \pi(W^B S^A_{j,f_j}) p^{S(B)}_{k,f_k} (1 - r^{S(A)}_{j,f_j}) \tag{10}$$
$$= \pi(S^A_{j,f_j} S^B_{k,f_k}) (r^{S(A)}_{j,f_j} + r^{S(B)}_{k,f_k})$$
$$j = 1...i-1; \; f_1 = 1...F_j + 1; \; k = 1...i-1; \; f_k = 1...F_k + 1$$

$$\pi(W^A B^B_{k,f_k}) p^{B(A)}_{j,f_j} (1 - r^{B(B)}_{k,f_k}) + \pi(W^B B^A_{j,f_1}) p^{B(B)}_{k,f_k} (1 - r^{B(A)}_{j,f_j}) \tag{11}$$
$$= \pi(B^A_{j,f_j} B^B_{k,f_k}) (r^{B(A)}_{j,f_j} + r^{B(B)}_{k,f_k})$$
$$j = i+1...K; \; f_j = 1...F_j + 1; \; k = i+1...K; \; f_k = 1...F_k + 1$$

$$\pi(W^A B^B_{k,f_k}) p^{S(A)}_{j,f_j} (1 - r^{B(B)}_{k,f_k}) + \pi(W^B S^A_{j,f_j}) p^{B(B)}_{k,f_k} (1 - r^{S(A)}_{j,f_j}) \tag{12}$$
$$= \pi(S^A_{j,f_j} B^B_{k,f_k}) (r^{S(A)}_{j,f_j} + r^{B(B)}_{k,f_k} - r^{S(A)}_{j,f_j} r^{B(B)}_{k,f_k})$$
$$j = 1...i-1; \; f_j = 1...F_j + 1; \; k = i+1...K; \; f_k = 1...F_k + 1$$

$$\pi(W^A S^B_{k,f_k}) p^{B(A)}_{j,f_j} (1 - r^{S(B)}_{k,f_k}) + \pi(W^B B^A_{j,f_j}) p^{S(B)}_{k,f_k} (1 - r^{B(A)}_{j,f_j}) \tag{13}$$
$$= \pi(S^B_{k,f_k} B^A_{j,f_j}) (r^{B(A)}_{j,f_j} + r^{S(B)}_{k,f_k} - r^{B(A)}_{j,f_j} r^{S(B)}_{k,f_k})$$
$$j = i+1...K; \; f_j = 1...F_j + 1; \; k = 1...i-1; \; f_k = 1...F_k + 1$$

Considering the set of states of type R , $R^A S^B$, $R^A B^B$, $R^B S^A$, $R^B B^A$, we can write equations balancing the probability of entering this set of states with the probability of leaving this set of states.

$$r_{i,f_i} (\pi(R_{i,f_i}) + \sum_{j=1}^{i-1} \sum_{f_j=1}^{F_j} \pi(R^B_{i,f_i} S^A_{j,f_j}) \tag{14}$$
$$+ \sum_{k=1}^{i-1} \sum_{f_k=1}^{F_k} \pi(R^A_{i,f_i} S^B_{k,f_k}) + \sum_{k=i+1}^{K} \sum_{f_k=1}^{F_k+1} \pi(R^A_{i,f_i} B^B_{k,f_k}) +$$

$$+\sum_{j=i+1}^{K}\sum_{f_j=1}^{F_j+1}\pi(R^B_{i,f_i}B^A_{j,f_j}))=p_{i,f_i}(\pi(W^AW^B)+\sum_{j=1}^{i-1}\sum_{f_1j=1}^{F_j+1}\pi(W^BS^A_{j,f_j})+$$
$$+\sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k+1}\pi(W^AS^B_{k,f_k})+\sum_{j=i+1}^{K}\sum_{f_j=1}^{F_j+1}\pi(W^BB^A_{j,f_j})$$
$$+\sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1}\pi(W^AB^B_{k,f_k}))\qquad f_i=1...F_i$$

In order to calculate the unknown transition probabilities, we first write the node equation for nodes of type W^AS^B, W^BS^A, W^AB^B and W^BB^A and then, after some manipulation, we obtain:

$$p^{S(B)}_{j,f_j}=\frac{Ps^B_{j,f_j}(i-1)}{E^B(i-1)}r^{S(B)}_{j,f_j}$$
$$p^{S(A)}_{j,f_j}=\frac{Ps^A_{j,f_j}(i-1)}{E^A(i-1)}r^{S(A)}_{j,f_j}\qquad j=1...i-1;f_j=1...F_j+1 \tag{15}$$
$$p^{B(B)}_{k,f_k}=\frac{Pb^B_{k,f_k}(i)}{E^B(i)}r^{B(B)}_{k,f_k}$$
$$p^{B(A)}_{k,f_k}=\frac{Pb^A_{k,f_k}(i)}{E^A(i)}r^{B(A)}_{k,f_k}\qquad k=i+1...K;f_k=1...F_k+1 \tag{16}$$

where $E^A(i)$ and $E^B(i)$ are the average production rates of the building blocks $a(i)$ and $b(i)$ respectively, $i=1,...,K-1$.
Repair probabilities of these failures are supposed to be equal to those local failures of the machines of the original line responsible for starvation and blocking.
Considering the expression of the efficiency in isolation e_i of the flexible machine M_i of the original line and the expression of the efficiency E_i of that machine related to the presence of buffers, it is simple to demonstrate that from normalization equation (1) it is possible to derive two conservation of flow equations, one for each product type:

$$\pi(W^AW^B)\alpha^A_i+\sum_{j=1}^{i-1}\sum_{f_j=1}^{F_j+1}\pi(W^AS^B_{j,f_j})$$
$$+\sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1}\pi(W^AB^B_{k,f_k})=E^A(i-1) \tag{17}$$
$$\pi(W^AW^B)\alpha^B_i+\sum_{j=1}^{i-1}\sum_{f_j=1}^{F_j+1}\pi(W^BS^A_{j,f_j})$$
$$+\sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1}\pi(W^BB^A_{k,f_k})=E^B(i-1) \tag{18}$$

4.2 Pseudo-machine models

Once the probabilities of the various states of the flexible machine M_i of the original line are obtained, it is possible to build two models (Fig. 5), one for each pseudo-machine $M^{U(A)}(i)$ and $M^{U(B)}(i)$. This results in two five state models. The probability of each state is obtained by adding up the values calculated in the previous Markov chain.

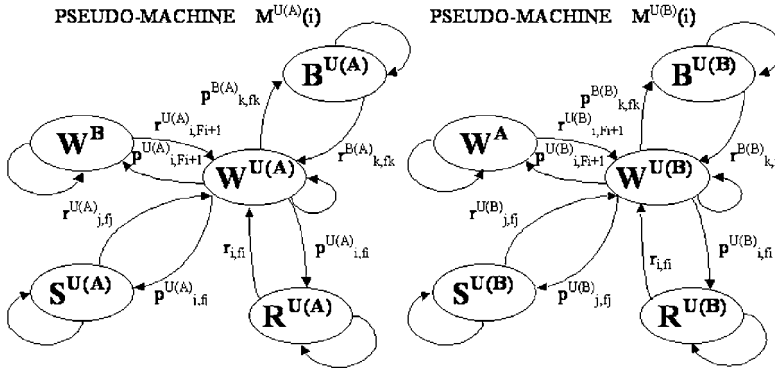


Fig. 5. Five state models for pseudo-machines $M^{U(A)}(i)$ and $M^{U(B)}(i)$

For the pseudo-machines $M^{U(A)}(i)$, we have:

$$\pi(W^{U(A)}) = \pi(W^A W^B)\alpha_i^A + \sum_{j=1}^{i-1}\sum_{f_j=1}^{F_j+1} \pi(W^A S_{j,f_j}^B) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1} \pi(W^A B_{k,f_k}^B) \tag{19}$$

$$\pi(S_{j,f_j}^{U(A)}) = \sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k+1} \pi(S_{j,f_j}^A S_{k,f_k}^B) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1} \pi(S_{j,f_j}^A B_{k,f_k}^B) + \pi(W^B S_{j,f_j}^A) + \sum_{f_i=1}^{F_i} \pi(R_{i,f_i}^B S_{j,f_j}^A) \tag{20}$$

$$j = 1, ..., i-1 \quad f_j = 1, ..., F_j$$

$$\pi(R_{i,f_i}^{U(A)}) = \pi(R_{i,f_i}) + \sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k} \pi(R_{i,f_i}^A S_{k,f_k}^B) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1} \pi(R_{i,f_i}^A B_{k,f_k}^B) \qquad f_i = 1, ..., F_i \tag{21}$$

$$\pi(B_{j,f_j}^{U(A)}) = \sum_{k=1}^{i-1}\sum_{f_k=1}^{F_k+1} \pi(S_{k,f_k}^B B_{j,f_j}^A) + \sum_{k=i+1}^{K}\sum_{f_k=1}^{F_k+1} \pi(B_{j,f_j}^A B_{k,f_k}^B)$$

$$+\sum_{f_i=1}^{F_i}\pi(R^B_{i,f_i}B^A_{j,f_j})+\pi(W^BB^A_{j,f_j}) \tag{22}$$

$$j=i+1,...,K \quad f_j=1,...,F_j$$

$$\pi(W^B) \quad =\pi(W^AW^B)\alpha^B_i \tag{23}$$

The state $W^{U(A)}$ represents the state in which the original machine M_i works part type A, that is the up state for the pseudo-machine $M^{U(A)}(i)$. The state W^B represents the state in which the original machine M_i works part type B even if it could work both part type, that is a down state for the pseudo-machine $M^{U(A)}(i)$. The same can be written for pseudo machine $M^{U(B)}(i)$.
Having these two approximate models, one for each pseudo-machine, and knowing all the state probabilities, it is possible to calculate new local failure probabilities and remote failure probabilities for the two pseudo-machines. It must be noticed that, for these machines, the probability of entering into local failure is higher than the one of the corresponding machine in the original line because we take into account the probability of failing while producing the other part type.
We can evaluate local failure parameters using balancing equation of nodes $R^{U(A)}$ and $R^{U(B)}$:

$$p^{U(A)}_{i,f_i}(i)=\frac{\pi(R^{U(A)})}{\pi(W^{U(A)})}r_{i,f_i}=\frac{\pi(R^{U(A)})}{E^A(i-1)}r_{i,f_i}\ f_i=1...F_i \tag{24}$$

$$p^{U(B)}_{i,f_i}(i)=\frac{\pi(R^{U(B)})}{\pi(W^{U(B)})}r_{i,f_i}=\frac{\pi(R^{U(B)})}{E^B(i-1)}r_{i,f_i}\ f_i=1...F_i \tag{25}$$

As originally proposed in [7], we introduce remote failure for the upstream pseudo-machine, to mimic starvation. So it is possible to write:

$$p^{U(A)}_{j,f_j}(i)=p^{S(A)}_{j,f_j} \qquad p^{U(B)}_{j,f_j}(i)=p^{S(B)}_{j,f_j} \tag{26}$$

$$j=1...i-1;f_j=1...F_j+1$$

$$r^{U(A)}_{j,f_j}(i)=r^{S(A)}_{j,f_j}=r_{j,f_j} \qquad r^{U(B)}_{j,f_j}(i)=r^{S(B)}_{j,f_j}=r_{j,f_j} \tag{27}$$

$$j=1...i-1;f_j=1...F_j+1$$

In addition, since we know the probability of being in the competition failure state (that models the situation in which the pseudo-machine does not produce because the other pseudo-machine is producing) we can use it to evaluate the parameters of the competition failure. Indeed, using a node equation balancing the probability of entering the competition failure state with the probability of leaving the same state we have, for part type A:

$$\pi(W^{U(A)})p^{U(A)}_{F_i+1}(i)=\pi(W^B)r^{U(A)}_{F_i+1}(i) \tag{28}$$

In this equation there are two unknowns that are the probabilities of failure and repair of the competition failure. Making considerations on the behavior of the pseudo-machine model it is possible to estimate the failure probability. The probability $W^{U(A)}$ of the pseudo-machine $M^{U(A)}(i)$ being operational, has been obtained by adding up the probabilities of being in three different states,

$\pi(W^A) = \pi(W^A W^B)\alpha_i^A$, $\pi(W^A S^B)$ and $\pi(W^A B^B)$. Therefore, it is possible to evaluate all the transition probabilities between these starting states and the competition failure state separately.

$$p_{F_i+1}^{U(A)}(i) = (1 - p^{U(A)}(i))\alpha_i^B \times \left(\frac{\pi(W^A)}{\pi(W^{U(A)})} + \sum_{j=1}^{i-1} \sum_{f_j=1}^{F_j} \frac{\pi(W^A S_{j,f_j}^B)}{\pi(W^{U(A)})} r_{j,f_j}^{S(B)} + \sum_{j=i+1}^{K} \sum_{f_j=1}^{F_j} \frac{\pi(W^A B_{j,f_j}^B)}{\pi(W^{U(A)})} r_{j,f_j}^{B(B)} \right) \quad (29)$$

Where $p^{U(A)}(i) = \sum_{j=1}^{i-1} \sum_{f_j=1}^{F_j} p_{j,f_j}^{U(A)}(i) + \sum_{j=i+1}^{K} \sum_{f_j=1}^{F_j} p_{j,f_j}^{B(A)}(i) + \sum_{i=1}^{f_i} p_{i,f_i}^{U(A)}$ is the sum of all the other failure probabilities of the pseudo-machine $M^{U(A)}(i)$. Now, using the equation (28), the probability of repair for the competition failure mode can be evaluated as follows:

$$r_{F_i+1}^{U(A)}(i) = \frac{\pi(W^{U(A)})}{\pi(W^B)} p_{F_i+1}^{U(A)}(i) \quad (30)$$

In a similar way we can find for machine $M^{U(B)}(i)$ the values of $p_{i,F_i+1}^{U(B)}$ and $r_{i,F_i+1}^{U(B)}$:

$$p_{F_i+1}^{U(B)}(i) = (1 - p^{U(B)}(i))\alpha_i^A \times \left(\frac{\pi(W^B)}{\pi(W^{U(B)})} + \sum_{j=1}^{i-1} \sum_{f_j=1}^{F_j} \frac{\pi(W^B S_{j,f_j}^A)}{\pi(W^{U(B)})} r_{j,f_j}^{S(A)} + \sum_{j=i+1}^{K} \sum_{f_j=1}^{F_j} \frac{W^B B_{j,f_j}^A}{\pi(W^{U(A)})} r_{j,f_j}^{B(A)} \right)$$

and

$$r_{F_i+1}^{U(B)}(i) = \frac{\pi(W^{U(B)})}{\pi(W^A)} p_{F_i+1}^{U(B)}(i) \quad (31)$$

Once local, remote and competition failure probabilities are evaluated they can be used within the building blocks $a(i)$ and $b(i)$.

5 Algorithm

Unknown failure parameters of all the pseudo-machines of the decomposed lines are determined by following an iterative algorithm inspired by the DDX algorithm. In particular it consists of the following steps:

1. Initialization: for each pseudo-machine of each building block, local failure parameters are initialized to the corresponding values of the machines of the original line, while remote failures and competition failures are initialized to a small value (for instance we used $\lambda = 0.05$).
   $M^{U(A/B)}(i)$:

   $$p_{i,f_i}^{U(A/B)}(i) = p_{i,f_i} \quad r_{i,f_i}^{U(A/B)}(i) = r_{i,f_i} \quad i = 1, ..., k-1 \; f_i = 1, .., F_i \quad (32)$$

   $$p_{i,F_i+1}^{U(A)}(i) = \lambda \quad r_{i,F_i+1}^{U(A)}(i) = \alpha_i^A \quad p_{i,F_i+1}^{U(B)}(i) = \lambda \quad r_{i,F_i+1}^{U(B)}(i) = \alpha_i^B \quad (33)$$

$$p_{j,f_j}^{U(A/B)}(i) = \lambda \quad r_{j,f_j}^{U(A/B)}(i) = r_{j,f_j} \quad j = 1, ..., i-1;\ f_j = 1, ..., F_j \tag{34}$$

$M^{D(A/B)}(i-1)$:

$$p_{i,f_i}^{D(A/B)}(i-1) = p_{i,f_i} \quad r_{i,f_i}^{D(A/B)}(i-1) = r_{i,f_i} \tag{35}$$
$$i = 2, ..., k\ f_i = 1, .., F_i$$

$$p_{i,F_i+1}^{D(A)}(i-1) = \lambda \quad r_{i,F_i+1}^{D(A)}(i-1) = \alpha_i^A$$
$$p_{i,F_i+1}^{D(B)}(i-1) = \lambda \quad r_{i,F_i+1}^{D(B)}(i-1) = \alpha_i^B \tag{36}$$

$$p_{j,f_j}^{B(A/B)}(i-1) = \lambda \quad r_{j,f_j}^{B(A/B)}(i-1) = r_{j,f_j} \tag{37}$$
$$j = i+1, ..., K;\ f_j = 1, ..., F_j$$

2. Step 1. For $i = 1, ..., K-1$: failure parameters of machines $M^{U(A)}(i)$ and $M^{U(B)}(i)$ are evaluated:
   - Unknown transition probabilities are calculated using equations (15).
   - Evaluation of all the state probabilities of the flexible machine M_i by using the linear system formed by equations (1) to (14). Blocking probabilities and transitions to blocking states are derived from previous iterations of the algorithm and they are equal to remote failures of downstream pseudo-machines $M^{D(A)}(i-1)$ and $M^{D(B)}(i-1)$, in case of $i > 1$, while for $i = 1$ they can be evaluated using equations (16).
   - Distribution of the calculated probabilities into the two pseudo-machine models of Figure 4, using equations (19) to (23), for both part types.
   - Evaluation of new local failures using equations (24), (25).
   - Calculation of remote failures using equations (26), (27).
   - Evaluation of competition failures using equations (29), (30), (31) and (32).
   - Insertion of calculated failure parameters into upstream pseudo-machines of building blocks $a(i)$ and $b(i)$.
   - Evaluation of average throughput, probabilities of blocking and probabilities of starvation of blocks a(i) and b(i) using the building block solution proposed in [5].
3. Step 2. For $i = K, ..., 2$: failure parameters of machines $M^{D(A)}(i-1)$ and $M^{D(B)}(i-1)$ are evaluated:
   - Unknown transition probabilities are calculated using equations (16).
   - Evaluation of all the state probabilities of the flexible machine M_i by using the linear system formed by equations (1) to (14). Starvation probabilities and transitions to starvation states are derived from previous iterations of the algorithm and they are equal to remote failures of upstream pseudo-machines $M^{U(A)}(i)$ and $M^{U(B)}(i)$, in case of $i < K$, while for $i = K$ they can be evaluated using equations (15).
   - Distribution of the calculated probabilities into the two pseudo-machine models similar to those of Figure 5, using equations similar to (19) to (23).
   - Evaluation of new local failures using equations similar to (24), (25).
   - Calculation of remote failures using equations similar to (26), (27).

- Evaluation of competition failures using equations similar to (29), (30), (31) and (32).
- Insertion of calculated failure parameters into downstream pseudo-machines of building blocks $a(i-1)$ and $b(i-1)$.
- Evaluation of average throughput, probabilities of blocking and probabilities of starvation of blocks $a(i-1)$ and $b(i-1)$ using the building block solution proposed in [5].

The algorithm stops when the following condition becomes true:

$$| E^A(i)-E^A(i-1) | \leq \varepsilon \ \ and \ \ | E^B(i)-E^B(i-1) | \leq \varepsilon \quad i = 1, ..., K-1 \quad (38)$$

Performance measures can be evaluated as follows:

Average throughput of the line

$$E^A = E^A(1) = E^A(2) = ... = E^A(K-1)$$
$$E^B = E^B(1) = E^B(2) = ... = E^B(K-1)$$

Average buffer level in the line

$$\overline{n}_i^A = \overline{n}^A(i) \qquad \overline{n}_i^B = \overline{n}^B(i) \qquad i = 1, ..., K-1$$

6 Numerical results

In order to show the accuracy of the new analytical method developed (method CMT) a set of numerical tests has been carried out comparing the analytical results with those obtained running simulation experiments. More than 150 lines producing two products have been analyzed using the proposed method with probability of failure and repair varying in the following ranges: $0 < p_{i,f_i} < 0.25$ and $0 < r_{i,f_i} < 0.8$. In particular, systems with three-machines/four-buffers (Table 1), four machines/six buffers (Table 2), five machines/eight buffers (Table 3) and six

Table 1. Three machines cases

CASE 1							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,034	0,66	16	16	SIM.	0,2278	0,52
2	0,03	0,088	16	16	CMT	0,2242	0,5215
3	0,06	0,43	/	/	Δ %	1,58	0,288
α=0,3 α1=0,3046							

CASE 2							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,029	0,44	25	25	SIM.	0,4891	0,3271
2	0,02	0,09	25	25	CMT	0,491	0,3272
3	0,034	0,66	/	/	Δ %	0,388	0,03
α=0,6 α1=0,5992							

CASE 3							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,06	0,44	22	12	SIM.	0,5585	0,3031
2	0,032	0,43	22	12	CMT	0,5605	0,3022
3	0,07	0,44	/	/	Δ %	0,358	0,297
α=0,65 α1=0,6482							

CASE 4							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,023	0,5	13	13	SIM.	0,5801	0,2474
2	0,034	0,54	13	13	CMT	0,5788	0,248
3	0,09	0,43	/	/	Δ %	0,224	0,2425
α=0,7 α1=0,7010							

CASE 5							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,023	0,55	12	12	SIM.	0,3107	0,5751
2	0,043	0,5	12	12	CMT	0,3111	0,5759
3	0,07	0,55	/	/	Δ %	0,128	0,139
α=0,35 α1=0,3507							

CASE 6							
M	p	r	N^A	N^B	/	E^A	E^B
1	0,033	0,7	20	20	SIM.	0,571	0,3815
2	0,012	0,44	20	20	CMT	0,5725	0,3817
3	0,033	0,7	/	/	Δ %	0,262	0,052
α=0,6 α1=0,5994							

Table 2. Four machines cases

CASE 1								CASE 2							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,034	0,66	9	9	SIM.	0,4166	0,2806	1	0,09	0,33	14	14	SIM.	0,3153	0,4678
2	0,032	0,43	9	9				2	0,043	0,65	14	14			
3	0,087	0,2	9	9	CMT	0,4167	0,2798	3	0,076	0,36	14	14	CMT	0,315	0,4714
4	0,0423	0,304	/	/				4	0,039	0,22	/	/			
α=0,6		α1=0,5975			Δ %	0,024	0,285	α=0,4		α1=0,4026			Δ %	0,095	0,769
CASE 3								CASE 4							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,07	0,44	9	9	SIM.	0,5276	0,2373	1	0,033	0,6	14	14	SIM.	0,4709	0,316
2	0,098	0,33	9	9				2	0,076	0,33	14	14			
3	0,057	0,43	9	9	CMT	0,5321	0,2387	3	0,04	0,66	14	14	CMT	0,4743	0,3162
4	0,0023	0,06	/	/				4	0,09	0,34	/	/			
α=0,7		α1=0,6897			Δ %	0,853	0,59	α=0,6		α1=0,5984			Δ %	0,722	0,063
CASE 5								CASE 6							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,03	0,66	12	12	SIM.	0,4665	0,3125	1	0,33	0,8	20	20	SIM.	0,4156	0,2785
2	0,08	0,33	12	12				2	0,28	0,64	20	20			
3	0,043	0,77	12	12	CMT	0,4713	0,3142	3	0,167	0,44	20	20	CMT	0,4198	0,2759
4	0,09	0,33	/	/				4	0,203	0,587	/	/			
α=0,6		α1=0,5988			Δ %	1,028	0,544	α=0,6		α1=0,5987			Δ %	1,01	0,933

Table 3. Five machines cases

CASE 1								CASE 2							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,032	0,5	12	12	SIM.	0,592	0,2588	1	0,043	0,66	14	14	SIM.	0,4751	0,3165
2	0,023	0,44	12	12				2	0,034	0,77	14	14			
3	0,09	0,54	12	12	CMT	0,5968	0,2603	3	0,0345	0,546	14	14	CMT	0,4742	0,3161
4	0,023	0,32	12	12				4	0,09	0,34	14	14			
5	0,0763	0,485	/	/	Δ %	0,803	0,587	5	0,0345	0,77	/	/	Δ %	0,189	0,126
α=0,7		α1=0,6958						α=0,6		α1=0,6002					
CASE 3								CASE 4							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,067	0,44	11	11	SIM.	0,596	0,2637	1	0,078	0,44	16	16	SIM.	0,4478	0,3538
2	0,034	0,55	11	11				2	0,039	0,88	16	16			
3	0,023	0,2	11	11	CMT	0,6071	0,2638	3	0,04	0,33	16	16	CMT	0,4505	0,3546
4	0,034	0,55	11	11				4	0,08	0,33	16	16			
5	0,048	0,34	/	/	Δ %	1,862	0,037	5	0,0546	0,32	/	/	Δ %	0,603	0,226
α=0,7		α1=0,6932						α=0,56		α1=0,5586					
CASE 5								CASE 6							
M	p	r	N^A	N^B	/	E^A	E^B	M	p	r	N^A	N^B	/	E^A	E^B
1	0,034	0,55	9	9	SIM.	0,1566	0,2349	1	0,078	0,55	14	14	SIM.	0,4681	0,3527
2	0,076	0,234	9	9				2	0,043	0,88	14	14			
3	0,0908	0,554	9	9	CMT	0,1549	0,2332	3	0,12	0,55	14	14	CMT	0,4678	0,3528
4	0,342	0,22	9	9				4	0,032	0,66	14	14			
5	0,065	0,55	/	/	Δ %	1,085	0,723	5	0,043	0,66	/	/	Δ %	0,064	0,028
α=0,4		α1=0,399						α=0,57		α1=0,5703					

machines/ten buffers (Table 4) with one failure parameter for each machine have been studied and a sampling of results are reported in the following tables. Also, systems with machines characterized by multiple failure modes are studied and the results are reported in Table 5. For each simulation experiment 10 replications have been performed, with a warm-up period of 10^5 time units followed by simulation period of 10^6 time units. Average throughput has been evaluated with a 95% half confidence interval of $0.000\overline{9}$ as maximum value. Average buffer level has been evaluated with a 95% half confidence interval of 0.08 as maximum value.

In all the tables, for each analyzed case, failure and repair probabilities of the machines are reported on the left, together with buffer capacities and the α_i^A parameters, that are equal for all the machines in the line. The average production rates of the line, calculated with the proposed method and with simulation, are

Table 4. Six machines cases

CASE 1

M	p	r	N^A	N^B	/	E^A	E^B
1	0,06	0,29	12	12	SIM.	0,496	0,332
2	0,012	0,33	12	12			
3	0,08	,55	12	12	CMT	0,498	0,33
4	0,0167	0,55	12	12			
5	0,09	0,65	12	12	Δ %	0,463	0,662
6	0,01	0,33	/	/			
α=0,6					α1=0,5987		

CASE 2

M	p	r	N^A	N^B	/	E^A	E^B
1	0,033	0,6	14	14	SIM.	0,507	0,339
2	0,08	0,44	14	14			
3	0,034	0,77	14	14	CMT	0,508	0,338
4	0,023	0,55	14	14			
5	0,054	0,7	14	14	Δ %	0,078	0,029
6	0,033	0,55	/	/			
α=0,6					α1=0,5997		

CASE 3

M	p	r	N^A	N^B	/	E^A	E^B
1	0,0045	0,33	12	12	SIM.	0,48	0,323
2	0,078	0,343	12	12			
3	0,012	0,32	12	12	CMT	0,486	0,324
4	0,043	0,66	12	12			
5	0,08	0,34	12	12	Δ %	1,166	0,154
6	0,0435	0,65	/	/			
α=0,6					α1=0,5976		

CASE 4

M	p	r	N^A	N^B	/	E^A	E^B
1	0,075	0,33	14	14	SIM.	0,487	0,326
2	0,0364	0,55	14	14			
3	0,023	0,32	14	14	CMT	0,489	0,326
4	0,09	0,44	14	14			
5	0,045	0,66	14	14	Δ %	0,307	0,091
6	0,0482	0,547	/	/			
α=0,6					α1=0,599		

CASE 5

M	p	r	N^A	N^B	/	E^A	E^B
1	0,0475	0,332	13	13	SIM.	0,594	0,261
2	0,0734	0,54	13	13			
3	0,0234	0,44	13	13	CMT	0,602	0,26
4	0,0703	0,44	13	13			
5	0,0372	0,45	13	13	Δ %	1,398	0,413
6	0,1	0,66	/	/			
α=0,7					α1=0,6943		

CASE 6

M	p	r	N^A	N^B	/	E^A	E^B
1	0,014	0,22	15	15	SIM.	0,291	0,535
2	0,089	0,43	15	15			
3	0,046	0,33	15	15	CMT	0,291	0,538
4	0,073	0,55	15	15			
5	0,0395	0,44	15	15	Δ %	0,137	0,523
6	0,082	0,63	/	/			
α=0,35					α1=0,3527		

Table 5. Multiple failure machines cases

CASE 1

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,012	0,4	0,06	0,55	14	14
2	0,09	0,43	0,03	0,6	14	14
3	0,014	0,33	0,05	0,77	14	14
4	0,09	0,37	/	/	/	/
α=0,6			α1=0,5997			

/	E^A	E^B
SIM.	0,472	0,3173
CMT	0,4764	0,3176
Δ %	0,932	0,094

CASE 2

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,0578	0,332	0,043	0,66	12	12
2	0,0786	0,44	0,033	0,25	12	12
3	0,056	0,77	0,0304	0,55	12	12
4	0,023	0,55	/	/	/	/
α=0,35			α1=0,3552			

/	E^A	E^B
SIM.	0,269	0,4882
CMT	0,27	0,4927
Δ %	0,371	0,921

CASE 3

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,044	0,5	0,03	0,42	14	14
2	0,019	0,33	0,04	0,5	14	14
3	0,078	0,34	0,05	0,6	14	14
4	0,09	0,44	/	/	/	/
α=0,65			α1=0,6475			

/	E^A	E^B
SIM.	0,4933	0,2685
CMT	0,4947	0,2669
Δ %	0,283	0,595

CASE 4

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,0034	0,66	0,054	0,5	14	14
2	0,0234	0,43	0,07	0,44	14	14
3	0,0519	0,43	/	/	14	14
4	0,0746	0,554	/	/	/	/
α=0,6			α1=0,5971			

/	E^A	E^B
SIM.	0,4911	0,3313
CMT	0,494	0,33
Δ %	0,59	0,392

CASE 5

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,05	0,6	0,034	0,5	9	9
2	0,109	0,33	/	/	9	9
3	0,08	0,44	/	/	/	/
α=0,6			α1=0,5965			

/	E^A	E^B
SIM.	0,4466	0,302
CMT	0,4495	0,3021
Δ %	0,649	0,033

CASE 6

M	p_1	r_1	p_2	r_2	N^A	N^B
1	0,055	0,3	0,023	0,44	14	14
2	0,1	0,66	/	/	14	14
3	0,084	0,56	/	/	/	/
α=0,4			α1=0,4004			

/	E^A	E^B
SIM.	0,3246	0,4861
CMT	0,3238	0,4854
Δ %	0,246	0,144

reported on the right and the error between the evaluations is estimated using the following equations:

$$\Delta\%E^A = \left| \frac{E^A_{SIM} - E^A_{CMT}}{E^A_{SIM}} \right| \cdot 100$$

$$\Delta\%E^B = \left| \frac{E^B_{SIM} - E^B_{CMT}}{E^B_{SIM}} \right| \cdot 100$$

As it can be seen by the results provided in this section and by the summary of results in Table 6, the algorithm has proven to be reliable and accurate in all the tested cases; indeed the maximum error in throughput evaluation is around 3% and a high percentage of cases have an error in throughput evaluation lower than 1%.

In Table 7 the error in the evaluation of the average buffer level is reported for the six machine cases of Table 4. The error between evaluations has been

Table 6. Summary of results in 150 test cases

$N.\ MACHINES$	3	4	5	6
$ERROR > 2\%$	6,8%	4,7%	2,9%	10%
$ERROR < 1\%$	81,8%	66,6%	70,6%	72,2%
$MAX\ ERROR$	2,5%	2,38%	2,77%	3,13%

Table 7. Error in average buffer level evaluation for cases of Table 4

		$\bar{n}_1^A$	$\bar{n}_1^B$	$\bar{n}_2^A$	$\bar{n}_2^B$	$\bar{n}_3^A$	$\bar{n}_3^B$	$\bar{n}_4^A$	$\bar{n}_4^B$	$\bar{n}_5^A$	$\bar{n}_5^B$
CASE 1	SIM	1,748	1,4686	3,6232	3,1863	1,11	0,85574	2,726	2,4771	0,83305	0,61647
	CMT	1,10785	1,13295	2,23295	1,71595	0,908375	0,748192	1,7447	1,40621	0,887981	0,70784
	Δ%	**-5,33458**	**-2,29708**	**-11,58542**	**-12,25292**	**-1,68021**	**-1,13790**	**-8,17750**	**-8,92408**	**0,45776**	**0,76142**
CASE 2	SIM	12,95600	13,20600	0,86650	0,65701	0,94171	0,70275	1,32440	1,06550	1,12500	0,88386
	CMT	12,79880	12,97950	1,09995	0,94743	1,10040	0,99403	1,29945	1,15973	1,11326	0,96026
	Δ%	**-1,12286**	**-1,61786**	**1,66750**	**2,07443**	**1,13350**	**2,08059**	**-0,17821**	**0,67307**	**-0,08386**	**0,50998**
CASE 3	SIM	11,39800	11,56600	3,84710	4,00540	6,49220	6,56476	8,77570	8,75540	1,01810	0,80531
	CMT	11,17650	11,31480	4,65107	5,46609	7,84880	8,10610	10,92440	11,11560	1,06864	0,87437
	Δ%	**-1,84583**	**-2,09333**	**6,69975**	**12,17242**	**11,30500**	**12,84500**	**17,90563**	**19,66833**	**0,42117**	**0,57551**
CASE 4	SIM	2,95910	2,73910	5,65910	5,11190	8,39680	8,02270	1,01160	0,79142	1,38660	1,14050
	CMT	1,14660	0,93787	3,42401	3,15131	6,24842	6,39853	1,09866	0,91944	1,24886	1,08014
	Δ%	**-12,94643**	**-12,86592**	**-15,96493**	**-14,00421**	**-15,34557**	**-11,60121**	**0,62186**	**0,91445**	**-0,98386**	**-0,43114**
CASE 5	SIM	7,82210	8,10910	6,34340	6,85420	9,66030	10,27300	3,75570	3,32880	6,28700	6,07820
	CMT	8,73431	9,88830	7,81789	8,73258	10,40810	10,87420	2,93400	2,11878	7,97050	7,93243
	Δ%	**7,01700**	**13,66615**	**11,34223**	**14,44908**	**5,75231**	**4,62462**	**-6,32077**	**-9,30765**	**12,95000**	**14,26331**
CASE 6	SIM	13,78800	13,37700	3,50440	4,01070	2,52080	2,98330	1,34420	1,70680	2,16040	2,39340
	CMT	14,01800	13,57760	1,64888	2,48656	1,75079	2,24086	1,17280	1,51024	1,83252	2,06845
	Δ%	**1,53333**	**1,33733**	**-12,37013**	**-10,16093**	**-5,13340**	**-4,94960**	**-1,14267**	**-1,31040**	**-3,51920**	**-2,16633**

THROUGHPUT

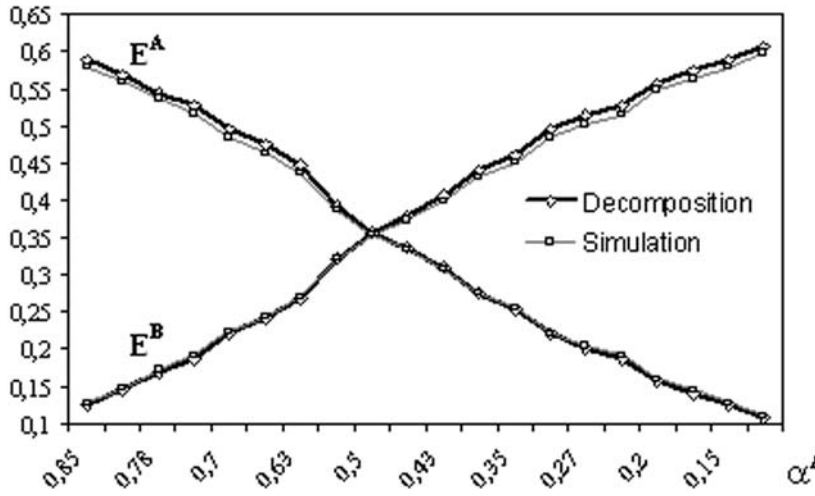


Fig. 6. Throughput evaluation with α_i^A equal for all the machines of the line and variable

calculated by using the following equations:

$$\Delta\%(\overline{n}_i^A) = \left| \frac{(\overline{n}_i^A)_{SIM} - (\overline{n}_i^A)_{CMT}}{N_i^A} \right| \cdot 100$$

$$\Delta\%(\overline{n}_i^B) = \left| \frac{(\overline{n}_i^B)_{SIM} - (\overline{n}_i^B)_{CMT}}{N_i^B} \right| \cdot 100$$

Table 8. Error in throughput evaluation

α^A	CMT MODEL		SIMULATOR		ERROR %	
	E^A	E^B	E^A	E^B	E^A	E^B
0,85	0,588	0,125	0,5778	0,1272	**1,765**	**1,729**
0,82	0,5695	0,1432	0,5594	0,146	**1,805**	**1,917**
0,78	0,5448	0,168	0,5361	0,1723	**1,622**	**2,495**
0,75	0,526	0,1872	0,5163	0,1909	**1,878**	**1,938**
0,7	0,494	0,2201	0,4848	0,2238	**1,897**	**1,653**
0,67	0,474	0,24	0,4636	0,2423	**2,243**	**0,949**
0,63	0,447	0,2671	0,4366	0,2696	**2,382**	**0,927**
0,55	0,3921	0,3225	0,3858	0,3216	**1,632**	**0,279**
0,5	0,357	0,357	0,3534	0,3545	**1,018**	**0,705**
0,47	0,3365	0,378	0,334	0,373	**0,748**	**1,34**
0,43	0,3086	0,4058	0,3082	0,3992	**0,129**	**1,653**
0,38	0,274	0,4402	0,2755	0,4315	**0,544**	**2,016**
0,35	0,2536	0,4606	0,2552	0,451	**0,626**	**2,128**
0,3	0,2202	0,4941	0,2238	0,484	**1,608**	**2,086**
0,27	0,2002	0,514	0,2045	0,5025	**2,102**	**2,288**
0,25	0,1873	0,526	0,1908	0,515	**1,834**	**2,135**
0,2	0,1557	0,5563	0,1594	0,5499	**2,321**	**1,163**
0,175	0,14	0,5727	0,1433	0,5647	**2,302**	**1,416**
0,15	0,1254	0,5886	0,1277	0,5788	**1,801**	**1,693**
0,12	0,1076	0,6064	0,1093	0,5974	**1,555**	**1,506**

As it normally happens in decomposition methods, errors in average buffers level evaluation are much higher than those regarding throughput.
It is worth noting that the total throughput of the line (the sum of throughputs of part types A and B) is divided between part types A and B differently from the values of α_i^A and α_i^B introduced for the machines of the line. This is due to the fact that the occurrence of blocking and starvation is different for part type A or B depending on their relative buffer capacities. In the following tables α indicates the value of α_i^A and $\alpha 1$ indicates the ratio between throughput of part type A and the total throughput, resulting from the simulation. It would be important, as a future development of the research, to develop a method able to assess the values of α_i^A parameters for each machine of the line, starting with the $\alpha 1$ value that we want to effectively obtain from the line.

In order to study the accuracy of the method for different values of α_i^A and α_i^B in the line and for each single machine, some focused tests have been realized. In particular we studied a six machine line with α_i^A variable for the bottleneck machine and equal to 0,6 for all the others machines of the line. The behavior of the system is well approximated by the method for values of α_5^A similar to those of other machines but, in other cases, the method doesn't evaluate performance measures of that line correctly. This limitation of the application field of the method is not very relevant, because in real automated multiproduct flow lines the α_i^A parameter is normally constant throughout the line. In this case the proposed method correctly

estimates average throughput of the test line as it is shown in (Fig. 6) and (Table 8) for a wide range of variability of parameter α_i^A.

As it can be seen by the results provided in this section, the algorithm has proven to be reliable and accurate in all the tested cases with α_i^A and α_i^B parameters equal for all the machines of the line.

7 Conclusions

A new approximate analytical method for the performance evaluation of multiproduct automated flow lines with multiple failure modes and finite buffer capacity has been proposed. The method has been applied to the case of lines producing two different part types, but is amenable of extension to the case of *n* part types. An algorithm inspired by the DDX algorithm has been developed to evaluate failure probabilities for all pseudo-machines of the decomposed lines. Extensive testing has proven the accuracy of the method. As a future development, the method could be extended to the case of continuous lines with multiple part types. In addition, the method in principle can be extended to study assembly/disassembly networks [1,2] and fork and join systems [3,4].

References

1. Tolio T, Matta A, Levantesi R (2000) Performance evaluation of assembly/disassembly systems with deterministic processing times and multiple failure modes. In: ICPR2000 International Conference on Production Research, Bangkok, Thailand
2. Gershwin SB (1991) Assembly/disassembly systems: an efficient decomposition algorithm for tree structured networks. IIE Transactions 23(4): 302–314
3. Helber S (1999) Performance analysis of flow lines with nonlinear flow of material, vol 243. Lecture notes in economics and mathematical systems. Springer, Berlin Heidelberg New York
4. Helber S (2000) Approximate analysis of unreliable transfer lines with splits in the flow of materials. Annals of Operations Research (93): 217–243
5. Tolio T, Gershwin SB, Matta A (2002) Analysis of two-machine lines with multiple failure modes. IIE Transactions 2002 34(1): 51–62
6. Nemec JE (1999) Diffusion and decomposition approximations of stochastic models of multiclass processing networks. PhD thesis, Massachusetts Institute of Technology, February
7. Tolio T, Matta A (1998) A method for performance evaluation of automated flow lines. Annals of CIRP 47(1): 373–376
8. Le Bihan H, Dallery Y (1999) An improved decomposition method for the analysis of production lines with unreliable machines and finite buffers. International Journal of Production Research 37(5): 1093–1117

Automated flow lines with shared buffer

A. Matta, M. Runchina, and T. Tolio

Politecnico di Milano, Dipartimento di Meccanica, via Bonardi 9, 20133 Milano, Italy
(e-mail: {andrea.matta,tullio.tolio}@polimi.it)

Abstract. The paper addresses the problem of fully using buffer spaces in manufacturing flow lines. The idea is to exploit recent technological devices to move in reasonable times pieces from a machine to a common buffer area of the system and vice versa. In such a way machines can avoid their blocking since they can send pieces to the shared buffer area. The introduction of the buffer area shared by all machines of the system leads to an increase of production rate as demonstrated by simulation experiments. Also, a preliminary economic evaluation on a real case has been carried out to estimate the profitability of the system comparing the increase of production rate, obtained with the new system architecture, with the related additional cost.

Keywords: Flow lines – Buffer allocation – System design – Performance evaluation

1 Introduction

A manufacturing flow line is defined in literature as a serial production system in which parts are worked sequentially by machines: pieces flow from the first machine, in which they are still raw parts, to the last machine where the process cycle is completed and the finished parts leave the system. When a machine is not available, parts wait in the buffer immediately upstream the machine. If the number of parts flowing in the system is constant during the production, these systems are also called closed flow lines (see Fig. 1 where rectangles and circles represent machines and buffers of the system respectively) to distinguish them from open flow lines where the number of parts is not maintained constant. Gershwin gives in [4] a general description of flow lines in manufacturing. The production rate of flow lines is clearly a function of speed and reliability of machines: faster and more reliable machines are and higher the production rate is. However, since machines

Correspondence to: A. Matta

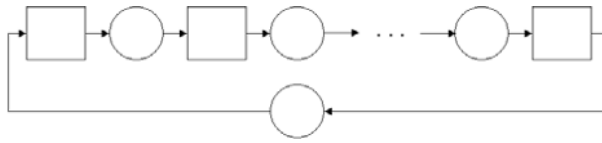


Fig. 1. Scheme of closed flow lines

can have different speeds and may be affected by random failures, the part flow can be interrupted at a certain point of the system causing blocking and starvation of machines. In particular, blocking in the line occurs when at least one machine cannot move the parts just worked (BAS, Blocking After Service) or still to work (BBS, Blocking Before Service) to the next station. In flow lines the blocking of a machine can be caused only by a long processing time or a failure of a downstream machine. Analogously, starvation occurs when one or more machines cannot be operational because they have no input part to work; in this case the machine cannot work and it is said to be starved. In flow lines the starvation of a machine can be caused only by a long processing time or a failure of an upstream machine. Therefore, in flow lines the state of a machine affects the rest of the system because of blocking and starvation phenomena that propagate upstream and downstream respectively the source of flow interruption in the line. If there is no area where to store pieces between two adjacent machines, the behavior of machines is strongly correlated.

In order to decrease blocking and starvation phenomena in flow lines, buffers between two adjacent machines are normally included to decouple the machines behavior. Indeed, buffers allow to adsorb the impact of a failure or a long processing time because (a) the presence of parts in buffers decreases the starvation of machines and (b) the possibility of storing parts in buffers decreases the blocking of machines. Therefore, production rate of flow lines is also a function of buffer capacities; more precisely, production rate is a monotone positive function of the total buffer capacity of the system. Refer to [5, 7] for a list of works focused on the properties of production rate in flow lines as a function of the buffer size.

Traditionally, flow lines have been deeply investigated in literature. Researchers' efforts have been devoted to develop new models for evaluating the performance of flow lines and for optimizing their design and management in shop floors. Operations research techniques like simulation and analytical methods have been widely used to estimate system performance parameters such as throughput and work in process level. Performance evaluation models are currently used in configuration algorithms for finding the optimal design of flow lines taking into account the total investment cost, operative cost and production rate of the system. In synthesis, academic innovation has been mainly focused on the development of performance evaluation and optimization methods of flow lines without entering into several mechanical details. See also the review [1] of Dallery and Gershwin on a detailed view of performance evaluation models for flow lines and an updated recent state of the art on optimization techniques applied in practice [8]. Indeed, most of works is at system level as they deal with optimization of macro variables such as number of machines in the line, buffer capacities and machines' speed and

efficiency. On the other hand, engineers of firms have had to face the complexity due to the fact that flow lines are designed in practice with all their mechanical components. Innovation from builders of manufacturing flow lines has been mainly dedicated to increase machines reliability and to reduce system costs by improving the design of specific mechanical components such as feed drives, spindles, transporters, etc.

Therefore, advancements in flow line evolution do not regard the main philosophy of the system. Parts are loaded into the system at the first machine and, after having been processed, they are moved into the first buffer waiting for the availability of the second machine. Blocking phenomena is limited by buffers, larger is their capacity and higher the throughput of the line is. However, buffers in flow lines are dedicated to machines; this characteristic implies that a buffer can contain only pieces worked by the immediately upstream machine. Therefore, when a long failure occurs at a machine of the line, the portion of the system upstream the failed machine is blocked but upstream machines continue to work until their corresponding buffers are full. On the other hand, the portion of the system downstream the failed machine is starved because downstream machines cannot work since they do not have any piece to work. In that case the buffer area downstream the failed machine cannot be used to store parts worked by machines that are upstream the failed machine since empty buffers are dedicated and cannot be used for pieces coming out from other machines. It appears that buffer spaces are not fully exploited when needed. The problem of properly using all the available space in flow lines represents the argument of this paper.

2 Flow lines with shared buffer

2.1 Motivation

The paper presents a new concept of manufacturing flow line characterized by two different types of buffers: traditional dedicated buffers and a common buffer shared by all the machines of the system. The common buffer allows to store pieces at any point of the system thus increasing the buffer capacity of each machine (see Fig. 2). The main advantage is related to the fact that wherever an interruption of flow is in the system, the common shared buffer can be used by all machines. As a consequence, blocking of machines should be lower than that of classical flow lines thus allowing an increase of production rate at constant total buffer capacity.

However, profitability of the new system architecture depends on costs incurred for the additional shared buffer. Traditionally the main goal in the design phase of flow lines is to find the system configuration at minimum costs constrained

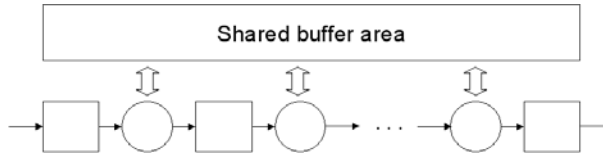


Fig. 2. Scheme of the proposed system architecture

to a minimum value of production rate. In this context, the introduction of the shared buffer in flow lines is possible only if the time necessary for moving parts from shared buffer to machines is small and the relative investment for additional mechanical components is reasonable.

Indeed, in our opinion costs are the main reason for which shared buffers have not still be adopted in manufacturing flow lines. Designing shared buffer in flow lines implies to have additional components, and thus larger costs, for moving pieces from machines to the central buffer and vice versa. However, technology is now mature to be used for this scope at affordable costs. Several manufacturers can provide at low costs a wide set of transport modules for part movements. These modules can be assembled in a flexible way to move parts through the system; actually the speed of conveyors is around 20 m/min on average depending on the weight of parts. Parts can follow linear paths, as usual in flow lines, and circular paths with small rounds. Furthermore in order to save floor space, parts can be moved up or down for reaching different heights. The cost of transporter modules is now affordable allowing their intensive usage in practice at the same productivity level, defined in the paper as the amount of output obtained for one unit of input. We consider the production rate of the system as the output and the total cost of the system as the input. It is rather difficult to increase productivity of manufacturing systems since a specific action that can increase the production rate of a system is normally balanced by the effort required. Actions that can improve system productivity should reduce the total costs (reduction of machines and fixtures cost, reduction of adaptation cost, etc.) without reducing the production rate or should increase the production rate (shorter system set-up times, reduction of unproductive times, improvement of system availability, etc.) without increasing costs. The proposed system can be considered interesting for practical exploitation if its productivity remains constant or increases in comparison with traditional systems.

2.2 System description

The proposed system architecture is a flow line composed of K machines separated by limited buffers. In case of open systems the number of buffers is equal to $K-1$ and we assume that the first machine is never starved and the last machine is never blocked; in case of closed systems the number of buffers is equal to the number of machines. We denote with M_i and B_i (with $i = 1, ..., K-1, K$) the i-th machine and the i-th dedicated buffer respectively.

Machines are normally unreliable and their efficiency depends on their failure and repair rates distributions. The $K-1$ buffers (or K in closed flow lines) are dedicated to their corresponding machines: buffer B_1 contains only pieces already worked by first machine M_1, buffer B_2 contains only pieces already worked by second machine M_2, and so on. If buffer B_i is full, i.e. the buffer level has reached the buffer capacity, machine M_i can send worked pieces to the shared buffer denoted with B_s that is located in a specific area of the system, shared by all the machines, where it is possible to put pieces independently by their process status. A generic machine M_i is blocked only if both dedicated and shared buffers, i.e. buffers B_i and

B_s, are full. The presence of shared buffer decreases blocking phenomena in the flow line: if the dedicated buffer is full, pieces worked by machine M_i can be moved to the shared buffer until the part flow resumes at machine M_{i+1} and the level of buffer B_i decreases. In more detail, a part which cannot be stored in a dedicated buffer stays in the shared buffer until a place in the dedicated buffer becomes available. The way in which parts in the shared buffer are positioned depends on the technology used and the management rules adopted. If the shared buffer consists of a simple conveyor on which parts flow until a new space is available at dedicated buffers, the ordering of parts depends on the their entering sequence in the conveyor. If the shared buffer consists of a series of racks, the ordering of parts depends on the particular management rule adopted; in this case it is necessary to have a resource like a robot or a carrier that takes parts from machines and put them on racks. Tempelmeier and Kuhn describes different mechanisms in the case of Flexible Manufacturing Systems (FMS) with central buffer [9].

A certain amount of time is necessary for physically moving parts from a dedicated buffer area to the shared buffer area and vice versa. Therefore, if blocking decreases with the introduction of the shared buffer the starvation increases [9]. In the remainder of the paper we call this time the *travel time* denoted with t_t. The profitability of the system depends on the value of travel time and its impact on system performance. If the travel time is reasonably small, then the penalty time incurred for using the shared buffer does not deeply decrease the system performance since, after the resumption of flow, the time spent by parts for going from the shared buffer area to the dedicated one is hidden, i.e. covered by the pieces already present in the dedicated area and processed in the meanwhile by machine M_{i+1}. If the travel time is large, then the penalty time incurred for using the shared buffer can strongly decrease the system performance since machines are frequently starved. The increase of starvation as a consequence of the transport time from/to the shared buffer has been previously described by Tempelmeier and Kuhn [9] in the analysis of a special FMS configured as a flexible flow line. The next section reports a numerical analysis for assessing the productivity of flow lines with shared buffer in different situations.

3 Numerical evaluation

The objective of the section is to evaluate the gain in terms of productivity due to the introduction of shared buffers in production lines. To do this, the experimentation has been carried out by simulating flow lines on simple test cases, created ad hoc to understand the system behavior in different situations (Sect. 3.1), and on a real flow line (Sect. 3.2).

3.1 Test cases

We consider a closed flow line composed of five machines, each one with a finite buffer capacity immediately downstream. Machines are unreliable and characterized by the same type of failure. Failures are time dependent and do not depend

on processing times of operations at machines. Failures have mean time to failure (MTTF) and mean time to repair (MTTR) exponentially distributed with means 1000 s and 100 s respectively. The blocking mechanism is the BAS (Blocking After Service) type. The cycle time of each machine of the system is the same and is denoted with t_c, i.e. the line is balanced because machines have also the same efficiency. The number of parts circulating in the system is maintained constant during production and equal to P. For simplicity, dedicated buffers have the same capacity N_i with $i = 1, ..., K$.

Table 1. Test case: factor levels of the 2^5 experiment

Factors	Low	High
Cycle time (t_c)	5 s	60 s
Total buffer capacity (N_{TOT})	100	125
Portion of dedicated buffer capacity (α)	0.5	1
Travel time (t_t)	0 s	30 s
Number of parts (P)	75	90

The goal of the experiment is to evaluate by means of steady state simulations the significance that potential factors may have on the main performance indicator as the production rate is. Factors taken into consideration in the experiment are: the machine cycle time t_c , the total buffer capacity N_{TOT}, the portion of dedicated buffer capacity α, the travel time t_t and the number of parts that circulate in the system P. The design of experiments is a 2^5 factorial plan; Table 1 reports the factors' levels chosen in the experiment. The parameter α can assume values between 0 and 1. Notice that systems with $\alpha = 1$ correspond to traditional flow lines in which the whole buffer capacity of the system is dedicated. In each treatment of the designed factorial plan 15 replications of simulation have been carried out and statistics have been collected after a warm-up period of 86400 simulated seconds and 25000 finished pieces. In each simulated scenario the capacity of dedicated buffers is calculated in the following way:

$$N_i = \frac{N_{TOT} \cdot \alpha}{K}, \qquad i = 1, ..., K \tag{1}$$

while the capacity of the shared buffer is equal to $N_{TOT} \cdot (1 - \alpha)$. The analysis of variance has been applied to test the significance of the analyzed factors on the system's efficiency, denoted with E and calculated as:

$$E = \frac{X}{X^*} \tag{2}$$

where X is the average production rate collected in a simulation run and X^* is the maximum production rate calculated without considering failures, blocking and starvation at machines. However, since normality assumptions on residuals is not satisfied, we have been forced to divide the analyzed response values into two distinct populations corresponding to low and high levels of the cycle time factor.

After that, all assumptions required by the analysis of variance have been satisfied and the main results are now presented. In particular the Anderson-Darling and Bartlett tests at 95 percent confidence level have been used to test normality and variance homogeneity of residuals respectively, the independence was assured by the randomized execution of experiments and different seeds used for generating pseudo-random numbers. Results from ANOVA are reported in Tables 2 and 3 where significant factors and interactions are recognizable: a source is significant on the system's efficiency if the p-value in the last column is lower than the Bonferroni's alpha family (chosen equal to 0.05) divided by the number of executed statistical tests (i.e. 15 in this experiment).

As far as the main effects are concerned, the number of pallets that circulate in the system, the portion of dedicated buffers and the total buffer capacity are significant for both the populations with different cycle times. The main effect of a factor is the average change in the response due to moving the factor from its low level to its high level [6]; this average is taken over all combinations of the factor levels in the design. A first conclusion is that in the analyzed system a travel time value equal to 0 or $30s$ is not relevant for the system's efficiency due to the values chosen for the levels. However increasing to a threshold value, greater than 30s, the travel time leads to bad performance; we will see at the end of the paragraph the threshold values after which the travel time becomes significant. The factor α is significant for both populations and, furthermore, it is possible to conclude, by comparing with the Tukey's method the two levels, that the analyzed closed flow line with shared buffer has statistically an efficiency superior than that of the analyzed traditional flow line. Notice that the difference of efficiency in the two levels is around 5% for $t_c = 5s$ and 1% for $t_c = 60s$. The significance of the number of pallets and the total buffer capacity is a well known result in literature [2–4].

As far as the interactions effects are concerned, it is possible to state that two-way interactions between P, α and N_{TOT} are statistically relevant (see also Fig. 3 and 4). A two-way interaction is significant if the combined variation of the two factors has a relevant effect on the response. The interaction between P and α shows that the system performance decreases when the number of pallets is high and all buffers are dedicated: in this case the blocking of machines is frequent and it deeply affects the line efficiency. Notice that the system with shared buffer has approximately the same efficiency value independently by how many pallet circulate in the line. On the contrary, performance decreases in the traditional system when the number of pallets is augmented. The interaction between N_{TOT} and α shows that system performance decreases when the total buffer capacity is low and all buffers are dedicated: in this case the blocking of machines is frequent due to the contemporary reduced and dedicated buffers capacity. The interaction between N_{TOT} and P is known in literature [3,9] and we do not comment more. The triple interaction among P, α and N_{TOT} results to be significant only for cycle time equal to 60 s.

The same system has been simulated also for a wider set of values to better understand the effect of the shared buffer on the system performance. Figure 5 shows the average throughput for different sharing levels of the central buffer when

Table 2. Test case: ANOVA results ($t_c = 5s$)

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Pallet number (P)	1	0.018475	0.018475	0.018475	441.97	0.000
Alpha (α)	1	0.154370	0.154370	0.154370	3693.01	0.000
Travel time (t_t)	1	0.000126	0.000126	0.000126	3.01	0.084
Total buffer capacity (N_{TOT})	1	0.077345	0.077345	0.077345	1850.32	0.000
P*α	1	0.010684	0.010684	0.010684	255.59	0.000
P*t_t	1	0.000004	0.000004	0.000004	0.09	0.762
P*N_{TOT}	1	0.017484	0.017484	0.017484	418.28	0.000
α*t_t	1	0.000068	0.000068	0.000068	1.63	0.203
α*N_{TOT}	1	0.020673	0.020673	0.020673	494.56	0.000
t_t*N_{TOT}	1	0.000063	0.000063	0.000063	1.50	0.222
P*α*t_t	1	0.000167	0.000167	0.000167	3.99	0.047
P*α*N_{TOT}	1	0.000358	0.000358	0.000358	8.57	0.004
P*t_t*N_{TOT}	1	0.000077	0.000077	0.000077	1.85	0.175
α*t_t*N_{TOT}	1	0.000005	0.000005	0.000005	0.12	0.732
P*α*t_t*N_{TOT}	1	0.000024	0.000024	0.000024	0.58	0.445
Error	224	0.009363	0.009363	0.000042		
Total	239	0.309285				

Table 3. Test case: ANOVA results ($t_c = 60s$)

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Pallet number (P)	1	0.0010176	0.0010176	0.0010176	440.48	0.000
Alpha (α)	1	0.0049118	0.0049118	0.0049118	2126.16	0.000
Travel time (t_t)	1	0.0000065	0.0000065	0.0000065	2.81	0.095
Total buffer capacity (N_{TOT})	1	0.0022927	0.0022927	0.0022927	992.46	0.000
P*α	1	0.0009786	0.0009786	0.0009786	423.60	0.000
P*t_t	1	0.0000011	0.0000011	0.0000011	0.49	0.483
P*N_{TOT}	1	0.0007292	0.0007292	0.0007292	315.65	0.000
α*t_t	1	0.0000001	0.0000001	0.0000001	0.06	0.804
α*N_{TOT}	1	0.0017108	0.0017108	0.0017108	740.57	0.000
t_t*N_{TOT}	1	0.0000018	0.0000018	0.0000018	0.78	0.380
P*α*t_t	1	0.0000014	0.0000014	0.0000014	0.60	0.438
P*α*N_{TOT}	1	0.0002936	0.0002936	0.0002936	127.10	0.000
P*t_t*N_{TOT}	1	0.0000099	0.0000099	0.0000099	4.30	0.039
α*t_t*N_{TOT}	1	0.0000009	0.0000009	0.0000009	0.40	0.525
P*α*t_t*N_{TOT}	1	0.0000013	0.0000013	0.0000013	0.55	0.461
Error	224	0.0005175	0.0005175	0.0000023		
Total	239	0.0124749				

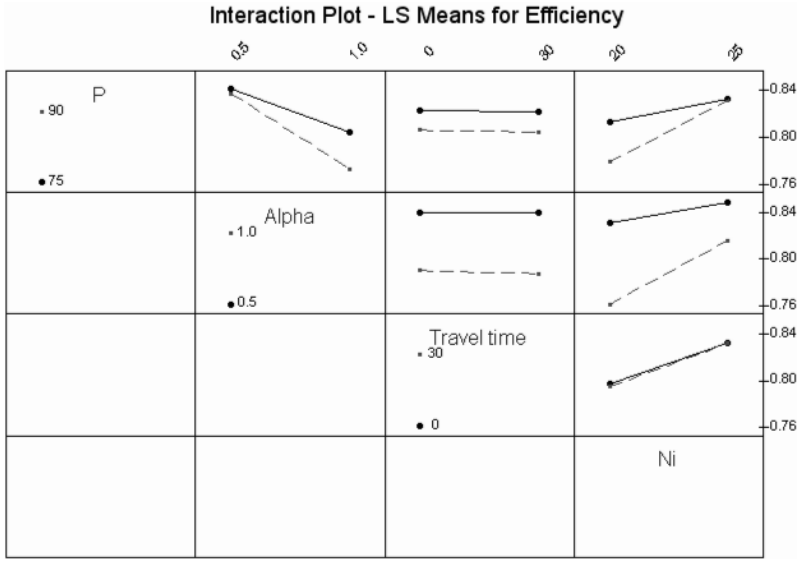


Fig. 3. Test case: interaction plot ($t_c = 5s$)

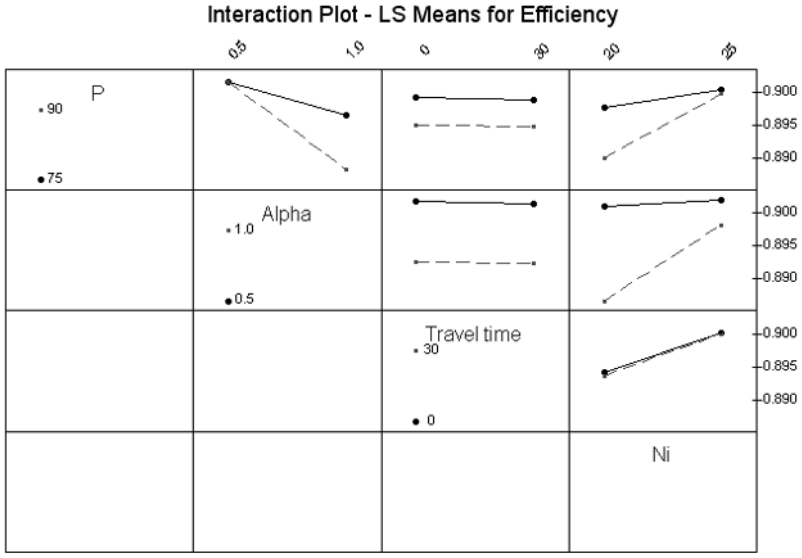


Fig. 4. Test case: interaction plot ($t_c = 60s$)

the travel time is equal to 5 s. When the number of pallets is small the shared buffer is never used and the different systems have the same performance. When the number of pallets increases, the starvation decreases and the throughput increases; however as the number of pallets increases the blocking occurs more frequently and the systems with shared buffer perform better than the traditional one (i.e. $\alpha = 1$). In more detail higher the sharing percentage is and better is the performance. After a certain value of pallets in the system the blocking penalizes the system performance

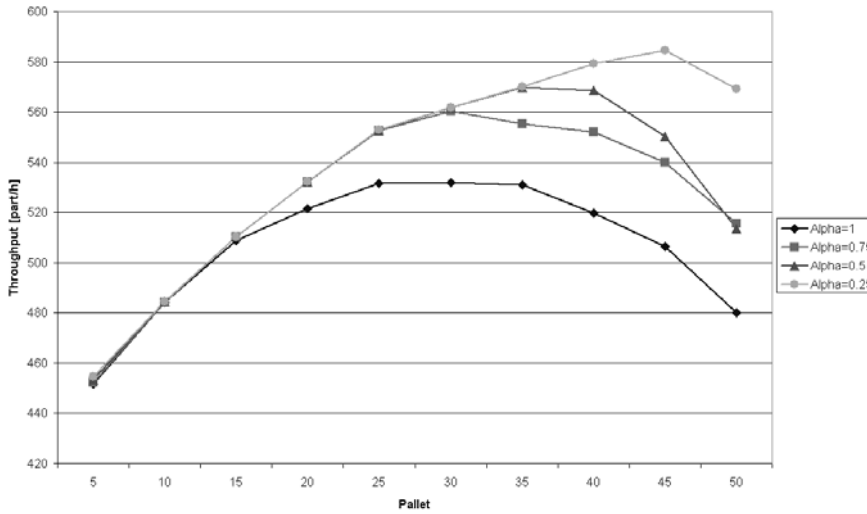


Fig. 5. Test case: average production rate (± 1.2 part/hour) vs P when $N_{TOT} = 50$ and $t_t = 5s$ for different values of α

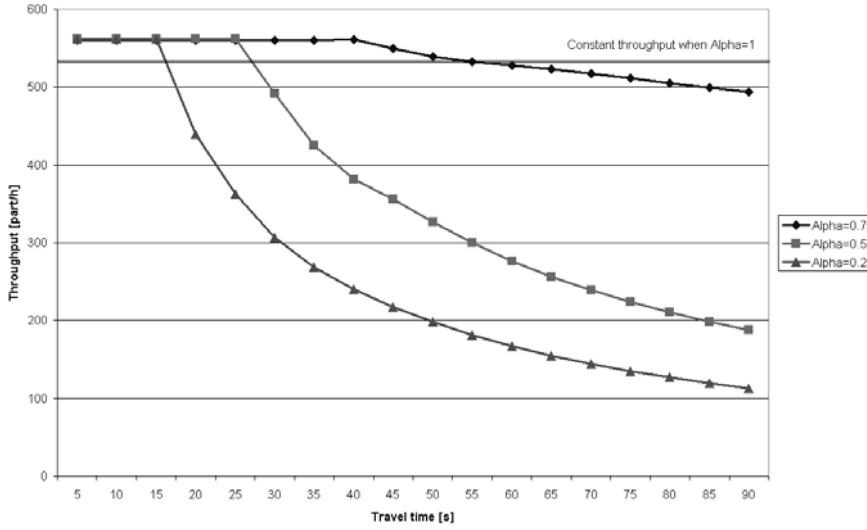


Fig. 6. Test case: average production rate (± 1.2 part/hour) vs t_t when $N_{TOT} = 50$ and $P = 30$ for different values of α

and the throughput decreases [2,3,9]. This inversion value is larger in systems with shared buffer than in traditional systems. In particular the inversion point increases as the percentage sharing of buffers increases. Thus, in order to increase the throughput the system's user could move a portion of the buffer capacity from dedicated to buffer and contemporary to increase the number of pallets.

Figure 6 shows the effect of the travel time on the average throughput. The system performance stays stable for values of the parameter travel time inferior to

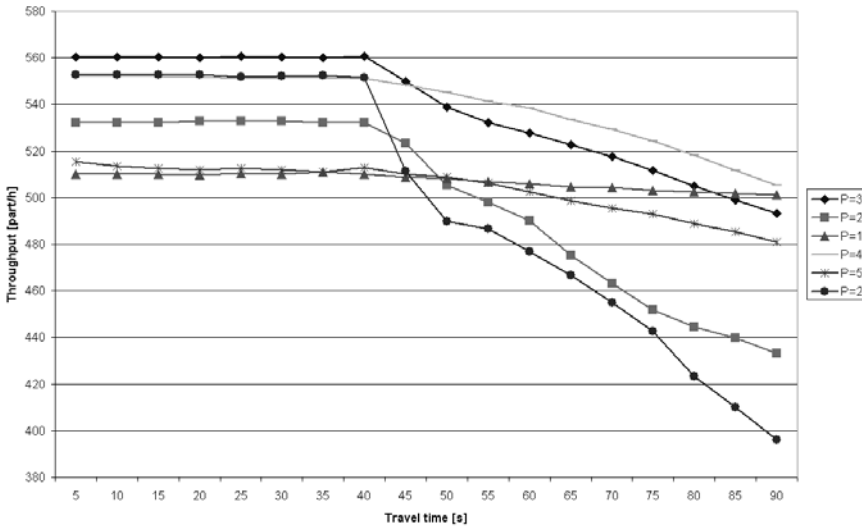


Fig. 7. Test case: average production rate (± 1.2 part/hour) vs t_t when $N_{TOT} = 50$ and $\alpha = 0.75$ for different values of P

a threshold value and deteriorates after it. In this experiment the threshold value is equal to 40 s, 25 s and 15 s for values of α equal to 0.75, 0.5 and 0.25 respectively when the number of pallets is 30. The threshold value of the travel time decreases as the percentage of shared buffer increases because there are less pallets in the dedicated buffers and starvation occurs more frequently. This effect could be compensated by increasing the number of pallets.

Figure 7 shows the effect of the travel time for the system with $\alpha = 0.5$ and different values of pallets. Notice that the loss of production after the threshold value of the travel time is larger for small number of pallets. It is worthwhile to notice that the results reported in this section are valid for closed flow lines. Open flow lines are more difficult to manage due to the large number of parts which may enter from the first machine. Indeed, if there is no limit to the number of parts entering into the system and the first machine is very efficient, it happens that all the parts just entered and processed by the first machine fills the shared buffer thus limiting the possibility to the other machines of recurring to the shared buffer. Thus, specific rules for managing the entering of parts should be designed.

3.2 Real case

In this paragraph we consider a real assembly line composed of five machines separated by buffers with limited capacity. Pallets are empties before entering into the first machine; then components are loaded on pallets until the assembled final product is obtained at the last machine of the system. The components are stored on containers located at each machine and are not modelled as customers, thus the system can be view as a flow line crossed by parts (i.e. the pallets) that visit machines in a fixed sequence. The number of pallets in the system remains constant

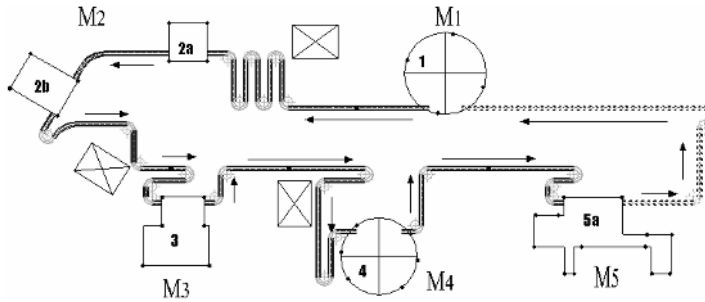


Fig. 8. Real case: lay-out of the real system

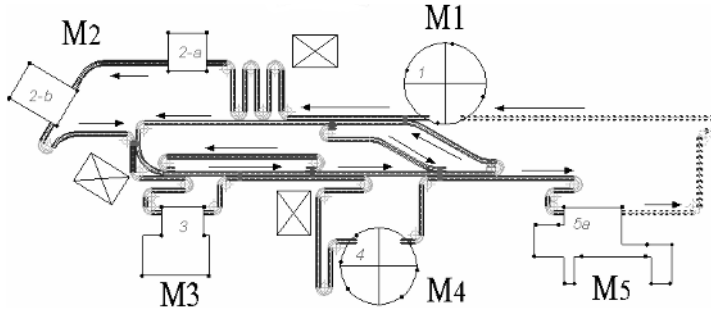


Fig. 9. Real case: lay-out of alternative 1

during the production. Machines are unreliable and characterized by different types of failures. In particular machines M_1, M_2 and M_5 can fail in three different ways while M_3 and M_4 in only one way. MTTF and MTTR for each failure type are exponentially distributed with values as reported in Table 4 and calculated by the firm. As in the real system, a machine can fail only when it is occupied by a part. The first failure type of each machine models mechanical and electronic failures in an aggregated way, and the second and third failure types of M_1, M_2 and M_5 model the emptying of component containers. Table 4 reports the failure parameters and the deterministic processing rates of machines. The BAS control rule is considered, that is machines may enter in a blocking state only after the completion of the process. A physical constraint in the lay-out does not allow changes in the portion of the system between M_5 and M_1, i.e. the buffer B_5 is dedicated and cannot be modified in its capacity. The lay-out of the system is shown in Figure 8. The real system already uses in the traditional way flexible transport modules for moving parts through the line at a constant speed of 17.6 m/min.

Among a large set of feasible solutions, two alternative reasonable systems with shared buffer are considered in the comparison with the real one. The first alternative has a shared buffer, located at the center of the line, in addition to the dedicated buffers of the real system. The increase of total buffer capacity is around 31% corresponding to an increase of approximately 44 kEuro of the total investment cost (this value has been estimated on the basis of additional conveyors, sensors, engines and control system). The second alternative has been designed

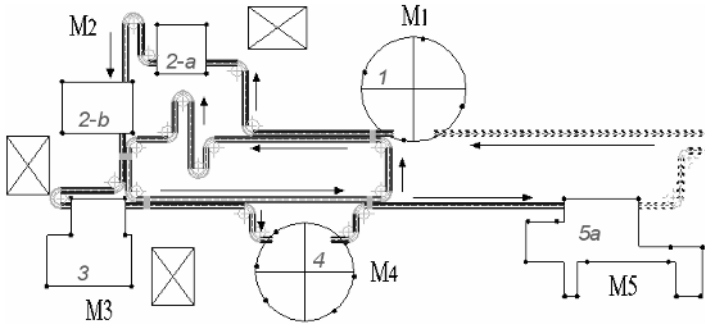


Fig. 10. Real case: lay-out of alternative 2

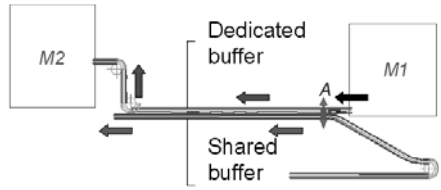


Fig. 11. Input/output into/fronm the shared buffer

with investment cost equal to that of the real line; the total buffer capacity is lower than that of the real line because of the additional costs of sensors and engines. Total buffer capacity is reported in Table 5 while the lay-outs of alternatives with shared buffer are shown in Figures 9 and 10.

In the proposed alternatives each machine is blocked only if its dedicated buffer and the common buffer are full. The mechanism of input/output into/from the shared buffer is now described referring to Figure 11. Let us consider the portion of the system between machines M_1 and M_2. Before the machine M_1 releases a processed part, the system controls the availability of space in the portion of conveyor between M_1 and M_2, i.e. in the dedicated buffer with size N_1. If there is space in the dedicated buffer the machine releases the part which then moves towards machine M_2, otherwise the system controls if the part can be introduced in the shared buffer. If there is space available in the shared buffer the machine releases the part that enters in the shared buffer, otherwise the machine is blocked until a new space becomes available in the dedicated buffer or shared buffer. This reaction

Table 4. Real case: processing rates [part/min] and MTTFs and MTTRs [min] of machines

Machine number	Processing rate	MTTF type 1	MTTR type 1	MTTF type 2	MTTR type 2	MTTF type 3	MTTR type 3
1	20.0	5.64	0.81	499.17	4.00	143.99	7.16
2	17.3	2.90	1.08	94.78	5.16	69.23	5.16
3	16.5	5.61	0.57	–	–	–	–
4	15.7	21.28	0.51	–	–	–	–
5	16.0	10.60	0.63	274.43	5.16	29.93	5.00

Table 5. Real case: buffer capacities of real system and alternatives with shared buffer

System	N_1	N_2	N_3	N_4	N_5	N_{Shared}	Total	Dedicated	Shared
Real	110	66	107	70	83	0	436	100 %	0 %
Alternative 1	110	43	92	52	83	192	572	61 %	39 %
Alternative 2	44	24	29	37	83	149	366	47 %	53 %

Table 6. Real case: comparison between real system and alternatives with shared buffer

System	Max average production rate [part/h]	Investment cost [kEuro]	Average productivity index
Real	650.9±3.6	2250	0.289
Alternative 1	677.7±3.9	2294	0.295
Alternative 2	667.3±3.1	2250	0.297

of the machine has been called as "block-and-recirculate" strategy by Tempelmeier and Kuhn in their book [9]. The point denoted with *A* in Figure 11 is the transfer point at which parts can change conveyor. The transfer point is bi-directional, that is a part is switched from the output conveyor of the machine to the shared conveyor and vice versa. Switching devices are available in the market at affordable costs and allow the machine to avoid the blocking state. An example of switching mechanism is shown in Figure 12. When a new space becomes available in the shared buffer a control rule must be defined to decide which part, if any, will access to the common area. In the proposed systems the precedence is given to the machine that has just made free the place in the shared buffer.

Each time a part must leave the shared buffer it is necessary that the part reaches the transfer point. If the shared buffer is large the time to reach the transfer point can be so high that the dedicated buffer empties and starvation thus occurs. In order to decrease this time, which is a portion of the above defined travel time, two inner alternative paths have been introduced in the first alternative (see Fig. 9). In the second alternative machines are closer and the travel time is not critical. The transfer time in which the part leaves changes the conveyor from the shared to the dedicated buffer takes few seconds.

The performance of systems has been calculated by terminating simulations of one production day because at the end of the shift the system is always emptied. The simulation model has been validated on the real production rates of seven days. Statistics have been collected and confidence intervals on production rate have been calculated at 95% confidence level. Figure 13 shows the average production rate for the analyzed systems which depends on the numbers of customers that circulate in the system [2,3,9]. As shown in Table 6, both the proposed alternative systems have productivity index (calculated as average production rate over investment cost) greater than that of the real system. In particular, at equal investment cost, the second alternative has an average production rate greater 2.5% than that in the real case. Figures 13–18 report the detailed states of machines for the real system and

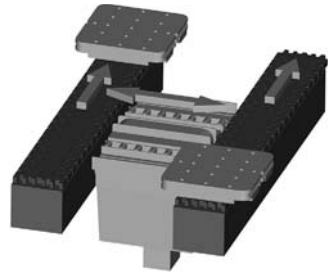


Fig. 12. Switching mechanism

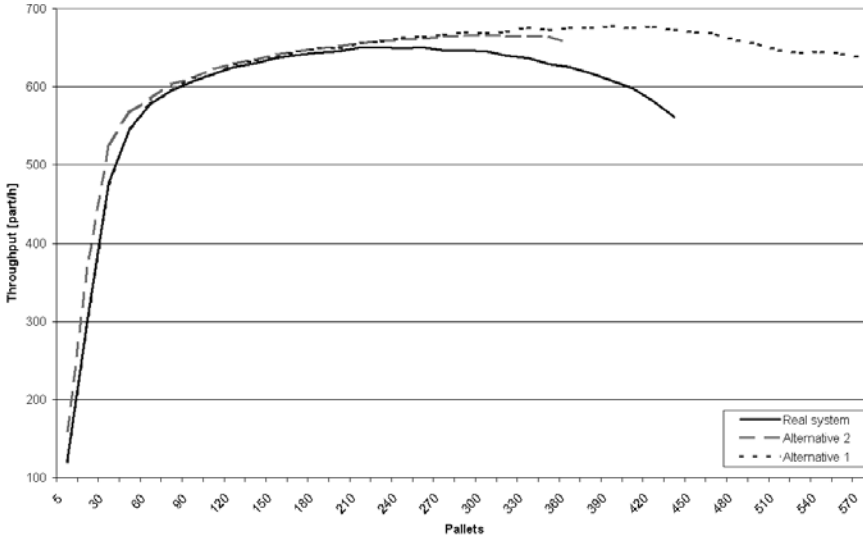


Fig. 13. Real case: average production rate vs number of pallets (± 4 part/h)

the alternative 2. It can be noticed how the blocking of machines decreases from the real system to the system with the shared buffer and starvation increases due to the fact that parts circulate in the shared buffer. However the disadvantages due to the increase of starvation do not compensate the advantages due to the reduction of blocking in the system.

A long term investment analysis must consider the Net Present Value (NPV) related to the investment, i.e. the sum of all discounted cash flows during the life of the system. The NPV considers the initial investment cost (fundamentally machines, buffers), the future discounted cash flows during the normal running of the system (revenues and production costs) and the residual value of the system after the planning horizon of the investment. In the investment analysis only the discriminating voices have to be considered. In this case the three alternative systems differ in the buffer capacity since machines remain the same. Thus the investment cost of machines is not differential and is not considered. The investment cost of buffers is different only for the alternative 1. Revenues are discriminant if the additional production capacity of the proposed alternatives is converted in additional sales.

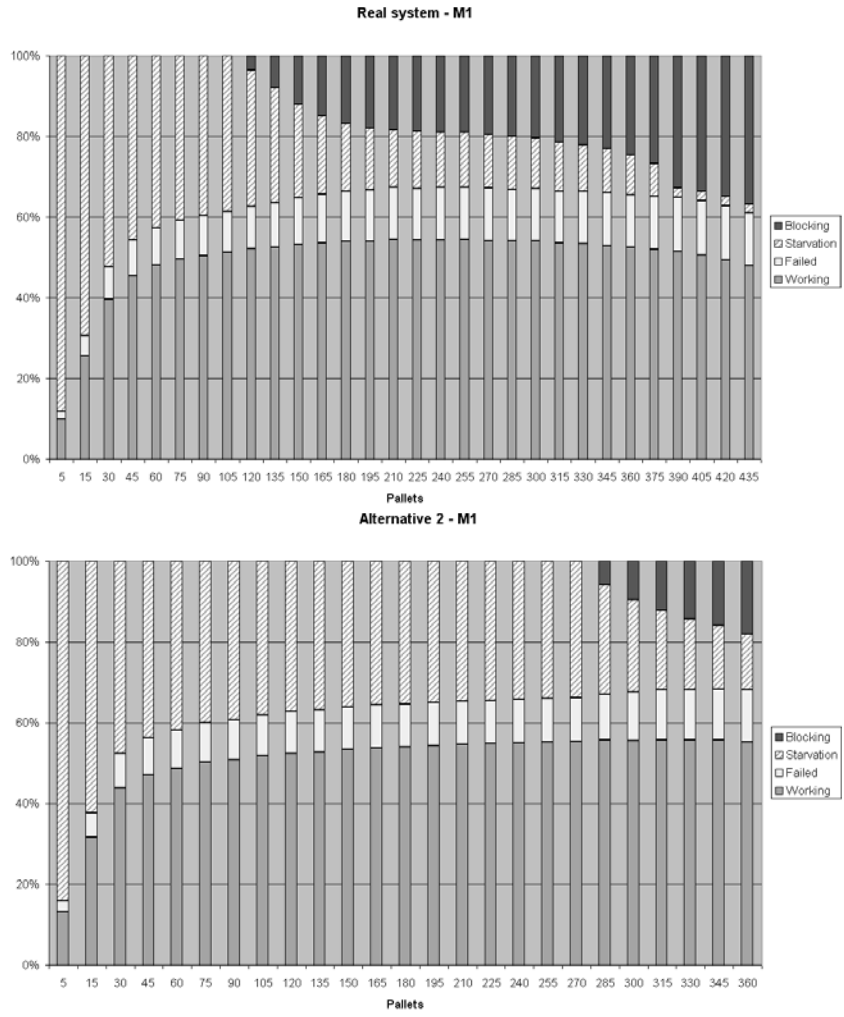


Fig. 14. Real case: Machine 1 states vs number of pallets for real system and alternative 2

A reasonable assumption is that the unit production cost does not differ because the process technology is the same for each alternative. Again, if the additional capacity is exploited in new sales the total variable production costs are a differential item. The residual value of a production system at the end of the planning horizon is very difficult to estimate. However, the difference between the initial investment with the real system is limited to 44 kEuro for the first alternative and null for the second one and we can imagine that the difference between the residual value will be smaller, so we can neglect the residual value voice in the analysis. The second alternative has the same investment cost of the real system with a higher production rate, and in this case the NPV analysis is not necessary because the real system is dominated by the alternative. For the first alternative we have to impose

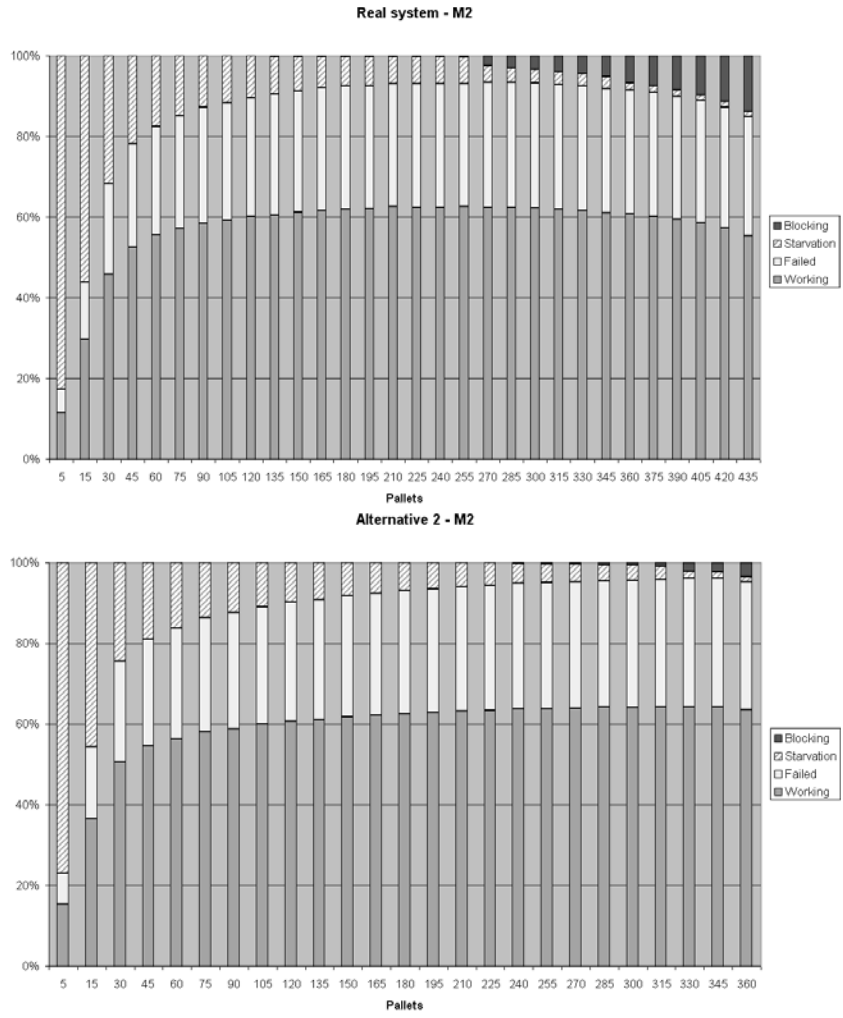


Fig. 15. Real case: Machine 2 states vs number of pallets for real system and alternative 2

some assumptions to compensate the lack of information on revenues and costs. Assuming a yearly discount rate of 0.1, a planning horizon of 5 years, 8 hours of production per day, and that all the additional capacity is sold in the market, the marginal gain (i.e. the difference between price and variable cost of the product) the product must have to compensate the additional investment of the first alternative is equal to 0.17 euro/part, corresponding approximately to 0.6 % of the market price of the product. We think that the marginal gain on the product is much larger than the calculated threshold value and therefore the proposed alternative 1 seems to be profitable. Obviously, if the additional capacity will not be exploited the real system is clearly more profitable than the first alternative, however in this situation the second alternative dominates the others.

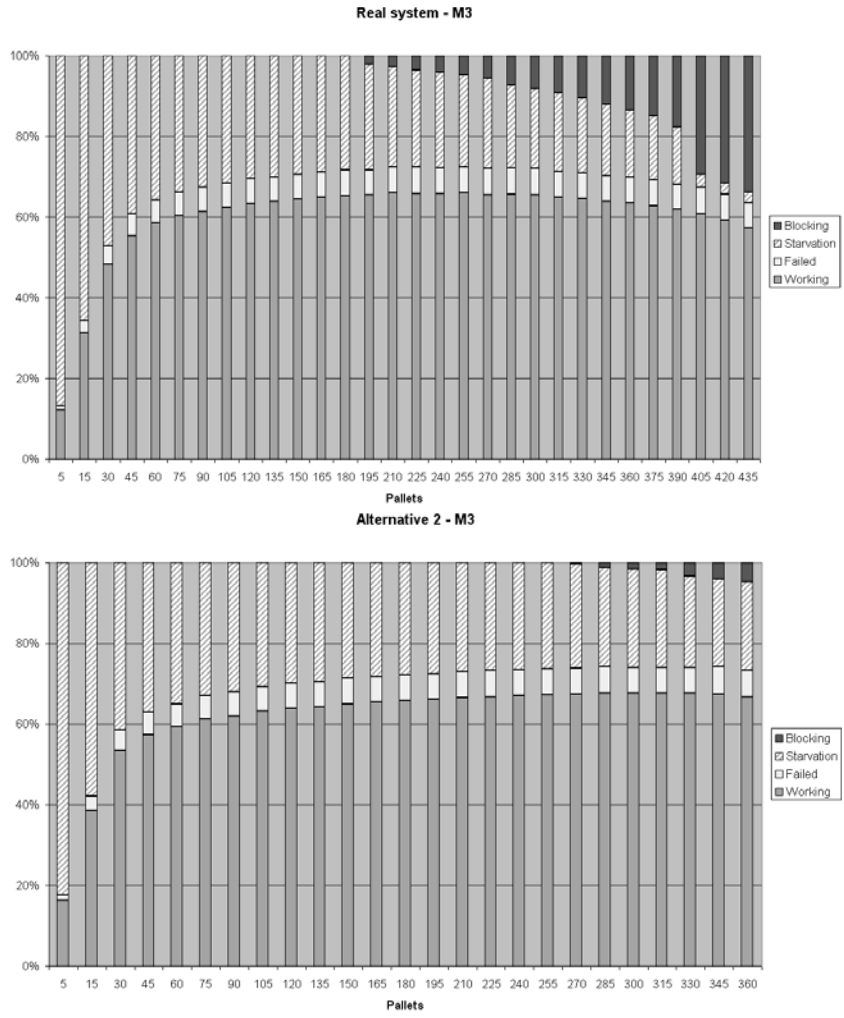


Fig. 16. Real case: Machine 3 states vs number of pallets for real system and alternative 2

4 Practical considerations

In order to introduce manufacturing flow lines with shared buffer in real shop floors several aspects, both technological and economic, have to be clarified. First of all, a necessary condition for the common buffer exploitation is that, in order to be able to dispatch parts to the different machines, parts must be tracked during their movements in the system. To do this, several technologies are available at low costs. Lasers markers can sculpture codes, easily readable by optical devices, on metal components in a fast and cheap way. Standard devices like chips can save information and exchange it with the system supervisor. Also radio frequency technology is now ready to be used in shop floors to exchange information without the limiting

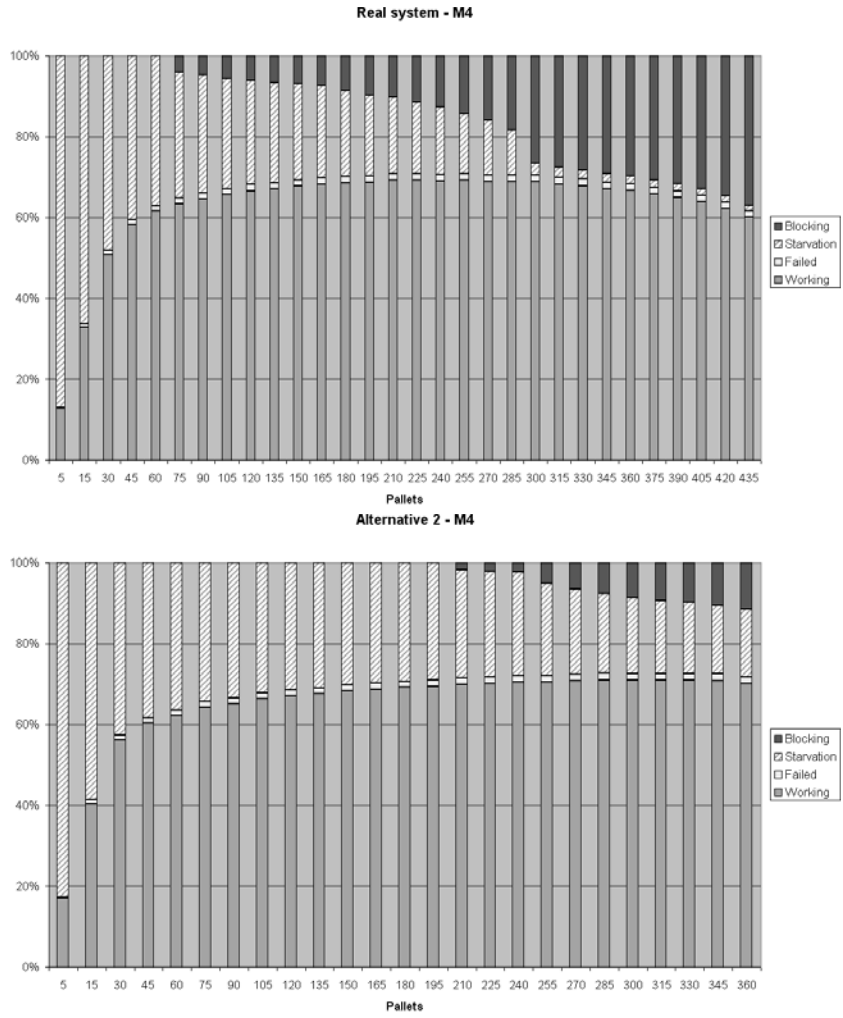


Fig. 17. Real case: Machine 4 states vs number of pallets for real system and alternative 2

constraint of designing control points in the system. Therefore, traceability of parts in the system does not seem to be an obstacle in future applications.

Conveyors seem to be a good and consolidated solution to move parts from machines to the common buffer and vice versa. However, other devices should be investigated such as robot manipulators that can move parts through the system. The main advantage of manipulators is their flexibility since they can be adapted to different situations (e.g. adaptation to react to changes in the lay-out of the system) by simply re-programming them. The main drawback of manipulators is related to their investment cost and the skills needed to instruct them. Shuttles and AGV (Automated Guided Vehicle) represents a traditional solution to part movement in

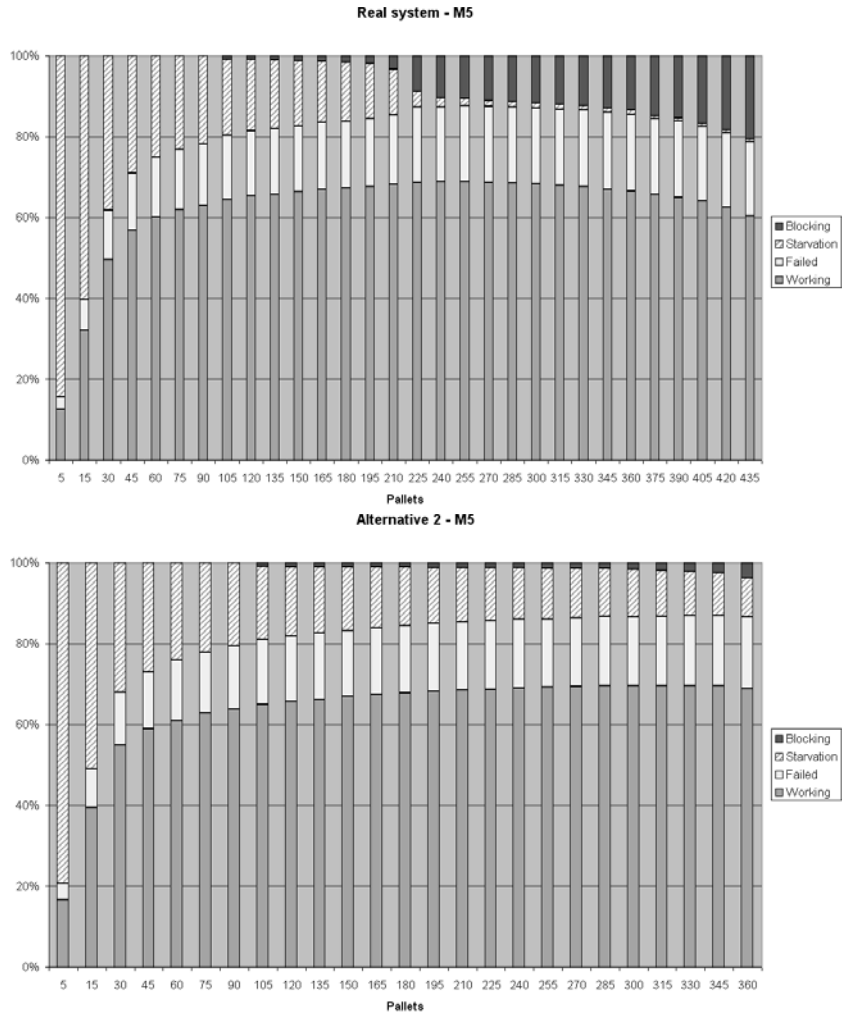


Fig. 18. Real case: Machine 5 states vs number of pallets for real system and alternative 2

Flexible Manufacturing Systems, which represent the first case of shared buffer in automated manufacturing systems.

Another important aspect that is normally taken into consideration in the design phase of a flow line is the floor space occupied by the system. Theoretically, it is necessary an additional space to locate the common buffer in a manufacturing flow line. However, it is also true that the space occupied by dedicated buffers decreases and consequently the machines of the line are closer. Therefore, it is not possible a priori to say anything about the effects of the shared buffer on the occupied floor space since this aspect is closely connected to the lay-out of the real system.

5 Conclusions and future developments

The paper addresses the problem of fully using buffer spaces in flow lines. The idea is to exploit recent technological devices to move in reasonable time pieces from a machine to a common buffer area of the system and vice versa. In such a way machines can avoid their blocking moving pieces to the shared buffer area. The decrease of blocking in flow lines has a positive impact on their production rate. The numerical analysis reported in the paper demonstrates the validity of the idea pointing out also the factors that affect the improvement of the proposed system architecture in terms of productivity.

In conclusion, several practical aspects have to be investigated before to state that shared buffers can be successfully adopted in real manufacturing flow lines, however the first results shown in this paper and the technologies now available motivate further research in this direction. Ongoing research is dedicated to identify the potential sectors for practical applications of the new concepts proposed in this paper. Then, further research will focus on new key-issues never addressed in literature and introduced by the architecture with the shared buffer:

- Allocation of dedicated and shared buffers. Traditionally only capacities of dedicated buffers have been considered in the design phase of manufacturing flow lines. In our opinion the buffer allocation problem in the case of shared buffer will be easier in comparison than the traditional one because the new system architecture is more robust, i.e. the system performance is stable in several conditions and do not decay after some changes in the design of the line.
- Performance evaluation of flow lines with shared buffer. New analytical methods are necessary to estimate performance of new system architectures. The method of Tempelmeier et al. [10], originally developed to evaluate the performance of Flexible Manufacturing Systems with blocking, could be adopted also for flow lines with shared buffer. This method is being tested in terms of accurateness of provided results.
- Management of flow lines with shared buffer. New dispatching rules could be necessary to avoid deadlock in the new system architecture when pieces converge to the same area coming from different positions.

References

1. Dallery Y, Gershwin SB (1992) Manufacturing flow line systems: a review of models and analytical results. Queueing Systems 12: 3–94
2. Dallery Y, Towsley D (1991) Symmetry property of the throughput in closed tandem queueing networks with finite capacity. Operations Research 10(9): 541–547
3. Frein Y, Commault C, Dallery Y (1996) Modeling and analysis of closed-loop production lines with unreliable machines and finite buffers. IIE Transactions 28: 545–554
4. Gershwin SB (1994) Manufacturing systems engineering. PTR Prentice Hall, New Jersey
5. Gershwin SB, Schor JE (2000) Efficient algorithms for buffer space allocation. Annals of Operations Research 93: 91–116

6. Law AM, Kelton WD (2000) Simulation modelling and analysis. McGraw–Hill, New York
7. Shantikumar JG, Yao DD (1989) Queueing networks with finite buffers. In: Perros HG, Altiok T (eds) chapter Monotonicity and concavity properties in cyclic queueing networks with finite buffers, pp. 325–344. North Holland, Amsterdam
8. Tempelmeier H (2003) Practical considerations in the optimization of flow production systems. International Journal of Production Research 41(1): 149–170
9. Tempelmeier H, Kuhn H (1993) Flexible manufacturing systems – Decision support for design and operation. Wiley, New York
10. Tempelmeier H, Kuhn H, Tetzlaff U (1989) Performance evaluation of flexible manufacturing systems with blocking. International Journal of Production Research, 27(11): 1963–1979

Integrated quality and quantity modeling of a production line*

Jongyoon Kim and Stanley B. Gershwin

Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139-4307, USA (e-mail: gershwin@mit.edu)

Abstract. During the past three decades, the success of the Toyota Production System has spurred much research in manufacturing systems engineering. Productivity and quality have been extensively studied, but there is little research in their intersection. The goal of this paper is to analyze how production system design, quality, and productivity are inter-related in small production systems. We develop a new Markov process model for machines with both quality and operational failures, and we identify important differences between types of quality failures. We also develop models for two-machine systems, with infinite buffers, buffers of size zero, and finite buffers. We calculate total production rate, effective production rate (ie, the production rate of good parts), and yield. Numerical studies using these models show that when the first machine has quality failures and the inspection occurs only at the second machine, there are cases in which the effective production rate *increases* as buffer sizes increase, and there are cases in which the effective production rate *decreases* for larger buffers. We propose extensions to larger systems.

Keywords: Quality, Productivity, Manufacturing system design

1 Introduction

1.1 Motivation

During the past three decades, the success of the Toyota Production System has spurred much research in manufacturing systems design. Numerous research papers have tried to explore the relationship between production system design and

* We are grateful for support from the Singapore-MIT Alliance, the General Motors Research and Development Center, and PSA Peugeot-Citroën.
Correspondence to: S.B. Gershwin

productivity, so that they can show ways to design factories to produce more products on time with less resources (such as people, material, and space). On the other hand, topics in quality research have captured the attention of practitioners and researchers since the early 1980s. The recent popularity of Statistical Quality Control (SQC), Total Quality Management (TQM), and Six Sigma have demonstrated the importance of quality.

These two fields, productivity and quality, have been extensively studied and reported separately both in the manufacturing systems research literature and the practitioner literature, but there is little research in their intersection. The need for such work was recently described by authors from the GM Corporation based on their experience [13]. All manufacturers must satisfy these two requirements (high productivity and high quality) at the same time to maintain their competitiveness.

Toyota Production System advocates admonish factory designers to combine inspections with operations. In the Toyota Production System, the machines are designed to detect abnormalities and to stop automatically whenever they occur. Also, operators are equipped with means of stopping the production flow whenever they note anything suspicious. (They call this practice *jidoka*.) Toyota Production System advocates argue that mechanical and human jidoka prevent the waste that would result from producing a series of defective items. Therefore jidoka is a means to improve quality and increase productivity at the same time [23], [24]. But this statement is arguable: quality failures are often those in which the quality of each part is independent of the others. This is the case when the defect takes place due to common (or chance or random) causes of variations [16]. In this case, there is no reason to stop a machine that has made a bad part because there is no reason to believe that stopping it will reduce the number of bad parts in the future. In this case, therefore, stopping the operation does not influence quality but it does reduce productivity. On the other hand, when quality failures are those in which once a bad part is produced, all subsequent parts will be bad until the machine is repaired (due to special or assignable or systematic causes of variations) [16], catching bad parts and stopping the machine as soon as possible is the best way to maintain high quality and productivity.

Non-stock or *lean* production is another popular buzzword in manufacturing systems engineering. Some lean manufacturing professionals advocate reducing inventory on the factory floor since the reduction of work-in-process (WIP) reveals the problems in the production lines [3]. Thus, it can help improve product quality. It is true in some sense: less inventory reduces the time between making a defect and identifying the defect. But it is also true that productivity would diminish significantly without stock [5]. Since there is a tradeoff, there must be optimal stock levels that are specific to each manufacturing environment. In fact, Toyota recently changed their view on inventory and are trying to re-adjust their inventory levels [9].

What is missing in discussions of factory design, quality, and productivity is a quantitative model to show how they are inter-related. Most of the arguments about this are based on anecdotal evidence or qualitative reasoning that lack a sound scientific quantitative foundation. The research described here tries to establish such a foundation to investigate how production system design and operation influence

productivity and product quality by developing conceptual and computational models of two-machine-one-buffer systems and performing numerical experiments.

1.2 Background

1.2.1 Quality models. There are two extreme kinds of quality failures based on the characteristics of variations that cause the failures. In the quality literature, these variations are called *common* (or chance or random) cause variations and *assignable* (or special or unusual) cause variations [18].

Figure 1 shows the types of quality failures and variations. Common cause failures are those in which the quality of each part is independent of the others. Such failures occur often when an operation is sensitive to external perturbations like defects in raw material or when the operation uses a new technology that is difficult to control. This is inherent in the design of the process. Such failures can be represented by independent Bernoulli random variables, in which a binary random variable, which indicates whether or not the part is good, is chosen each time a part is operated on. A good part is produced with probability π, and a bad part is produced with probability $1 - \pi$. The occurrence of a bad part implies nothing about the quality of future parts, so no permanent changes can have occurred in the machine. For the sake of clarity, we call this a *Bernoulli-type quality failure*. Most of the quantitative literature on inspection allocation assumes this kind of quality failure [21]. In this case, if bad parts are destined to be scrapped, it is useful to catch them as soon as possible because the longer before they are scrapped, the more they consume the capacity of downstream machines. However, there is no reason to stop a machine that has produced a bad part due to this kind of failure.

The quality failures due to assignable cause variations are those in which a quality failure only happens after a change occurs in the machine. In that case, it is very likely that once a bad part is produced, all subsequent parts will be bad until the machine is repaired. Here, there is much more incentive to catch defective parts and stop the machine quickly. In addition to minimizing the waste of downstream capacity, this strategy minimizes the further production of defective parts. For this kind of quality failure, there is no inherent measure of yield because the fractions

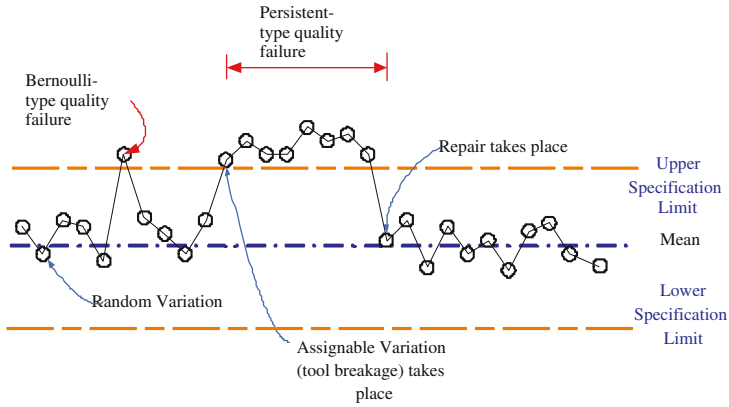


Fig. 1. Types of quality failures

of parts that are good and bad depend on how soon bad parts are detected and how quickly the machine is stopped for repair. In this paper, we call this a *persistent-type quality failure*. Most quantitative studies in Statistical Quality Control (SQC) are dedicated to finding efficient inspection policies (sampling interval, sample size, control limits, and others) to detect this type of quality failure [26].

In reality, failures are mixtures of Bernoulli-type quality failures and persistent-type quality failures. It can be argued that the quality strategy of the Toyota Production System [17], in which machines are stopped as soon as a bad part is detected, is implicitly based on the assumption of the persistent-type quality failure. In this paper, we focus on persistent failures.

1.2.2 System yield. System yield is defined here as the fraction of input to a system that is transformed into output of acceptable quality. This is an important metric because customers observe the quality of products only after all the manufacturing processes are done and the products are shipped. The system yield is a complex function of how the factory is designed and operated, as well as of the characteristics of the machines. Some of influencing factors include individual operation yields, inspection strategies, operation policies, buffer sizes, and other factors. Comprehensive approaches are needed to manage system yield effectively. This research aims to develop mathematical models to show how the system yield is influenced by these factors.

1.2.3 Quality improvement policy. System yield is a complex function of various factors such as inspection, individual operation yields, buffer size, operation policies, and others. There are many ways to affect the system yield. Inspection policy has received the most attention in the literature. Research on inspection policies can be divided into optimizing inspection parameters at a single station and the inspection station allocation problem. The former issue has been investigated extensively in the SQC literature [26]. Here, optimal SQC parameters such as control limits, sampling size, and frequency are sought for an optimal balance between the inspection cost and the cost of quality. The latter research looks for the optimal distributions of inspection stations along production lines [21].

Improving individual operation yield is another important way to increase the system yield. Many studies in this field try to stabilize the process either by finding root causes of variation and eliminating them or by making the process insensitive to external noise. The former topic has numerous qualitative research papers in the fields of Total Quality Management (TQM) [2] and Six Sigma [19]. Quantitative research is more oriented toward the latter topic. Robust engineering [20] is an area that has gained substantial attention.

It has been argued that inventory reduction is an effective means to improve system yield. Many lean manufacturing specialists have asserted that less inventory on the factory floor reveals problems in the manufacturing lines more quickly and helps quality improvement activities [1, 17].

There also have been investigations to explain the relationship between plant layout design and quality [7]. They argue that U-shaped lines are better than straight lines for producing higher quality products since there are more points of contact between operators. There is also less material movement, and there are other reasons.

There are many ways to improve system yield, but using only a single method will give limited gains. The effectiveness of each method is greatly dependent on the details of the factory. Thus, there is need to determine which method or which combination of methods is most effective in each case. The quantitative tools that will be developed from this research can help fulfill this need.

1.3 Outline

In Section 2 we introduce the structure of the modeling techniques used in this paper. We present modeling, solution techniques, and validation of the 2-machine-1-finite buffer case in Section 3. Discussions on the behavior of a production line based on numerical experiments are provided in Section 5. A future research plan is shown in Section 6. Parameters of many of the systems studied numerically here, and details of the analytical solution of the two-machine line, can be found in the appendices.

2 Mathematical models

2.1 Single machine model

There are many possible ways to characterize a machine for the purpose of simultaneously studying quality and quantity issues. Here, we model a machine as a discrete state, continuous time Markov process. Material is assumed continuous, and μ_i is the speed at which Machine i processes material while it is operating and not constrained by the other machine or the buffer. It is a constant, in that μ_i does not depend on the repair state of the other machine or the buffer level.

Figure 2 shows the proposed state transitions of a single machine with persistent-type quality failures. In the model, the machine has three states:

* State 1: The machine is operating and producing good parts.
* State -1: The machine is operating and producing bad parts, but the operator does not know this yet.
* State 0: The machine is not operating.

The machine therefore has two different failure modes (i.e. transition to failure states from state 1):

* *Operational failure*: transition from state 1 to state 0. The machine stops producing parts due to failures like motor burnout.
* *Quality failure*: transition from state 1 to state -1. The machine stops producing good parts (and starts producing bad parts) due to a failure like a sudden tool damage.

When a machine is in state 1, it can fail due to a non-quality related event. It goes to state 0 with transition probability rate p. After that an operator fixes it, and the machine goes back to state 1 with transition rate r. Sometimes, due to an assignable cause, the machine begins to produce bad parts, so there is a transition

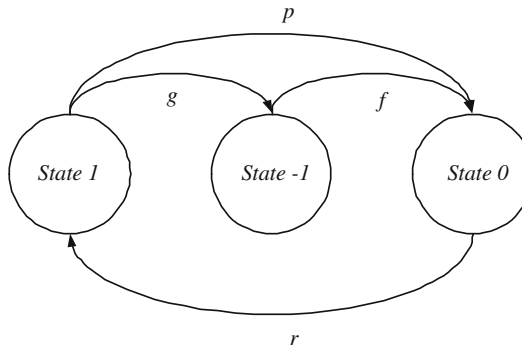


Fig. 2. States of a machine

from state 1 to state -1 with a probability rate g. Here g is the reciprocal of the *Mean Time to Quality Failure (MTQF)*. A more stable operation leads to a larger MTQF and a smaller g.

The machine, when it is in state -1, can be stopped for two reasons: it may experience the same kind of operational failure as it does when it is in state 1; and the operator may stop it for repair when he learns that it is producing bad parts. The transition from state -1 to state 0 occurs at probability rate $f = p + h$ where h is the reciprocal of the *Mean Time To Detect (MTTD)*. A more reliable inspection leads to a shorter MTTD and a larger f. (The detection can take place elsewhere, for example at a remote inspection station.) Note that this implies that $f > p$. Here, for simplicity, we assume that whenever a machine is repaired, it goes back to state 1. All the indicated transitions are assumed to follow exponential distributions.

Single machine analysis. To determine the production rate of a single machine, we first determine the steady-state probability distribution. This is calculated based on the probability balance principle: the probability of leaving a state is the same as the probability of entering that state. We have

$$(g + p)P(1) = rP(0) \tag{1}$$

$$fP(-1) = gP(1) \tag{2}$$

$$rP(0) = pP(1) + fP(-1) \tag{3}$$

The probabilities must also satisfy the normalization equation:

$$P(0) + P(1) + P(-1) = 1 \tag{4}$$

The solution of (1)–(4) is

$$P(1) = \frac{1}{1 + (p + g)/r + g/f} \tag{5}$$

$$P(0) = \frac{(p + g)/r}{1 + (p + g)/r + g/f} \tag{6}$$

$$P(-1) = \frac{g/f}{1 + (p + g)/r + g/f} \tag{7}$$

The *total production rate*, including good and bad parts, is

$$P_T = \mu(P(1) + P(-1)) = \mu \frac{1 + g/f}{1 + (p+g)/r + g/f} \tag{8}$$

The *effective production rate*, the production rate of good parts only, is

$$P_E = \mu P(1) = \mu \frac{1}{1 + (p+g)/r + g/f} \tag{9}$$

The *yield* is

$$\frac{P_E}{P_T} = \frac{P(1)}{P(1) + P(-1)} = \frac{f}{f+g} \tag{10}$$

2.2 2-machine-1-buffer continuous model

A flow (or transfer) line is a manufacturing system with a very special structure. It is a linear network of service stations or machines (M_1, M_2, ..., M_k) separated by buffer storages (B_1, B_2, ..., B_{k-1}). Material flows from outside the system to M_1, then to B_1, then to M_2, and so forth until it reaches M_k, after which it leaves. Figure 3 depicts a flow line. The rectangles represent machines and the circles represent buffers.

Fig. 3. Five-machine flow line

2-machine-1-buffer (2M1B) models should be studied first. Then a decomposition technique, that divides a long transfer line into multiple 2-machine-1-buffer models, could be developed. (See [14].) Among the various modeling techniques for the 2M1B case, including deterministic, exponential, and continuous models, the continuous material line model is used for this research because it can handle deterministic but different operation times at each operation. This is an extension of the continuous material serial line modeling of [10] by adding another machine failure state. Figure 4 shows the 2M1B continuous model where the machines, buffer and discrete parts are represented as valves, a tank, and a continuous fluid.

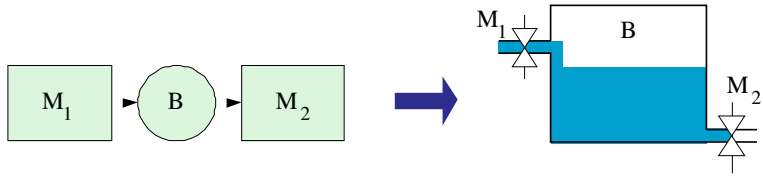


Fig. 4. Two-machine-one-buffer continuous model

We assume that an inexhaustible supply of workpieces is available upstream of the first machine in the line, and an unlimited storage area is present downstream

of the last machine. Thus, the first machine is never starved, and the last machine is never blocked. Also, failures are assumed to be operation dependent (ODF).

Finally, we assume that each machine works on a different feature. For example, the two machines may be making two different holes. We do not consider cases where the both machines work on the same hole, in which the first machine does a roughing operation and the second does a finishing operation. This allows us to assume that the failures of the two machines are independent.

2.3 Infinite buffer case

An infinite buffer case is a special 2M1B line in which the size of the buffer (B) is infinite. This is an extreme case in which the first machine (M_1) never suffers from blockage. To derive expressions for the total production rate and the effective production rate, we observe that when there is infinite buffer capacity between two machines (M_1, M_2), the total production rate of the 2M1B system is a minimum of the total production rates of M_1 and M_2. The total production rate of machine i is given by (8), so the total production rate of the 2M1B system is

$$P_T^\infty = \min\left[\frac{\mu_1(1+g_1/f_1)}{1+(p_1+g_1)/r_1+g_1/f_1}, \frac{\mu_2(1+g_2/f_2)}{1+(p_2+g_2)/r_2+g_2/f_2}\right] \tag{11}$$

The probability that machine M_i does not add non-conformities is

$$Y_i = \frac{P_i(1)}{P_i(1)+P_i(-1)} = \frac{f_i}{f_i+g_i} \tag{12}$$

Since there is no scrap and rework in the system, the system yield becomes

$$\frac{f_1 f_2}{(f_1+g_1)(f_2+g_2)} \tag{13}$$

As a result, the effective production rate is

$$P_E^\infty = \frac{f_1 f_2}{(f_1+g_1)(f_2+g_2)} P_T^\infty \tag{14}$$

The effective production rate evaluated from (14) has been compared with a discrete-event, discrete-part simulation. Table 1 shows good agreement. The parameters for these cases are shown in Appendix B.

As indicated in Section 2.1, the detection of quality failures due to machine M_1 need not occur at that machine. For example, the inspection of the feature that M_1 works on could take place at an inspection station at M_2, and this inspection could trigger a repair of M_1. (We call this *quality information feedback*. See Section 4.) In that case, the MTTD of M_1 (and therefore f_1) will be a function of the amount of material in the buffer. We return to this important case in Section 4.

2.4 Zero buffer case

The zero buffer case is one in which there is no buffer space between the machines. This is the other extreme case where blockage and starvation take place most frequently.

Table 1. Validation of infinite buffer case

Case #	$P_E^{\infty}(Analytic)$	$P_E^{\infty}(Simulation)$	%Difference
1	0.762	0.761	0.17
2	0.708	0.708	0.00
3	0.657	0.657	0.00
4	0.577	0.580	−0.50
5	0.527	0.530	−0.42
6	0.745	0.745	0.01
7	0.762	0.760	0.30
8	1.524	1.522	0.14
9	0.762	0.762	0.00
10	1.524	1.526	−0.13

In the zero-buffer case in which machines have different operation times, whenever one of the machines stops, the other one is also stopped. In addition, when both of them are working, the production rate is $\min[\mu_1, \mu_2]$. Consider a long time interval of length T during which M_1 fails m_1 times and M_2 fails m_2 times. If we assume that the average time to repair M_1 is $1/r_1$ and the average time to repair M_2 is $1/r_2$, then the total system down time will be close to $D = \frac{m_1}{r_1} + \frac{m_2}{r_2}$. Consequently, the total up time will be approximately

$$U = T - D = T - (\frac{m_1}{r_1} + \frac{m_2}{r_2}) \tag{15}$$

Since we assume operation-dependent failures, the rates of failure are reduced for the faster machine. Therefore,

$$p_i^b = p_i \frac{\min(\mu_1, \mu_2)}{\mu_i},\ g_i^b = g_i \frac{\min(\mu_1, \mu_2)}{\mu_i},\ \text{and}\ f_i^b = f_i \frac{\min(\mu_1, \mu_2)}{\mu_i} \tag{16}$$

The reduction of p_i is explained in detail in [10]. The reductions of g_i and f_i are done for the same reasons.

Table 2 lists the possible working states α_1 and α_2 of M_1 and M_2. The third column is the probability of finding the system in the indicated state. The fourth and fifth columns indicate the expected number of transitions to down states during the time interval from each of the states in column 1.

Table 2. Zero-buffer states, probabilities, and expected numbers of events

α_1	α_2	Probability $\pi(\alpha_1, \alpha_2)$	$Em_1(\alpha_1, \alpha_2)$	$Em_2(\alpha_1, \alpha_2)$
1	1	$\frac{f_1^b}{f_1^b+g_1^b}\frac{f_2^b}{f_2^b+g_2^b}$	$p_1^b U\pi(1,1)$	$p_2^b U\pi(1,1)$
1	−1	$\frac{f_1^b}{f_1^b+g_1^b}\frac{g_2^b}{f_2^b+g_2^b}$	$p_1^b U\pi(1,-1)$	$f_2^b U\pi(1,-1)$
−1	1	$\frac{g_1^b}{f_1^b+g_1^b}\frac{f_2^b}{f_2^b+g_2^b}$	$f_1^b U\pi(-1,1)$	$p_2^b U\pi(-1,1)$
−1	−1	$\frac{g_1^b}{f_1^b+g_1^b}\frac{g_2^b}{f_2^b+g_2^b}$	$f_1^b U\pi(-1,-1)$	$f_2^b U\pi(-1,-1)$

From Table 2, the expectations of m_1 and m_2 are

$$Em_1 = \sum_{\alpha_1=-1}^{1} \sum_{\alpha_2=-1}^{1} Em_1(\alpha_1, \alpha_2) = \frac{U f_1^b (p_1^b + g_1^b)}{f_1^b + g_1^b} \tag{17}$$

$$Em_2 = \sum_{\alpha_1=-1}^{1} \sum_{\alpha_2=-1}^{1} Em_2(\alpha_1, \alpha_2) = \frac{U f_2^b (p_2^b + g_2^b)}{f_2^b + g_2^b}$$

By plugging them into equation (15), we find total production rate:

$$P_T^0 = \frac{\min[\mu_1, \mu_2]}{1 + \frac{f_1^b (p_1^b + g_1^b)}{r_1 (f_1^b + g_1^b)} + \frac{f_2^b (p_2^b + g_2^b)}{r_2 (f_2^b + g_2^b)}} \tag{18}$$

The effective production rate is

$$P_E^0 = \frac{f_1^b f_2^b}{(f_1^b + g_1^b)(f_2^b + g_2^b)} P_T^0 \tag{19}$$

The comparison with simulation is shown in in Table 3. The parameters of the cases are shown in Appendix B.

Table 3. Zero buffer case

Case #	$P_E^0(Analytic)$	$P_E^0(Simulation)$	%Difference
1	0.657	0.662	−0.73
2	0.620	0.627	−1.15
3	0.614	0.621	−1.03
4	0.529	0.534	−0.99
5	0.480	0.484	−0.77
6	0.647	0.651	−0.57
7	0.706	0.712	−0.91
8	1.377	1.406	−2.10
9	0.706	0.711	−0.77
10	1.377	1.380	−0.22

3 2-machine-1-finite-buffer line

The two-machine line is the simplest non-trivial case of a production line. In the existing literature on the performance evaluation of systems in which quality is not considered, two-machine lines are used in decomposition approximations of longer lines (see [10]).

We define the model here and show the solution technique in Appendix A.

3.1 State definition

The state of the 2M1B line is defined as (x, α_1, α_2) where

* x: the total amount of material in buffer B, $0 \leq x \leq N$,
* α_1: the state of M_1. ($\alpha_1 = -1, 0$, or 1),
* α_2: the state of M_2. ($\alpha_2 = -1, 0$, or 1)

The parameters of machine M_i are $\mu_i, r_i, p_i, f_i, g_i$ and the buffer size is N.

3.2 Model development

3.2.1 Internal transition equations. When buffer B is neither empty nor full, its level can rise or fall depending on the states of adjacent machines. Since it can change only a small amount during a short time interval, it is reasonable to use a continuous probability density $f(x, \alpha_1, \alpha_2)$ and differential equations to describe its behavior. The probability of finding both machines at state 1 with a storage level between x and $x + \delta x$ at time $t + \delta t$ is given by $f(x, 1, 1, t + \delta t)\delta x$, where

$$f(x,1,1,t+\delta t) = \{1-(p_1+g_1+p_2+g_2)\delta t\}f(x+(\mu_2-\mu_1)\delta t,1,1) + r_2\delta t f(x-\mu_1\delta t,1,0) + r_1\delta t f(x+\mu_2\delta t,0,1) + o(\delta t) \quad (20)$$

Except for the factor of δx, the first term is the probability of transition from between $(x + (\mu_2 - \mu_1)\delta t, 1, 1)$ and $(x + (\mu_2 - \mu_1)\delta t + \delta x, 1, 1)$ at time t to between $(x, 1, 1)$ and $(x + \delta x, 1, 1)$ at time $t + \delta t$. This is because

* The probability of neither machine failing between t and $t + \delta t$ is

$$\{1-(p_1+g_1)\delta t\}\{1-(p_2+g_2)\delta t\} \simeq \{1-(p_1+g_1+p_2+g_2)\delta t\} \quad (21)$$

* If there are no failures between t and $t + \delta t$ and the buffer level is between x and $x + \delta x$ at time $t + \delta t$, then it could only have been between $x + (\mu_2 - \mu_1)\delta t$ and $x + (\mu_2 - \mu_1)\delta t + \delta x$ at time t.

The other terms, which represent the probabilities of transition from (1) machine states (1,0) with buffer level between $x - \mu_1\delta t$ and $x - \mu_1\delta t + \delta x$ and (2) machine states (0,1) with buffer level between $x + \mu_2\delta t$ and $(x + \mu_2\delta t + \delta x$ can be found similarly. No other transitions are possible. After linearizing, and letting $\delta t \to 0$, this equation becomes

$$\frac{\partial f(x,1,1)}{\partial t} = (\mu_2-\mu_1)\frac{\partial f(x,1,1)}{\partial x} - (p_1+g_1+p_2+g_2)f(x,1,1) + r_2 f(x,1,0) + r_1 f(x,0,1) \quad (22)$$

In steady state $\frac{\partial f}{\partial t} = 0$. Then, we have

$$(\mu_2-\mu_1)\frac{df(x,1,1)}{dx} - (p_1+g_1+p_2+g_2)f(x,1,1) + r_2 f(x,1,0) + r_1 f(x,0,1) = 0 \quad (23)$$

In the same way, the eight other internal transition equations for the probability density function are

$$p_2 f(x,1,1) - \mu_1 \frac{df(x,1,0)}{dx} - (p_1 + g_1 + r_2) f(x,1,0) + f_2 f(x,1,-1) + r_1 f(x,0,0) = 0 \quad (24)$$

$$g_2 f(x,1,1) + (\mu_2 - \mu_1) \frac{df(x,1,-1)}{dx} - (p_1 + g + f_2) f(x,1,-1) + r_1 f(x,0,-1) = 0 \quad (25)$$

$$p_1 f(x,1,1) + \mu_2 \frac{df(x,0,1)}{dx} - (r_1 + p_2 + g_2) f(x,0,1) + r_2 f(x,0,0) + f_1 f(x,-1,1) = 0 \quad (26)$$

$$p_1 f(x,1,0) + p_2 f(x,0,1) - (r_1 + r_2) f(x,0,0) + f_2 f(x,0,-1) + f_1 f(x,-1,0) = 0 \quad (27)$$

$$p_1 f(x,1,-1) + g_2 f(x,0,1) - (r_1 + f_2) f(x,0,-1) + \mu_2 \frac{df(x,0,-1)}{dx} + f_1 f(x,-1,-1) = 0 \quad (28)$$

$$g_1 f(x,1,1) - (p_2 + g_2 + f_1) f(x,-1,1) + (\mu_2 - \mu_1) \frac{df(x,-1,1)}{dx} + r_2 f(x,-1,0) = 0 \quad (29)$$

$$g_1 f(x,1,0) - \mu_1 \frac{df(x,-1,0)}{dx} - (r_2 + f_1) f(x,-1,0) + p_2 f(x,-1,1) + f_2 f(x,-1,-1) = 0 \quad (30)$$

$$g_1 f(x,1,-1) + g_2 f(x,-1,1) + (\mu_2 - \mu_1) \frac{df(x,-1,-1)}{dx} - (f_1 + f_2) f(x,-1,-1) = 0 \quad (31)$$

3.2.2 Boundary transition equations. While the internal behavior of the system can be described by probability density functions, there is a nonzero probability of finding the system in certain boundary states. For example, if $\mu_1 < \mu_2$ and both machines are in state 1, the level of storage tends to decrease. If both machines remain operational for enough time, the storage will become empty ($x = 0$). Once the system reaches state $(0,1,1)$, it will remain there until a machine fails. There are 18 probability masses for boundary states ($P(N, \alpha_1, \alpha_2)$ and $P(0, \alpha_1, \alpha_2)$ where $\alpha_1 = -1$, 0 or 1, and $\alpha_2 = -1$, 0 or 1) and 22 boundary equations for the $\mu_1 = \mu_2$ case.

To arrive at state $(0,1,1)$ at time $t + \delta t$ when $\mu_1 = \mu_2$, the system may have been in one of two states at time t. It could have been in state $(0,1,1)$ without any of operational failures and quality failures for both of machines. It could have been in state $(0,0,1)$ with a repair of the first machine. (The second machine could not have failed since it was starved). If the second order terms are ignored,

$$P(0,1,1,t+\delta t) = \{1 - (p_1 + g_1 + p_2^b + g_2^b)\delta t\} P(0,1,1) + r_1 P(0,0,1) \quad (32)$$

After the usual analysis, (32) becomes

$$\frac{\partial P(0,1,1)}{\partial t} = (p_1 + g_1 + p_2^b + g_2^b)P(0,1,1) + r_1 P(0,0,1) \tag{33}$$

In steady state

$$-(p_1 + g_1 + p_2^b + g_2^b)P(0,1,1) + r_1 P(0,0,1) = 0 \tag{34}$$

There are 21 other boundary equations derived similarly for $\mu_1 = \mu_2$ [14]:

$$P(0,1,0) = 0 \tag{35}$$
$$g_2^b P(0,1,1) - (p_1 + g_1 + f_2^b)P(0,1,-1) + r_1 P(0,0,-1) = 0 \tag{36}$$
$$p_1 P(0,1,1) - r_1 P(0,0,1) + \mu_2 f(0,0,1) + f_1 P(0,-1,1) + r_2 P(0,0,0) = 0 \tag{37}$$
$$-(r_1 + r_2)P(0,0,0) = 0 \tag{38}$$
$$p_1 P(0,1,-1) - r_1 P(0,0,-1) + \mu_2 f(0,0,-1) + f_1 P(0,-1,-1) = 0 \tag{39}$$
$$g_1 P(0,1,1) - (f_1 + p_2^b + g_2^b)P(0,-1,1) = 0 \tag{40}$$
$$P(0,-1,0) = 0 \tag{41}$$
$$g_1 P(0,1,-1) + g_2^b P(0,-1,1) - (f_1 + f_2^b)P(0,-1,-1) = 0 \tag{42}$$
$$-(p_1^b + g_1^b + p_2 + g_2)P(N,1,1) + r_2 P(N,1,0) = 0 \tag{43}$$
$$p_2 P(N,1,1) - r_2 P(N,1,0) + \mu_1 f(N,1,0) + f_2 P(N,1,-1) + r_1 P(N,0,0) = 0 \tag{44}$$
$$g_2 P(N,1,1) - (p_1^b + g_1^b + f_2)P(N,1,-1) = 0 \tag{45}$$
$$P(N,0,1) = 0 \tag{46}$$
$$-(r_1 + r_2)P(N,0,0) = 0 \tag{47}$$
$$P(N,0,-1) = 0 \tag{48}$$
$$g_1^b P(N,1,1) - (f_1^b + g_2 + p_2)P(N,-1,1) + r_2 P(N,-1,0) = 0 \tag{49}$$
$$-r_2 P(N,-1,0) + \mu_1 f(N,-1,0) + f_2 P(N,-1,-1) + p_2 P(N,-1,1) = 0 \tag{50}$$
$$g_1^b P(N,1,-1) + g_2 P(N,-1,1) - (f_1^b + f_2)P(N,-1,-1) = 0 \tag{51}$$
$$\mu_1 f(0,1,0) = r_1 P(0,0,0) + p_2^b P(0,1,1) + f_2^b P(0,1,-1) \tag{52}$$
$$\mu_1 f(0,-1,0) = p_2^b P(0,-1,1) + f_2^b P(0,-1,-1) \tag{53}$$
$$\mu_2 f(N,0,1) = r_2 P(N,0,0) + p_1^b P(N,1,1) + f_1^b P(N,-1,1) \tag{54}$$
$$\mu_2 f(N,0,-1) = p_1^b P(N,1,-1) + g_2 P(N,0,1) + f_1^b P(N,-1,-1) \tag{55}$$

3.2.3 Normalization. In addition to these, all the probability density functions and probability masses must satisfy the normalization equation:

$$\sum_{\alpha_1=-1,0,1} \sum_{\alpha_2=-1,0,1} \left[\int_0^N f(x,\alpha_1,\alpha_2)dx + P(0,\alpha_1,\alpha_2) + P(N,\alpha_1,\alpha_2) \right] = 1 \tag{56}$$

3.2.4 Performance measures. After finding all probability density functions and probability masses, we can calculate the average inventory in the buffer from

$$\overline{x} = \sum_{\alpha_1=-1,0,1} \sum_{\alpha_2=-1,0,1} \left[\int_0^N x f(x, \alpha_1, \alpha_2) dx + N P(N, \alpha_1, \alpha_2) \right] \tag{57}$$

The total production rate is

$$P_T = P_T^1 = \sum_{\alpha_2=-1,0,1} \mu_1 \left[\int_0^N \{f(x,-1,\alpha_2)+f(x,1,\alpha_2)\} dx + P(0,1,\alpha_2) + P(0,-1,\alpha_2) \right] + \mu_2 \{P(N,1,-1) + P(N,1,1) + P(N,-1,-1) + P(N,-1,1\} \tag{58}$$

The rate at which machine M_1 produces good parts is

$$P_E^1 = \sum_{\alpha_2=-1,0,1} \mu_1 [\int_0^N f(x,1,\alpha_2) dx + P(0,1,\alpha_2)] + \mu_2 \{P(N,1,-1) + P(N,1,1)\} \tag{59}$$

The probability that the first machine produces a non-defective part is then $Y_1 = P_E^1/P_T$. Similarly, the probability that the second machine finishes its operation without adding a bad feature to a part is $Y_2 = P_E^2/P_T$, where

$$P_E^2 = \sum_{\alpha_1=-1,0,1} \mu_2 [\int_0^N f(x,\alpha_1,1) dx + P(N,\alpha_1,1)] + \mu_1 \{P(0,-1,1) + P(0,1,1)\} \tag{60}$$

Therefore, the effective production rate is

$$P_E = Y_1 Y_2 P_T \tag{61}$$

3.3 Validation

The 2M1B systems with the same machine speed ($\mu_1 = \mu_2$) are solved in Appendix A. As we have indicated, we represent discrete parts in this model as a continuous fluid and time as a continuous variable. We compare analytical and simulation results in this section. In the simulation, both material and time are discrete. Details are presented in [14].

Figure 5 shows the comparison of the effective production rate and the average inventory from the analytic model and the simulation. 50 cases are generated by changing machines and buffer parameters and % errors are plotted in the vertical

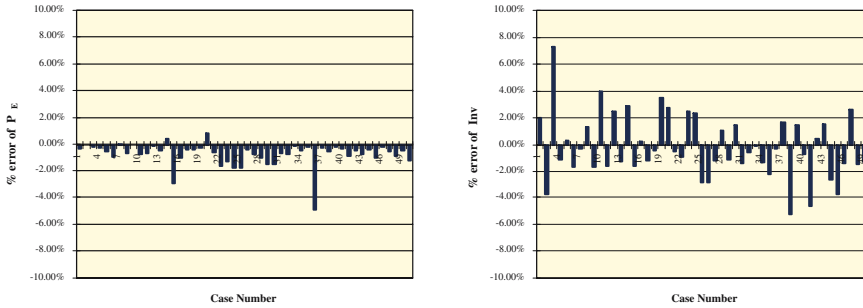


Fig. 5. Validation of the intermediate buffer size case

axis. The parameters for theses cases are given in Appendix B. The % error in the effective production rate is calculated from

$$P_E\%\text{error} = \frac{P_E(A) - P_E(S)}{P_E(S)} \times 100(\%) \tag{62}$$

where $P_E(A)$ and $P_E(S)$ are the effective production rates estimated from the analytical model and the simulation respectively. But the % error in the average inventory is calculated from

$$Inv_E\%\text{error} = \frac{Inv_E(A) - Inv_E(S)}{0.5 \times N} \times 100(\%) \tag{63}$$

where $Inv_E(A)$ and $Inv_E(S)$ are average inventory estimated from the analytical model and the simulation respectively and N is a buffer size[1].

The average absolute value of the % error in the effective production rate estimation is 0.76% and it is 1.89% for average inventory estimation.

4 Quality information feedback

Factory designers and managers know that it is ideal to have inspection after every operation. However, it is often costly to do this. As a result, factories are usually designed so that multiple inspections are performed at a small number of stations. In this case, inspection at downstream operations can detect bad features made by upstream machines. We call this *quality information feedback*. A simple example of the quality information feedback in 2M1B systems is when M_1 produces defective features but does not have inspection and M_2 has inspection and it can detect bad features made by M_1. In this situation, as we demonstrate below, the yield of a line is a function of the size of buffer. This is because when buffer gets larger, more material can accumulate between an operation (M_1) and the inspection of that operation (M_2). All such material will be defective if a persistent quality failure

[1] This is an unbiased way to calculate the error in average inventory. If it were calculated in the same way as the error in the effective production rate, the error would depend on the relative speeds of the machines. This is because there will be a lower error when the buffer is mostly full (ie, when M_1 is faster than M_2) and a higher error when the buffer is empty (when M_1 is faster than M_2).

takes place. In other words, if buffer is larger, there tends to be more material in the buffer and consequently more material is defective. In addition it takes longer to have inspections after finishing operations. We can capture this phenomenon with the adjustment of a transition probability rate of M_1 from state -1 to state 0.

Let us define f_1^q as a transition rate of M_1 from state -1 to state 0 when there is a quality information feedback and f_1 as the transition rate without the quality information feedback. The adjustment can be done in a way that the yield of M_1 is the same as $\frac{Z_1^g}{Z_1^g+Z_1^b}$ where

* Z_1^b: the expected number of bad parts generated by M_1 while it stays in state -1.
* Z_1^g: the expected number of good parts produced by M_1 from the moment when M_1 leaves the -1 state to the next time it arrives at state -1.

From (10), the yield of M_1 is

$$\frac{P(1)}{P(1)+P(-1)} = \frac{f_1^q}{f_1^q+g_1} \tag{64}$$

Suppose that M_1 has been in state 1 for a very long time. Then all parts in the buffer B are non-defective. Suppose that M_1 goes to state -1. Defective parts will then begin to accumulate in the buffer. Until all the parts in the buffer are defective, the only way that M_1 can go to state 0 is due to its own inspection or its own operation failure. Therefore, the probability of a transition to 0 before M_1 finishes a part is

$$\frac{f_1}{\mu_1} \equiv \chi_{11}$$

Eventually all the parts in the buffer are bad so that defective parts reach M_2. Then, there is another way that M_1 can move to state 0 from state -1: quality information feedback. The probability that the inspection at M_2 detects a nonconformity made by M_1 is

$$\chi_{21} \equiv \frac{h_{21}}{\mu_2}$$

where $1/h_{21}$ is the mean time until the inspection at M_2 detects a bad part made by M_1 after M_2 receives the bad part.

The expected value of the number of bad parts produced by M_1 before it is stopped by either operational failures or quality information feedback is

$$\begin{aligned} Z_1^b = &[\chi_{11} + 2\chi_{11}(1-\chi_{11}) + 3\chi_{11}(1-\chi_{11})^2 + \ldots + w\chi_{11}(1-\chi_{11})^{w-1}] \\ &+[(w+1)(1-\chi_{11})^w\chi_{21} + (w+2)(1-\chi_{11})^{w+1}\chi_{21}(1-\chi_{21}) + \ldots] \end{aligned} \tag{65}$$

where w is average inventory in the buffer B. This is an approximate formula since we simply use the average inventory rather than averaging the expected number of bad parts produced by M_1 depending on different inventory levels w_i. After some mathematical manipulation,

$$\begin{aligned} Z_1^b = &\frac{1-(1-\chi_{11})^w}{\chi_{11}} - w(1-\chi_{11})^w \\ &+\frac{(1-\chi_{11})^w\chi_{21}[(w+1)-w(1-\chi_{11})(1-\chi_{21})]}{[1-(1-\chi_{11})(1-\chi_{21})]^2} \end{aligned} \tag{66}$$

On the other hand, Z_1^g is given as

$$Z_1^g = \frac{\mu_1}{p_1+g_1} + \frac{p_1}{p_1+g_1}\frac{\mu_1}{p_1+g_1} + \left(\frac{p_1}{p_1+g_1}\right)^2\left(\frac{\mu_1}{p_1+g_1}\right)\ldots = \frac{\mu_1}{g_1} \tag{67}$$

By setting $\frac{f_1^q}{f_1^q+g_1} = \frac{Z_1^g}{Z_1^g+Z_1^b}$ we have

$$f_1^q = \frac{\mu_1}{\frac{1-(1+w\chi_{11})(1-\chi_{11})^w}{\chi_{11}} + \frac{(1-\chi_{11})^w\chi_{21}[1+w(\chi_{21}+\chi_{11}-\chi_{21}\chi_{11})]}{[1-(1-\chi_{11})(1-\chi_{21})]^2}} \tag{68}$$

Since the average inventory is a function of f_1^q and f_1^q is dependent on the average inventory, an iterative method is required to determine these values.

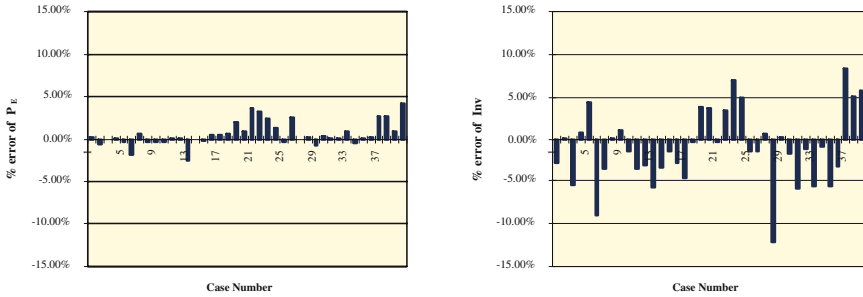


Fig. 6. Validation of the quality information feedback formula

Figure 6 shows the comparison of the effective production rate and the average inventory from the analytic model and the simulation. 50 cases are generated by selecting different machine and buffer parameters and % errors are plotted in the y-axis. The parameters for theses cases are given in Appendix B. % errors in the effective production rate and average inventory are calculated using equations (62) and (63) respectively. The average absolute value of the % error in P_E and $\overline{x}$ estimations are 1.01% and 3.67% respectively.

5 Insights from numerical experimentation

In this section, we perform a set of numerical experiments to provide intuitive insight into the behavior of production lines with inspection. The parameters of all the cases are presented in Appendix B.

5.1 Beneficial buffer case

5.1.1 Production rates. Having quality information feedback means having more inspection than otherwise. Therefore, machines tend to stop more frequently. As a result, the total production rate of the line decreases. However, the effective production rate can increase since added inspections prevent the making of defective parts. This phenomenon is shown in Figure 7. Note that the total production rate P_T without quality information feedback is consistently higher than P_T with quality information feedback regardless of buffer size and the opposite is true for the

effective production rate P_E. Also it should be noted that in this case, both the total production rate and the effective production rate increase with buffer size, with or without quality information feedback.

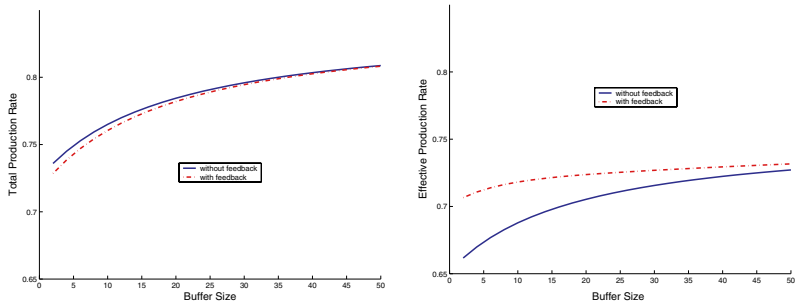


Fig. 7. Production rates with/without quality information feedback

5.1.2 System yield and buffer size. Even though a larger buffer increases both total and effective production rates in this case, it decreases yield. As explained in Section 4, the system yield is a function of the buffer size if there is quality information feedback. Figure 8 shows system yield decreasing as buffer size increases when there is quality information feedback. This happens because when the buffer gets larger, more material accumulates between an operation and the inspection of that operation. All such material will be defective when the first machine is at state -1 but the inspection at the first machine does not find it. This is a case in which *a smaller buffer improves quality,* which is widely believed to be generally true. If there is no quality information feedback, then the system yield is independent of the buffer size (and is substantially less).

5.2 Harmful buffer case

5.2.1 Production rates. Typically, increasing the buffer size leads to higher effective production rate. This is the case in Figure 7. But under certain conditions, the effective production rate can actually decrease as buffer size increases. This can happen when

* The first machine produces bad parts frequently: this means g_1 is large.
* The inspection at the first machine is poor or non-existent and inspection at the second machine is reliable: this means $h_1 \ll h_2$ or $f_1 - p_1 \ll f_2 - p_2$.
* There is quality information feedback.
* The isolated production rate of the first machine is higher than that of the second machine:

$$\mu_1 \frac{1 + g_1/f_1}{1 + (p_1 + g_1)/r_1 + g_1/f_1} > \mu_2 \frac{1 + g_2/f_2}{1 + (p_2 + g_2)/r_2 + g_2/f_2}$$

Figure 9 shows a case in which a buffer size increase leads to a lower effective production rate. Note that even in this case, total production rate monotonically increases as buffer size increases.

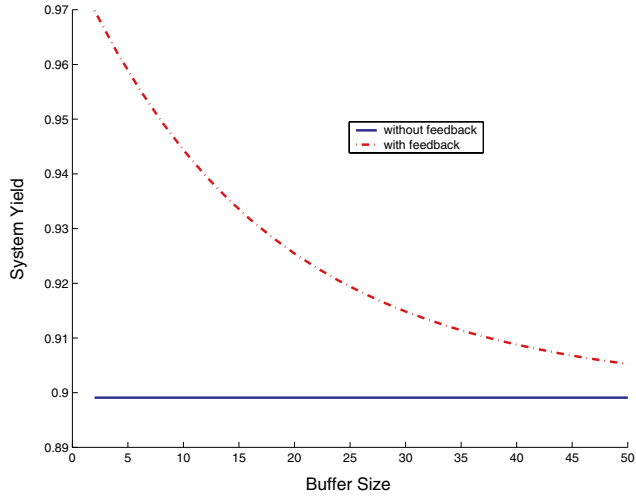


Fig. 8. System yield as a function of buffer size

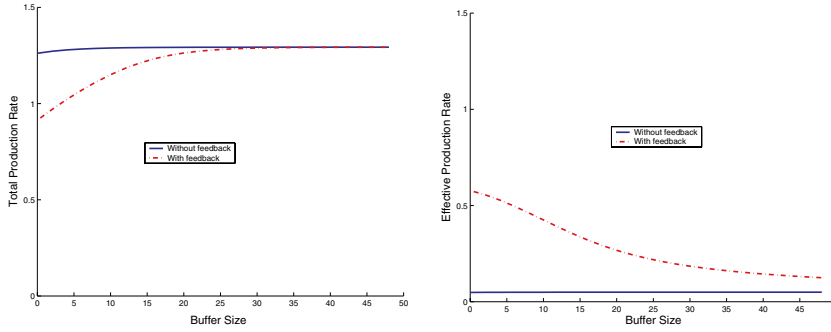


Fig. 9. Total production rate and effective production rate

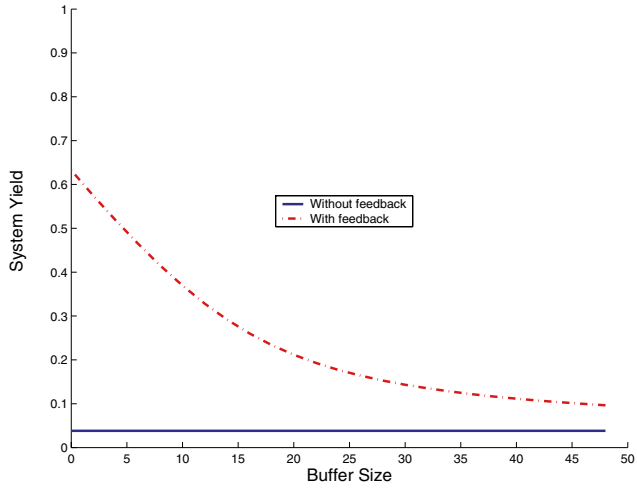


Fig. 10. System yield as a function of buffer size

5.2.2 System yield. The system yield for this case is shown in Figure 10. Note that the yield decreases dramatically as the buffer size increases. In this case, the decrease of the system yield is more than the increase of the total production rate so that the effective production rate monotonically decreases as buffer size gets bigger.

5.3 How to improve quality in a line with persistent quality failures

There are two major ways to improve quality. One is to increase the yield of individual operations and the other is to perform more rigorous inspection. Having extensive preventive maintenance on manufacturing equipment and using robust engineering techniques to stabilize operations have been suggested as tools to increase yield of individual operations. Both approaches increase the Mean Time to Quality Failure (MTQF) (i.e. decrease g). On the other hand, the inspection policy aims to detect bad parts as soon as possible and prevent their flow toward downstream operations. More rigorous inspection decreases the mean time to detect (MTTD) (i.e. increases h and therefore increases f). It is natural to believe that using only one kind of method to achieve a target quality level would not give the most cost efficient quality assurance policy. Figure 11 indicates that the impact of individual operation stabilization on the system yield decreases as the operation becomes more stable. It also shows that effect of improving inspection (MTTD) on the system yield decreases. Therefore, it is optimal to use a combination of both methods to improve quality.

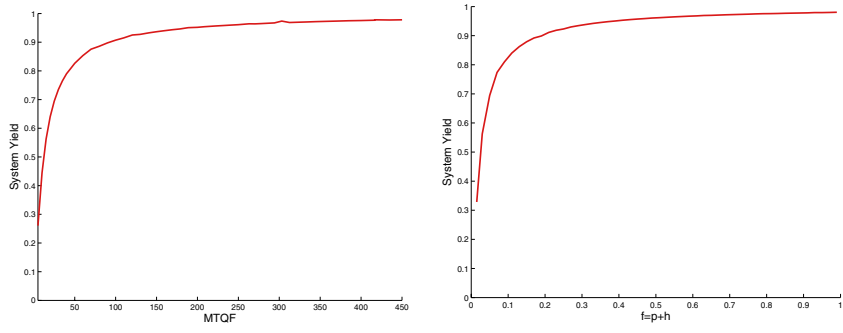


Fig. 11. Quality improvement

5.4 How to increase productivity

Improving the stand-alone throughput of each operation and increasing the buffer space are typical ways to increase the production rate of manufacturing systems. If operations are apt to have quality failures, however, there may be other ways to increase the effective production rate: increasing the yield of each operation and conducting more extensive inspections. Stabilizing operations, thus improving the yield of individual operations, will increase effective throughput of a manufacturing

system regardless of the type of quality failure. On the other hand, reducing the mean time to detect (MTTD) will *increase* the effective production rate only if the quality failure is persistent but it will *decrease* the effective production rate if the quality failure is Bernoulli. This is because the quality of each part is independent of the others when the quality failure is Bernoulli. Therefore, stopping the line does not reduce the number of bad parts in the future.

In a situation in which machines produce defective parts frequently and inspection is poor, increasing inspection reliability is more effective than increasing buffer size to boost the effective production rate. Figure 12 shows this. Also, in other situations in which machines produce defective parts frequently and inspection is reliable, increasing machine stability is more effective than increasing buffer size to enhance effective production rate. Figure 13 shows this phenomenon.

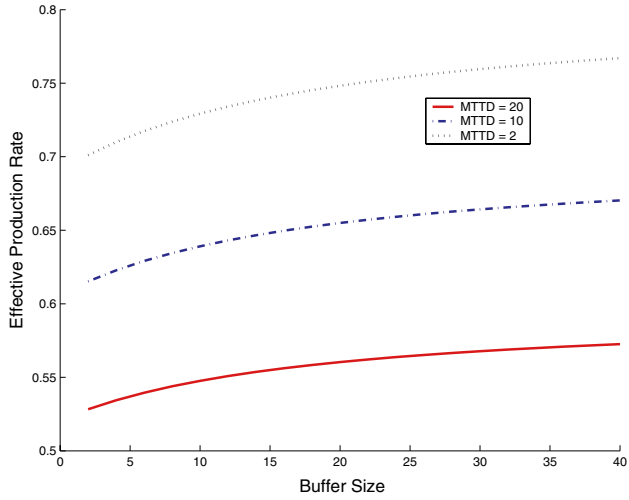


Fig. 12. Mean time to detect and effective production rate

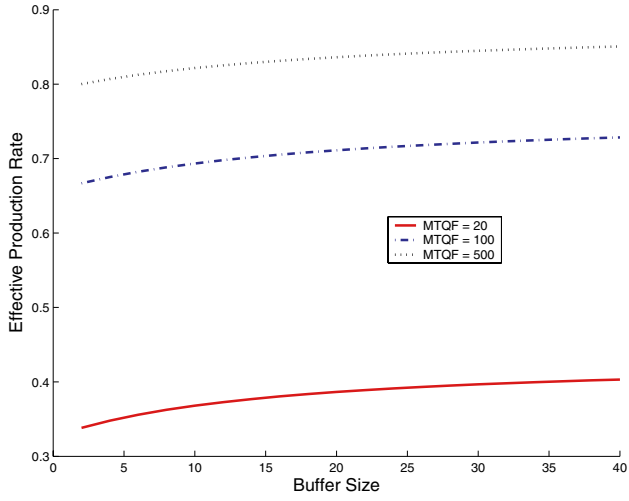


Fig. 13. Quality failure frequency and effective production rate

6 Future research

The 2-Machine-1-Buffer (2M1B) model with $\mu_1 \neq \mu_2$ is analyzed in [14]. This case is more challenging because the number of roots of the internal transition equations depends on parameters of machine. A more general 2M1B model with multiple-yield quality failures (a mixture of Bernoulli- and persistent-type quality failures) should also be studied. A long line analysis using decomposition is under the development. Refer to Kim [14] for more detailed information.

Appendix

A Solution technique

It is natural to assume an exponential form for the solution to the steady state density functions since equations (23)–(31) are coupled ordinary linear differential equations. A solution of the form $e^{\lambda x}K_1^{\alpha_1}K_2^{\alpha_2}$ worked successfully in the continuous material two-machine line with perfect quality [10]. Therefore, a solution of a form

$$f(x, \alpha_1, \alpha_2) = e^{\lambda x} G_1(\alpha_1) G_2(\alpha_2) \tag{69}$$

is assumed here. This form satisfies the transition equations if all of the following equations are met. Equations (23)–(31) become, after substituting (69),

$$\{(\mu_2 - \mu_1)\lambda - (p_1 + g_1 + p_2 + g_2)G_1(1)G_2(1)\}$$
$$+r_2 G_1(1)G_2(0) + r_1 G_1(0)G_2(1) = 0 \tag{70}$$
$$-\{\mu_1\lambda + (p_1 + g_1 + r_2)\}G_1(1)G_2(0) + p_2 G_1(1)G_2(1) + f_2 G_1(1)G_2(-1)$$
$$+r_1 G_1(0)G_2(0) = 0 \tag{71}$$
$$\{(\mu_2 - \mu_1)\lambda - (p_1 + g_1 + f_2)\}G_1(1)G_2(-1) + g_2 G_1(1)G_2(1)$$
$$+r_1 G_1(0)G_2(-1) = 0 \tag{72}$$
$$\{\mu_2\lambda - (r_1 + p_2 + g_2)\}G_1(0)G_2(1) + p_1 G_1(1)G_2(1) + r_2 G_1(0)G_2(0)$$
$$+f_1 G_1(-1)G_2(1) = 0 \tag{73}$$
$$p_1 G_1(1)G_2(0) + p_2 G_1(0)G_2(1) - (r_1 + r_2)G_1(0)G_2(0) + f_2 G_1(0)G_2(-1)$$
$$+f_1 G_1(-1)G_2(0) = 0 \tag{74}$$
$$\{\mu_2\lambda - (r_1 + f_2)\}G_1(0)G_2(-1) + p_1 G_1(1)G_2(-1) + g_2 G_1(0)G_2(1)$$
$$+f_1 G_1(-1)G_2(-1) = 0 \tag{75}$$
$$\{(\mu_2 - \mu_1)\lambda - (p_2 + g_2 + f_1)\}G_1(-1)G_2(1) + g_1 G_1(1)G_2(1)$$
$$+r_2 G_1(-1)G_2(0) = 0 \tag{76}$$
$$-\{\mu_1\lambda + (r_2 + f_1)\}G_1(-1)G_2(0) + g_1 G_1(1)G_2(0) + p_2 G_1(-1)G_2(1)$$
$$+f_2 G_1(-1)G_2(-1) = 0 \tag{77}$$
$$\{(\mu_2 - \mu_1)\lambda - (f_1 + f_2)\}G_1(-1)G_2(-1) + g_1 G_1(1)G_2(-1)$$
$$+g_2 G_1(-1)G_2(1) = 0 \tag{78}$$

These are nine equations in seven unknowns ($\lambda, G_1(1), G_2(0), G_1(-1), G_2(1), G_2(0)$, and $G_2(-1)$). Thus, there must be seven independent equations and two dependent ones.

If we divide equations (70) – (78) by $G_1(0)G_2(0)$ and define new parameters

$$\Gamma_i = p_i \frac{G_i(1)}{G_i(0)} - r_i + f_i \frac{G_i(-1)}{G_i(0)} \tag{79}$$

$$\Psi_i = -p_i - g_i + r_i \frac{G_i(0)}{G_i(1)} \tag{80}$$

$$\Theta_i = -f_i + g_i \frac{G_i(1)}{G_i(-1)} \tag{81}$$

then equations (70)–(78) can be rewritten as

$$\Gamma_1 + \Gamma_2 = 0 \tag{82}$$

$$-\mu_2\lambda = \Gamma_1 + \Psi_2 \tag{83}$$

$$\mu_1\lambda = \Gamma_2 + \Psi_1 \tag{84}$$

$$(\mu_1 - \mu_2)\lambda = \Psi_1 + \Psi_2 \tag{85}$$

$$(\mu_1 - \mu_2)\lambda = \Theta_1 + \Theta_2 \tag{86}$$

$$\mu_1\lambda = \Gamma_2 + \Theta_1 \tag{87}$$

$$-\mu_2\lambda = \Gamma_1 + \Theta_2 \tag{88}$$

$$(\mu_1 - \mu_2)\lambda = \Psi_2 + \Theta_1 \tag{89}$$

$$(\mu_1 - \mu_2)\lambda = \Psi_1 + \Theta_2 \tag{90}$$

From equations (82)–(90), it is clear that only seven equations are independent. After much mathematical manipulation [14], these equations become

$$0 = \frac{\{(M + r_1)(\mu_1 N - 1) - f_1\}^2}{(f_1 - p_1)(\mu_1 N - 1)} - \frac{\{(p_1 + g_1 - f_1) + r_1(\mu_1 N - 1)\}\{(M + r_1)(\mu_1 N - 1) - f_1\}}{(f_1 - p_1)(\mu_1 N - 1)} - r_1 \tag{91}$$

$$0 = \frac{\{(-M + r_2)(\mu_2 N - 1) - f_2\}^2}{(f_2 - p_2)(\mu_2 N - 1)} - \frac{\{(p_2+g_2-f_2)+r_2(\mu_2 N-1)\}\{(-M+r_2)(\mu_2 N-1)-f_2\}}{(f_2-p_2)(\mu_2 N-1)} - r_2 = 0 \tag{92}$$

where

$$p_1 \frac{G_1(1)}{G_1(0)} - r_1 + f_1 \frac{G_1(-1)}{G_1(0)} = -\left(p_2 \frac{G_2(1)}{G_2(0)} - r_2 + f_2 \frac{G_2(-1)}{G_2(0)}\right) = M \tag{93}$$

$$\frac{1}{\mu_1}\left(1 + \frac{1}{G_1(1)/G_1(0) + G_1(-1)/G_1(0)}\right) = \frac{1}{\mu_2}\left(1 + \frac{1}{G_2(1)/G_2(0) + G_2(-1)/G_2(0)}\right) = N \tag{94}$$

Now all the equations and unknowns are simplified into two unknowns and two equations. By solving equations (91) and (92) simultaneously we can calculate

M and N. An example of these equations is plotted in Figure 14. Equation (91) is represented with lighter lines and equation (92) is shown as darker lines. The intersections of the two sets of lines are the solutions of the equations.

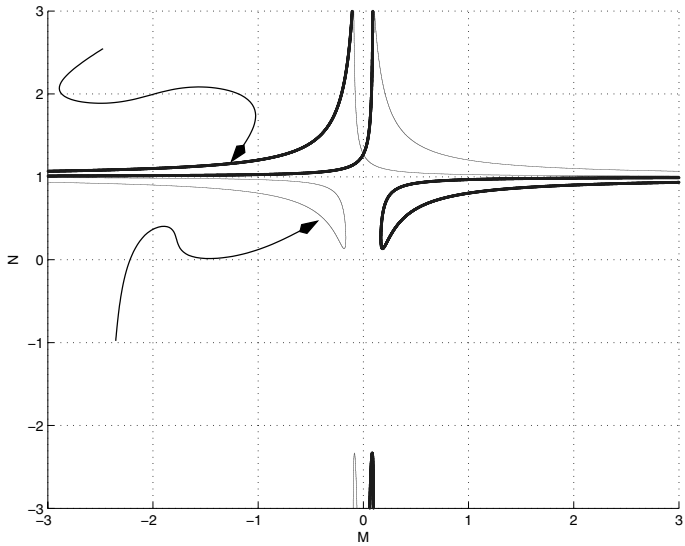


Fig. 14. Plot of equations (91) and (92)

These are high order polynomial equations for which no general analytical solution exists. A numerical approach is required to find the roots of the equations. A special algorithm to find the solutions has been developed [14] based on the characteristics of the equations. Once we find roots of equations (91) and (92), we can get ratios $\frac{G_i(1)}{G_i(0)}$ and $\frac{G_i(-1)}{G_i(0)}$ $(i = 1, 2)$ from equation (94). By setting $G_1(0) = G_2(0) = 1$, we can calculate $G_1(1), G_1(-1), G_2(1)$, and $G_2(-1)$. After some mathematical manipulation, we find that λ can be expressed as

$$\lambda = \frac{-p_1 - g_1 + {}^{r_1}\!/_{G_1(1)} - p_1 G_1(1) + r_1 - f_1 G_1(-1)}{\mu_1} \tag{95}$$

Therefore, we can get a probability density function $f(x, \alpha_1, \alpha_2)$ corresponding to an (M, N) pair. The number of roots in equations (91) and (92) depends on machine parameters. There are only 3 roots when $\mu_1 = \mu_2$ regardless of other parameters. Therefore, a general expression of the probability density function in this case is

$$f(x, \alpha_1, \alpha_2) = c_1 f_1(x, \alpha_1, \alpha_2) + c_2 f_2(x, \alpha_1, \alpha_2) + c_3 f_3(x, \alpha_1, \alpha_2) \tag{96}$$

where $f_1(x, \alpha_1, \alpha_2), f_2(x, \alpha_1, \alpha_2), f_3(x, \alpha_1, \alpha_2)$ are the roots of the equations (91) and (92).

The remaining unknowns, including c_1, c_2, c_3 and probability masses at the boundaries, can be calculated by solving boundary equations ((34)–(55)) and the normalization equation (56) with $f_i(x, \alpha_1, \alpha_2)$ given by equation (96).

B Machine parameters for numerical and simulation experiments

Table 4. Machine parameters for infinite buffer case and zero buffer case

Case #	μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2
1	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2
2	1.0	1.0	0.3	0.3	0.005	0.005	0.05	0.05	0.5	0.5
3	1.0	1.0	0.2	0.05	0.01	0.01	0.01	0.01	0.2	0.2
4	1.0	1.0	0.1	0.1	0.05	0.005	0.01	0.01	0.2	0.2
5	1.0	1.0	0.1	0.1	0.01	0.01	0.05	0.005	0.2	0.2
6	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.5	0.1
7	2.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.5	0.1
8	3.0	2.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2
9	1.0	2.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2
10	2.0	3.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2

Table 5. Machine parameters for Figures 7 and 8

μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2
1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.1	0.9

Table 6. Machine parameters for Figures 9 and 10

μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2
2.0	2.0	0.5	0.1	0.005	0.05	0.5	0.005	0.02	0.9

Table 7. Machine parameters for Figure 11

μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2
1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2

Table 8. Machine parameters for Figure 12

μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2
1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01

Table 9. Machine parameters for Figure 13

μ_1	μ_2	r_1	r_2	p_1	p_2	f_1	f_2
1.0	1.0	0.1	0.1	0.01	0.01	0.2	0.2

Table 10. Machine parameters for intermediate buffer case validation

Case #	μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2	N
1	1.0	1.0	0.1	0.1	0.01	0.01	0.02	0.01	0.1	0.2	30
2	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	5
3	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	10
4	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	15
5	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	20
6	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	25
7	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	35
8	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	40
9	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	45
10	0.5	0.5	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
11	1.5	1.5	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
12	2.0	2.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
13	2.5	2.5	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
14	3.0	3.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
15	1.0	1.0	0.01	0.01	0.01	0.01	0.01	0.01	0.2	0.2	30
16	1.0	1.0	0.05	0.05	0.01	0.01	0.01	0.01	0.2	0.2	30
17	1.0	1.0	0.2	0.2	0.01	0.01	0.01	0.01	0.2	0.2	30
18	1.0	1.0	0.5	0.5	0.01	0.01	0.01	0.01	0.2	0.2	30
19	1.0	1.0	0.8	0.8	0.01	0.01	0.01	0.01	0.2	0.2	30
20	1.0	1.0	0.1	0.1	0.001	0.001	0.01	0.01	0.2	0.2	30
21	1.0	1.0	0.1	0.1	0.005	0.005	0.01	0.01	0.2	0.2	30
22	1.0	1.0	0.1	0.1	0.02	0.02	0.01	0.01	0.2	0.2	30
23	1.0	1.0	0.1	0.1	0.05	0.05	0.01	0.01	0.2	0.2	30
24	1.0	1.0	0.1	0.1	0.1	0.1	0.01	0.01	0.2	0.2	30
25	1.0	1.0	0.1	0.1	0.01	0.01	0.001	0.001	0.2	0.2	30
26	1.0	1.0	0.1	0.1	0.01	0.01	0.005	0.005	0.2	0.2	30
27	1.0	1.0	0.1	0.1	0.01	0.01	0.02	0.02	0.2	0.2	30
28	1.0	1.0	0.1	0.1	0.01	0.01	0.05	0.05	0.2	0.2	30
29	1.0	1.0	0.1	0.1	0.01	0.01	0.10	0.10	0.2	0.2	30
30	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.02	0.02	30
31	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.05	0.05	30
32	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.1	0.1	30
33	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.5	0.5	30
34	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.95	0.95	30
35	1.0	1.0	0.5	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
36	1.0	1.0	0.01	0.1	0.01	0.01	0.01	0.01	0.2	0.2	30
37	1.0	1.0	0.1	0.5	0.01	0.01	0.01	0.01	0.2	0.2	30
38	1.0	1.0	0.1	0.01	0.01	0.01	0.01	0.01	0.2	0.2	30
39	1.0	1.0	0.1	0.1	0.1	0.01	0.01	0.01	0.2	0.2	30
40	1.0	1.0	0.1	0.1	0.001	0.01	0.01	0.01	0.2	0.2	30
41	1.0	1.0	0.1	0.1	0.01	0.1	0.01	0.01	0.2	0.2	30
42	1.0	1.0	0.1	0.1	0.01	0.001	0.01	0.01	0.2	0.2	30
43	1.0	1.0	0.1	0.1	0.01	0.01	0.1	0.01	0.2	0.2	30
44	1.0	1.0	0.1	0.1	0.01	0.01	0.001	0.01	0.2	0.2	30
45	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.1	0.2	0.2	30
46	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.001	0.2	0.2	30
47	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.9	0.2	30
48	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.05	0.2	30
49	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.9	30
50	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	0.05	30

Table 11. Machine parameters for quality information feedback validation

Case #	μ_1	μ_2	r_1	r_2	p_1	p_2	g_1	g_2	f_1	f_2	N
1	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
2	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	0
3	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	5
4	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	20
5	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	30
6	1.0	1.0	0.01	0.01	0.01	0.01	0.01	0.01	0.01	1.0	10
7	1.0	1.0	0.05	0.05	0.01	0.01	0.01	0.01	0.01	1.0	10
8	1.0	1.0	0.4	0.4	0.01	0.01	0.01	0.01	0.01	1.0	10
9	1.0	1.0	0.8	0.8	0.01	0.01	0.01	0.01	0.01	1.0	10
10	1.0	1.0	0.1	0.1	0.001	0.001	0.01	0.001	0.01	1.0	10
11	1.0	1.0	0.1	0.1	0.005	0.005	0.01	0.005	0.01	1.0	10
12	1.0	1.0	0.1	0.1	0.02	0.02	0.01	0.01	0.02	1.0	10
13	1.0	1.0	0.1	0.1	0.1	0.1	0.01	0.01	0.1	1.0	10
14	1.0	1.0	0.1	0.1	0.01	0.01	0.001	0.001	0.01	1.0	10
15	1.0	1.0	0.1	0.1	0.01	0.01	0.005	0.005	0.01	1.0	10
16	1.0	1.0	0.1	0.1	0.01	0.01	0.02	0.02	0.01	1.0	10
17	1.0	1.0	0.1	0.1	0.01	0.01	0.05	0.05	0.01	1.0	10
18	0.5	0.5	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
19	1.5	1.5	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
20	2.0	2.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
21	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.05	1.0	10
22	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	1.0	10
23	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.5	1.0	10
24	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.8	1.0	10
25	1.0	1.0	0.5	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
26	1.0	1.0	0.01	0.1	0.01	0.01	0.01	0.01	0.01	1.0	10
27	1.0	1.0	0.1	0.5	0.01	0.01	0.01	0.01	0.01	1.0	10
28	1.0	1.0	0.1	0.01	0.01	0.01	0.01	0.01	0.01	1.0	10
29	1.0	1.0	0.1	0.1	0.1	0.01	0.01	0.01	0.1	1.0	10
30	1.0	1.0	0.1	0.1	0.001	0.01	0.01	0.01	0.001	1.0	10
31	1.0	1.0	0.1	0.1	0.01	0.1	0.01	0.01	0.01	1.0	10
32	1.0	1.0	0.1	0.1	0.01	0.001	0.01	0.01	0.01	1.0	10
33	1.0	1.0	0.1	0.1	0.01	0.01	0.05	0.01	0.01	1.0	10
34	1.0	1.0	0.1	0.1	0.01	0.01	0.001	0.01	0.01	1.0	10
35	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.05	0.01	1.0	10
36	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.001	0.01	1.0	10
37	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.5	1.0	10
38	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.2	1.0	10
39	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	0.8	10
40	1.0	1.0	0.1	0.1	0.01	0.01	0.01	0.01	0.01	0.2	10

References

1. Alles M, Amershi A, Datar S, Sarkar R (2000) Information and incentive effects of inventory in JIT production. Management Science 46(12): 1528–1544
2. Besterfield DH, Besterfield-Michna C, Besterfield G, Besterfield-Sacre M (2003) Total quality management. Prentice Hall, Englewood Cliffs
3. Black JT (1991) The design of the factory with a future. McGraw-Hill, New York
4. Bonvik AM, Couch CE, Gershwin SB (1997) A comparison of production line control mechanisms. International Journal of Production Research 35(3): 789–804

5. Burman M, Gershwin SB, Suyematsu C (1998) Hewlett-Packard uses operations research to improve the design of a printer production line. Interfaces 28(1): 24–26
6. Buzacott JA, Shantikumar JG (1993) Stochastic models of manufacturing systems. Prentice-Hall, Englewood Cliffs
7. Cheng CH, Miltenburg J, Motwani J (2000) The effect of straight and U shaped lines on quality. IEEE Transactions on Engineering Management 47(3): 321–334
8. Dallery Y, Gershwin SB (1992) Manufacturing flow line systems: a review of models and analytical results. Queuing Systems Theory and Applications 12: 3–94
9. Fujimoto T (1999) The evolution of a manufacturing systems at Toyota. Oxford University Press, Oxford
10. Gershwin SB (1994) Manufacturing systems engineering. Prentice Hall, Englewood Cliffs
11. Gershwin SB (2000) Design and operation of manufacturing systems – the control-point policy. IIE Transactions 32(2): 891–906
12. Gershwin SB, Schor JE (2000) Efficient algorithms for buffer space allocation. Annals of Operations Research 93: 117–144
13. Inman RR, Blumenfeld DE, Huang N, Li J (2003) Designing production systems for quality: research opportunities from an automotive industry perspective. International Journal of Production Research 41(9): 1953–1971
14. Kim J (2004) Integrated quality and quantity modeling of a production line. Massachusetts Institute of Technology PhD thesis (in preparation)
15. Law AM, Kelton DW, Kelton WD, Kelton DM (1999) Simulation modeling and analysis. McGraw-Hill, New York
16. Ledolter J, Burrill CW (1999) Statistical quality control. Wiley, New York
17. Monden Y (1998) Toyota production system – an integrated approach to just-in-time. EMP Books, Norcross
18. Montgomery DC (2001) Introduction to statistical quality control, 4th edn. Wiley, New York
19. Pande P, Holpp L (2002) What is six sigma? McGraw-Hill, New York
20. Phadke M (1989) Quality engineering using robust design. Prentice Hall, Englewood Cliffs
21. Raz T (1986) A survey of models for allocating inspection effort in multistage production systems. Journal of Quality Technology 18(4): 239–246
22. Shin WS, Mart SM, Lee HF (1995) Strategic allocation of inspection stations for a flow assembly line: a hybrid procedure. IIE Transactions 27: 707–715
23. Shingo S (1989) A study of the Toyota production system from an industrial engineering viewpoint. Productivity Press, Portland
24. Toyota Motor Corporation (1996) The Toyota production system
25. Wein L (1988) Scheduling semiconductor wafer fabrication. IEEE Transactions on semiconductor manufacturing 1(3): 115–130
26. Wooddall WH, Montgomery DC (1999) Research issues and ideas in statistical process control. Journal of Quality Technology 31(4): 376–386

Stochastic cyclic flow lines with blocking: Markovian models

Young-Doo Lee and Tae-Eog Lee

Department of Industrial Engineering, Korea Advanced Institute of Science and Technology, 373-1, Kuseong-Dong, Yuseong-Gu, Taejon 305-701, Korea
(e-mail: {ydlee,telee}@kaist.ac.kr)

Abstract. We consider a cyclic flow line model that repetitively produces multiple items in a cyclic order. We examine performance of stochastic cyclic flow line models with finite buffers of which processing times have exponential or phase-type distributions. We develop an exact method for computing a two-station model by making use of the matrix geometric structure of the associated Markov chain. We present a computationally tractable approximate performance computing method that decomposes the line model into a number of two-station submodels and parameterizing the submodels by propagating the starvation and blocking probabilities through the adjacent submodels. We discuss performance characteristics including comparison with random order processing and effects of the job variation and the job processing sequence. We also report the accuracy of our proposed method.

Keywords: Cyclic flow line – Stochastic – Blocking – Performance – Decomposition

1 Introduction

Cyclic production is a way of producing multiple items simultaneously in a shop. It repetitively produces an identical set of items in the same loading and sequence at each station. For instance, we have production requirement of 100, 200, and 300 items for item types a, b, and c, respectively. Then, the minimal set of 1 a, 2 b's, and 3 c's is produced 100 times in the same production method. Depending on the visit sequence of the items through the stations, the shop can be a job shop or a flow line. In a *cyclic flow line*, each item flows through the stations in the same sequence. Each station processes the items in the order of first come first service. Therefore, each station repeats an identical cyclic sequence of processing the items, for instance,

Correspondence to: T.-E. Lee, 373-1 Gusung-Dong, Yusong-Gu, Daejon 305-701, Korea

a, b, b, c, c, c, which is the same as the release sequence of the items into the line. Cyclic flow lines are widely used for assembly lines or serial processing lines where multiple types of items are simultaneously produced and the setup times are not significant. Advantages of cyclic production over conventional batch production or random order production include better utilization of the machines, simplified flow control, continuous and smooth supply of complete part sets for downstream assembly, timely delivery, and reduced work-in-progress inventory [14].

There have been studies on cyclic shops. Essential issues can be found in [1, 7, 10, 11, 14–18, 22]. Cyclic flow lines are often used for printed circuit board assembly and electronics or other home appliance assembly, and integrated with an accumulation-type conveyor system. Such a conveyor system allows only a few parts to wait before each station. Such cyclic flow lines with blocking have been examined [1, 18, 22]. They deal with scheduling issues for the cases where process times are completely known. However, cyclic shops are subject to random disruptions such as tool jamming and recovery, retrials of an assembly operation, etc. These tend to contribute to random variation in job processing times. Scheduling models often neglect transport times or tend to increase the processing times by the transport times. Such approximate modeling simplifies the scheduling model, but adds randomness of the transport times to the combined process times.

There are a few works on stochastic cyclic shop models. Rao and Jackson [20] develop an approximate algorithm to compute the average cycle time for a cyclic job shop with general processing time distributions, which makes use of Clark's approximation method for stochastic PERT networks. Bowman and Muckstadt [2] deliberately develop a finite Markov chain model for a cyclic job shop with exponential processing times and compute the average cycle time, but do not discuss the queue length. Zhang and Graves [23] find the schedules that are least disturbed by machine failures in a re-entrant cyclic flow shop. For cyclic flow lines, Seo and Lee [21] examine the queue length distributions of the cases that have exponential processing times and infinite buffers. Stochastic cyclic flow lines with limited buffers have distinct performance characteristics and require a different performance analysis method due to blocking. Therefore, it is necessary to examine performance of stochastic flow lines with blocking. Karabati and Tan [13] propose a heuristic procedure for scheduling stochastic cyclic transfer lines that move jobs between the stations synchronously.

Stochastic cyclic flow lines are comparable to conventional tandem queues with multiple customer classes. While the former produces different types of items in a cyclic order, the latter processes the items in random order. Therefore, stochastic flow lines require a distinct performance analysis method. Nonetheless, it is expected that some ideas for analyzing tandem queues also will be useful for examining stochastic cyclic flow lines.

An important technique for analyzing a tandem queue model is to decompose the model into multiple two-station models, each of which is modeled by an appropriately parameterized single-queue model, and approximate the performance of the tandem queue from the performance estimates of the decomposed single-queue models [4–6, 8]. While Dallery et al. [4, 5] and Gershwin [8] propose such decomposition technique for transfer lines with unreliable machines and finite buffers,

the technique is popularly used for tandem queues. Various decomposed single-queue models and different approximation schemes can be found in the survey on modeling and analysis of tandem queues [6]. We note that most works on tandem queues assume single customer class while a stochastic cyclic flow line processes multiple customer classes simultaneously. It is expected that stochastic cyclic flow lines with blocking require yet another decomposition and approximation method.

In this paper, we examine performance of cyclic flow line models that have finite buffers and processing times of exponential or phase-type distributions. Phase-type distributions are more realistic for modeling processing time distributions since any distribution can be arbitrarily closely approximated by phase-type distributions. While such a cyclic flow line model would be modeled by a finite continuous-time Markov chain, the number of states tends to explode and the chain easily becomes computationally intractable as the number of stations, the buffer capacities, the number of job types, and the number of phases in the processing time distributions increase. Therefore, we present a computationally tractable performance approximation method that decomposes the line model into a number of two-station submodels and appropriately parameterizes the mean processing times of the decomposed submodels. We examine the performance characteristics and report the experimental accuracy of the proposed algorithm. We also compare the performance of cyclic production with that of random order production. The effect of the job processing sequence is also discussed.

2 Stochastic cyclic flow line models with finite buffers

We first explain stochastic cyclic flow line models. We consider a cyclic flow line that consists of $(K + 1)$ stations (S_0, S_1, ..., S_K). Each station has a single machine. The first station S_0 has an unlimited buffer. Each subsequent station $S_i (i = 1, \ldots, K)$ has an input buffer of capacity $B - 1$ (that is, each station has capacity B). Each station can process the next job in the buffer after the previous one completes and leaves the station. A job completed at a station immediately leaves the station and enters the input buffer of the next station. When the next input buffer is full, the job cannot leave the station and waits until the next buffer is available. Such waiting is called *blocking*, more specifically blocking after service (BAS). When there is no job available at the input buffer, the station is idle. This is called *starvation*. The transport times of jobs between the stations are negligible or included in the processing times. The jobs in an input buffer are processed in the order of first come and first service. A job being processed at a station cannot be preempted. We assume that the stations are all reliable and there is no breakdown. There are enough jobs available and hence no shortage or starvation at the first station S_0. The last station S_K has no blocking since there is no next station. The J types of jobs are repetitively loaded into the line in a predefined cyclic order. Therefore, each station repeats the identical cyclic order of processing the jobs. We assume that the processing times of the jobs at a station have exponential or phase-type distributions. The setup times are negligible.

3 Two-station models

We first examine performance of a two-station model that has exponential processing time distributions. We introduce parameters for the two-station model. λ_i and μ_i are the processing rates of job i at station 1 and 2, respectively, and B is the capacity of station 2. The state of the line model is then denoted by (m_1, m_2, n), where n = the number of jobs at station 2 including the job in progress, and m_i = the state of station i. m_i usually indicates the job type being processed at station i. However, when station 1 is being blocked, its state is indicated by $m_1 = b$. $m_2 = s$ means that station 2 is starving. For example, (1,1,3) indicates that both stations 1 and 2 are processing job 1, and 3 jobs at station 2 (2 in the buffer and 1 in progress). $(b, 2, 2)$ represents that station 1 is blocked after processing a job since station 2 has capacity 2. We note that if we know the number of jobs at the buffer and the job type in progress at a station, the job type in progress or just completed at another station is easily determined. By examining the operational behavior of the line model, the state transition diagram is obtained as in Figure 1. Since all event occurrences

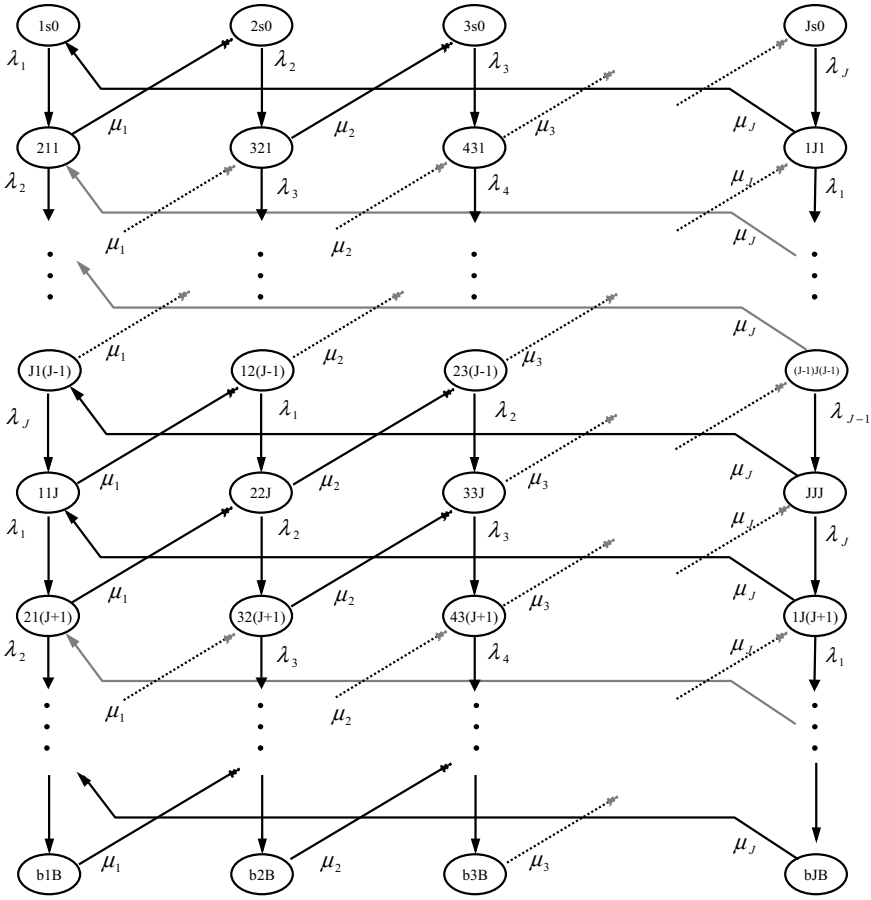


Fig. 1. State transition diagram of a two-station model

are governed by exponential processing times, the state transition process forms a continuous-time Markov chain.

We observe that the diagram repeats an identical structure each multiple of J for state variable n. The transition rates are marked on the corresponding arcs. We therefore expect that the generator of the Markov chain has a repeating pattern. Define $r \equiv \lfloor \frac{B}{J} \rfloor$. We let $\pi(m_1, m_2, n)$ denote the probability that the line is at state (m_1, m_2, n) in the steady state. For exposition convenience, we explain the case of $J = 2$. Define the steady state distribution vector as $\boldsymbol{\pi} \equiv (\pi_s, \pi_0, \pi_1, \ldots, \pi_k, \ldots, \pi_{r-1}, \pi_b)$, where $\pi_s \equiv (\pi(1, s, 0), \pi(2, s, 0))$, $\pi_k \equiv (\pi(2, 1, 2k{+}1), \pi(1, 2, 2k{+}1), \pi(1, 1, 2k{+}2), \pi(2, 2, 2k{+}2))$, $k = 0, 1, \ldots, r{-}1$, and $\pi_b \equiv (\pi(b, 1, B), \pi(b, 2, B))$. From the state transition diagram, we have the generator matrix Q that is represented by block matrices with special structures as

$$Q = \begin{pmatrix} S_1 & S_2 & & \cdots & \\ S_3 & S_4 & A_0 & \cdots & \\ & A_2 & A_1 & A_0 & \cdots \\ & & \vdots & & \\ & & A_2 & A_1 & A_0 \\ & & & A_2 & B_4 & B_3 \\ & & & & B_1 & B_2 \end{pmatrix},$$

where $S_1 = \begin{pmatrix} -\lambda_1 & 0 & \lambda_1 & 0 \\ 0 & -\lambda_2 & 0 & \lambda_2 \\ 0 & \mu_1 & -(\mu_1 + \lambda_2) & 0 \\ \mu_2 & 0 & 0 & -(\mu_2 + \lambda_1) \end{pmatrix}$,

$$S_2 = A_0 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \lambda_2 & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 \end{pmatrix},$$

$$S_4 = B_4 = A_1 = \begin{pmatrix} -(\mu_1{+}\lambda_1) & 0 & \lambda_1 & 0 \\ 0 & -(\mu_2{+}\lambda_2) & 0 & \lambda_2 \\ 0 & \mu_1 & -(\mu_1{+}\lambda_2) & 0 \\ \mu_2 & 0 & 0 & -(\mu_2{+}\lambda_1) \end{pmatrix},$$

$$S_3 = A_2 = \begin{pmatrix} 0 & 0 & 0 & \mu_1 \\ 0 & 0 & \mu_2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},\ B_1 = \begin{pmatrix} 0 & 0 & 0 & \mu_1 \\ 0 & 0 & \mu_2 & 0 \end{pmatrix},\ B_2 = \begin{pmatrix} -\mu_1 & 0 \\ 0 & -\mu_2 \end{pmatrix},$$

$$\text{and}\ \ B_3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ \lambda_2 & 0 \\ 0 & \lambda_1 \end{pmatrix}.$$

Generator Q for $J > 2$ that has the same block structure also can be similarly identified. The size of each block matrix is determined by the number of job types, J, and the station capacity, B. A_i is a $J^2 \times J^2$ square matrix regardless of B.

It is easily seen that Q is irreducible and the finite Markov chain is positive recurrent and hence ergodic. Therefore, the steady state probability $\boldsymbol{\pi}$ is the solution of the balance equation $\boldsymbol{\pi}Q = \mathbf{0}$. The balance equation can be efficiently solved since Q has a special structure called a generalized birth-and-death process. Such a structure is also a special case of the general matrix geometric structure. There are two generally known strategies for solving the balance equation for such a structured generator, matrix geometric technique [19] and recursive technique [3]. Buzacott and Kostelski [3] report that there is no significant difference between the two methods in their accuracy, but the recursive method is more efficient than the matrix geometric algorithm. There can be different implementations of the recursive method depending on the detailed matrix structure. We adapt the recursive algorithm of Hong et al. [12] that is used for two-station tandem queues with random failures of stations.

From $\boldsymbol{\pi}Q = \mathbf{0}$, we obtain the following equations.

$$\pi_s S_1 + \pi_0 S_3 = 0, \tag{1}$$

$$\pi_s S_2 + \pi_0 S_4 + \pi_1 A_2 = 0, \tag{2}$$

$$\pi_{k-1} A_0 + \pi_k A_1 + \pi_{k+1} A_2 = 0, k = 1, 2, \ldots, r-2, \tag{3}$$

$$\pi_{r-2} A_0 + \pi_{r-1} B_4 + \pi_b B_1 = 0, \text{ and} \tag{4}$$

$$\pi_{r-1} B_3 + \pi_b B_2 = 0. \tag{5}$$

From (1), (2), (3), and (4), we derive following relationships. From (1) and (2),

$$\pi_s = \pi_0 T_0, \text{ where } T_0 = -S_3 S_1^{-1}, \text{ and} \tag{6}$$

$$\pi_0 = \pi_1 T_1, \text{ where } T_1 = -A_2 (S_4 + T_0 S_2)^{-1}. \tag{7}$$

From (3), we can derive

$$\pi_k = \pi_{k+1} T_{k+1}, \text{ where } T_{k+1} = -A_2 (A_1 + T_k A_0)^{-1}, k = 1, \ldots, r-2. \tag{8}$$

From (4), we have

$$\pi_{r-1} = \pi_b T_b, \text{ where } T_b = -B_1 (B_4 + T_{r-1} A_0)^{-1}. \tag{9}$$

From the initial value of π_b, $T_0, T_1, T_2, \ldots, T_{r-1}$, and T_b are successively obtained and vector $\boldsymbol{\pi}$ is computed from the normalizing condition:

$$\pi_s \mathbf{1} + \sum_{k=0}^{r-1} \pi_k \mathbf{1} + \pi_b \mathbf{1} = 1, \tag{10}$$

where $\mathbf{1}$ is the column vector of $(1, 1, \ldots, 1)$ with an appropriate dimension. The procedure for computing the performance is summarized as follows.

Algorithm: Two-station

Step 1. Set $\pi_b = \mathbf{1}$ and $SAVE = \pi_b = \mathbf{1}$.
Step 2. Compute π_{r-1} from (9).
Step 3. Compute π_b from (5).
Step 4. If $||SAVE - \pi_b|| \leq \epsilon$, go to Step 5.
Else $SAVE = \pi_b$, and go to Step 2.
Step 5. Compute $\pi_{r-1}, \pi_{r-2}, \ldots, \pi_0, \pi_s$ from (6)–(9).
Step 6. Normalize the steady state distribution vector $\boldsymbol{\pi}$.

Vectors π_b, π_k, and π_s are computed recursively by matrix operations. After taking some initial value of π_b, compute π_k's and π_b recursively until π_b value converges to a finite value. Then, $\boldsymbol{\pi}$ vector is normalized. The steady state queue length distributions, the starvation probability, the blocking probability, the throughput rate, and the mean queue length are computed, respectively, as

$$p_n = \sum_{m_1=1}^{J} \sum_{m_2=1}^{J} \pi(m_1, m_2, n),\ n = 1, \ldots, B,$$

$$p_s = \sum_{m_1=1}^{J} \pi(m_1, s, 0),$$

$$p_b = \sum_{m_2=1}^{J} \pi(b, m_2, B),$$

$$T = \sum_{j=1}^{J} \mu_j [\sum_{y=1}^{B} \sum_{m_1=1}^{J} \pi(m_1, j, y)] + \mu_J \pi(b, J, B), \text{ and}$$

$$L = \sum_{y=1}^{B} \sum_{m_1=1}^{J} \sum_{m_2=1}^{J} y\pi(m_1, m_2, y) + \sum_{m_2=1}^{J} B\pi(b, m_2, B).$$

Note that $1/T$ is the mean cycle time for all types of items. The mean cycle time of job sets is J/T.

4 Models with more than two stations

In order to analyze the performance of a line model with more than two stations, we extend the decomposition technique that has been used for tandem queues [8]. The

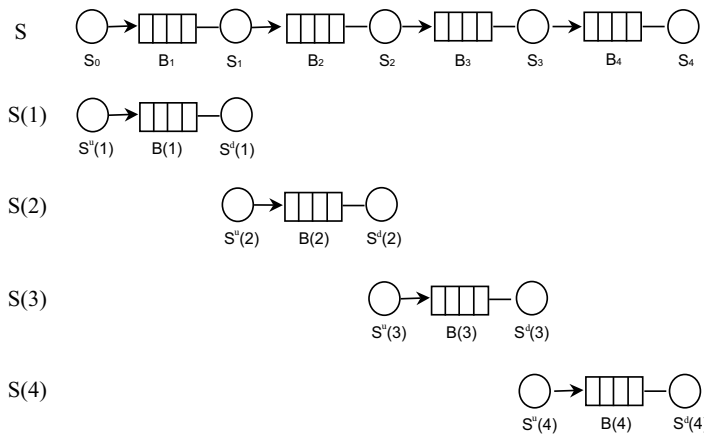


Fig. 2. Two-station decomposition

procedure is outlined as follows. First, the line model is decomposed into K two-station submodels as shown in Figure 2. Each two-station submodel $S(i)$ consists of upstream station $S^u(i)$, downstream station $S^d(i)$, and buffer $B(i)$ between them with the same capacity $B-1$ as in the original line model S. Stations $S^u(i)$ and $S^d(i)$ are parameterized to have the performances close to those of stations S_{i-1} and S_i, respectively, in the original line, which are subject to starvation and blocking.

4.1 Exponential models

We first examine the decomposition method for the case where all processing times are exponentially distributed. Let $t_j(i)$ denote the mean processing time of job j at station i in the original line model. Let $\mathbf{t}(i) \equiv (t_1(i), \ldots, t_J(i))', i = 0, \ldots, K$, be the mean processing time vector at station S_i. Let $\mathbf{t}^u(i) \equiv (t_1^u(i), \ldots, t_J^u(i))'$ and $\mathbf{t}^d(i) \equiv (t_1^d(i), \ldots, t_J^d(i))', i = 1, \ldots, K$, be the mean processing times of $S^u(i)$ and $S^d(i)$ in submodel $S(i)$, respectively. The processing capacity at each station of a decomposed two-station submodel is parameterized to be as close as possible to the effective processing capacity of the corresponding station in the original line. The processing capacity of a station in the original line is reduced due to starvation or blocking at the station. Therefore, the processing times $\mathbf{t}^u(i)$ of the upstream station $S^u(i)$ of each submodel $S(i)$ are extended as much as the delays due to starvation of the corresponding station S_{i-1} in the original line. Similarly, the processing times $\mathbf{t}^d(i)$ of the downstream station $S^d(i)$ of each submodel $S(i)$ are extended as much as the delays due to blocking of the corresponding stations S_i in the original line. However, for the first submodel $S(1)$, the processing times of station $S^u(1)$ are kept same as those of S_0 in the original line. It is because the first station S_0 is never starved. Similarly, for the last submodel $S(K)$, the processing times of $S^d(K)$ are kept same as those of S_K in the original line because the last

station S_K is never blocked. Therefore, we have the following boundary conditions:

$$\mathbf{t}^u(1) = \mathbf{t}(0) \text{ and } \mathbf{t}^d(K) = \mathbf{t}(K). \tag{11}$$

Consider a submodel $S(i)$ such that $1 \leq i < K$. The processing time of job j at the upstream station $S^u(i)$ should be taken as the sum of the processing time of the job at station S_{i-1} and the starvation time of the station in the original line. Suppose that S_{i-1} is starved, i.e., B_{i-1} is empty, at the instant of completion of job j at the station. Since the exact starvation probability at station S_{i-1} in the original line is not available, it is approximated by the starvation probability at the corresponding station $S^d(i-1)$ of the preceding submodel $S(i-1)$, denoted by $p_s(i-1)$, where the preceding submodel was appropriately parameterized. The starved station S_{i-1} is delayed as long as the next job $j+1$'s residual processing time at station S_{i-2}. The residual processing time is approximated to be job $j+1$'s residual processing time at station $S^u(i-1)$ of submodel $S(i-1)$. The residual processing time is exponentially distributed with mean $t^u_{j+1}(i-1)$ because of the memoryless property. The delay due to starvation is regarded to extend the processing time of job $j+1$ at station $S^u(i)$. Therefore, the mean processing times of station $S^u(i)$ in submodel $S(i)$ are parameterized to be

$$\mathbf{t}^u(i) = \mathbf{t}(i-1) + \mathbf{t}^u(i-1) \times p_s(i-1), i = 2, \ldots, K. \tag{12}$$

We observe that the processing times of the upstream station $S^u(i)$ of each submodel $S(i)$ are recursively modified from the mean processing times of the upstream station $S^u(i-1)$ and the starvation probability of the preceding submodel $S(i-1)$. Such kind of recursion is often called *starvation propagation* [6].

Similarly, the mean processing time of job j at the downstream station $S^d(i)$ of each submodel $S(i)$, that is, $t^d_j(i)$, is parameterized to be the sum of the mean processing time of the job at station S_i and the blocking time of the station in the original line. Suppose that S_i is blocked, i.e., B_{i+1} is full at the instant of completion of job j at the station. The blocking probability at station S_i is approximated by the blocking probability at the corresponding station $S^u(i+1)$ of the succeeding submodel $S(i+1)$, denoted by $p_b(i+1)$, where the succeeding submodel was appropriately parameterized. The blocked station waits until the job in progress at $S^d(i+1)$ is finished. The type of the job in progress, $\theta(j)$, is determined from the jobs at station $S^d(i+1)$. The job list is in the sequence of $j-1, j-2, \ldots, 1, J, J-1, \ldots, 2, 1, \ldots, J, J-1, \ldots, 2, 1, J, J-1, \ldots, \theta(j)$, where the number of identical subsequences is appropriately determined. Since the capacity of station $S^d(i+1)$ is B, $j-1+mJ+(J-\theta(j)+1) = B$, where m is an appropriate nonnegative integer and $1 \leq j, \theta(j) \leq J$. From some reasoning, it can be seen that $\theta(j) = j - (B \bmod J) \pmod J$. Therefore, the residual processing time of job $\theta(j)$, for which the upstream station $S^u(i+1)$ of submodel $S(i+1)$ is being blocked, is added to the processing time of job j at the downstream station $S^d(i)$ of submodel $S(i)$, which corresponds to station $S^u(i+1)$. Due to the memoryless property, the residual processing time of job $\theta(j)$ is exponentially distributed with mean $t^d_{\theta(j)}(i+1)$. We let $\mathbf{t}^d_\theta(i) \equiv (t^d_{\theta(1)}(i), \ldots, t^d_{\theta(J)}(i))'$. By matching the index of the blocked job with that of the job in progress at the next station, the mean processing times of station $S^d(i)$ of submodel $S(i)$ are parameterized to be

$$\mathbf{t}^d(i) = \mathbf{t}(i) + \mathbf{t}^d_\theta(i+1) \times p_b(i+1), i = 1, \ldots, K-1. \tag{13}$$

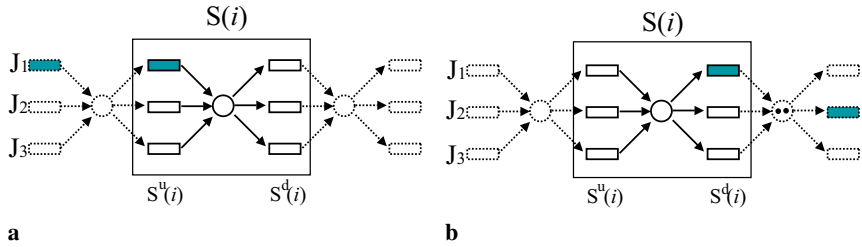


Fig. 3a,b. Propagation of starvation and blocking. **a** Starvation propagation. **b** Blocking propagation

The mean processing times of the downstream station $S^d(i)$ of each submodel $S(i)$ are recursively computed from the mean processing times of the downstream station $S^d(i+1)$ and the blocking probability of the succeeding submodel $S(i+1)$. Such kind of recursion is often called *blocking propagation* [6]. Figure 3 illustrates the propagation mechanism of starvation and blocking. The mean processing time of a specific job(shaded box) at $S(i)$ is elongated by the starvation (or blocking) time, which is the remaining processing time of the job in progress at the preceding (or succeeding) station.

We now have a simultaneous equation system that has $2JK$ unknown parameters, $\mathbf{t}^u(i)$ and $\mathbf{t}^d(i), i = 1, \ldots, K$, and $2JK$ independent equations. The performance of the original line is then approximated by the decomposed submodels that are parameterized by the decomposition procedure summarized below. Once each submodel is parameterized based on the starvation and blocking probabilities of the adjacent submodels, the starvation and blocking probabilities of each submodel change due to the modifications in its processing times. Therefore, the submodels should be parameterized again based on the changed starvation and blocking probabilities. The algorithm repeats such computing cycle until the process times of the submodels do not change anymore. We note that our decomposition algorithm is structurally similar to the well-known decomposition procedures of [4,5,8] for conventional transfer lines or tandem queues. It is known that such algorithms based on propagation of starvation and blocking converge. In fact, our algorithm converged quickly, mostly within 10 iterations, for all experimental cases, which are explained in Section 5. The computation times were within 1∼2 CPU seconds at Pentimum 1 GHz PC.

Algorithm: Decomposition

Step 1. Initialize.
Set $\mathbf{t}^u(1) \equiv \mathbf{t}(0)$ and $\mathbf{t}^d(K) \equiv \mathbf{t}(K)$.
Let $\mathbf{t}^d(i) = \mathbf{t}(i), i = 1, \ldots, K-1$.
Step 2. For $i = 2, \ldots, K$,
compute $p_s(i-1)$ from submodel $S(i-1)$, and
compute $\mathbf{t}^u(i)$ using equation (12).
Step 3. For $i = K-1, \ldots, 1$,
compute $p_b(i+1)$ from submodel $S(i+1)$, and
compute $\mathbf{t}^d(i)$ using equation (13).

Step 4. Go to step 2 until $\mathbf{t}^u(i)$ and $\mathbf{t}^d(i)$ converge.

4.2 Phase-type distribution models

We now explain how the proposed decomposition method can be extended to the case of phase-type distributions. A phase-type distribution with k phases is represented as $(1-\beta_1)exp(\mu_1) + (1-\beta_2)\beta_1 exp(\mu_1) * exp(\mu_2) + \ldots + \prod_{j=1}^{k-1}\beta_j[exp(\mu_1) * \ldots * exp(\mu_k)]$, where $k \geq 1$ is an integer, $\beta_j \in (0,1), j = 1,\ldots,k-1$, and $\beta_k = 0$. k-Erlang, Coxian, and hyper-exponential distributions are the special cases. For a two-station model with phase-type processing time distributions, a Markov chain model can be constructed once the state is taken to include the phase of the job in progress at each station. It is because the time to complete each phase has an exponential distribution and hence all event occurrences are governed by exponential time distributions. The state of the two-station model can be represented by (m_1, m_2, n), where $m_i = j_l$, the current phase l of job j at station i. For notational convenience, we assume that the processing time distributions have the same number of phases l. For example, $(1_2, 1_1, 3)$ represents that job 1 is in phase 2 at station 1, job 1 is in phase 1 at station 2 while 3 jobs are at station 2. The performance computing procedure is then similar to that for the exponential case except that the size of the generator matrix increases due to the multiple phases. Using a two-station, two-job case with Coxian-2 distributions, we outline how the decomposition method can be extended for the cases with phase-type distributions. Let $(\mu_{ij1}, \beta_{ij}, \mu_{ij2})$ be the parameters of Coxian-2 distribution for processing time of job j at station i. μ_{ijl} is the mean processing rate of phase l job j at station i. β_{ij} is the probability that job j enters the second phase after completion of the first phase at station i. Hence, a job leaves station i immediately after completion of the first phase at station i with probability $1-\beta_{ij}$. Let $t_{ijl} \equiv 1/\mu_{ijl}, \mathbf{t}_j(i) \equiv (t_{ij1}, t_{ij2})'$, and $\mathbf{t}(i) \equiv (\mathbf{t}_1(i), \ldots, \mathbf{t}_J(i))', i = 0, \ldots, K$, be the mean processing time vector of the jobs at station i. The starvation time at station S_{i-1} can be approximated by the residual processing time of job j in progress at station $S^u(i-1)$, which has a Coxian distribution with parameter $(1/t^u_{(i-1)j1}, \beta_{(i-1)j}, 1/t^u_{(i-1)j2})$. To find the mean residual processing time, we should know $\alpha_{jl}(i-1)$, the probability of station $S^u(i-1)$ being in phase l for processing job j when starvation occurs at the downstream station $S^d(i-1)$. This can be derived from the two-station model. We note that only the mean time for processing the first phase of job j is extended because the completed job enters the first phase at the downstream station. Therefore, the mean processing time of job j at station $S^u(i)$ is parameterized to be

$$\begin{aligned}\mathbf{t}^u_j(i) &= \mathbf{t}_j(i-1) + (1,0)'\boldsymbol{\alpha}_j(i-1)\boldsymbol{\beta}_j(i-1)\mathbf{t}^u_j(i-1)p_s(i-1),\\ &\quad i = 2,\ldots,K,\end{aligned} \tag{14}$$

where $\boldsymbol{\alpha}_j(i-1) \equiv (\alpha_{j1}(i-1), \alpha_{j2}(i-1))$, and $\boldsymbol{\beta}_j(i-1) \equiv \begin{pmatrix} 1 & \beta_{(i-1)j} \\ 0 & 1 \end{pmatrix}$.

The blocking time of job j at station S_i is approximated by the residual processing time of job $\theta(j)$ in progress at station $S^d(i+1)$, which has a Coxian distribution

Table 1. Effects of traffic intensity for M(3,3)

Buffer size	W(0.5,0.5,3)			Exact			Error		
	CT	L_1	L_2	CT	L_1	L_2	CT	L_1	L_2
1	36.774	0.652	0.540	37.189	0.670	0.537	−1.12(%)	−2.69(%)	0.56(%)
10	30.058	1.049	1.078	30.030	1.058	1.082	0.09(%)	−0.85(%)	0.37(%)
20	30.000	1.060	1.104	30.003	1.060	1.093	−0.01(%)	0.00(%)	1.01(%)
Buffer size	W(0.8,0.8,3)			Exact			Error		
	CT	L_1	L_2	CT	L_1	L_2	CT	L_1	L_2
1	46.289	0.996	0.774	46.447	0.998	0.766	−0.34(%)	−0.20(%)	1.04(%)
10	31.333	3.080	2.885	31.466	3.170	2.879	−0.42(%)	−2.84(%)	0.21(%)
20	30.164	4.090	4.363	30.112	4.138	4.216	0.17(%)	−1.16(%)	1.11(%)

with parameter $(1/t^d_{(i+1)\theta(j)1}, \beta_{(i+1)\theta(j)}, 1/t^d_{(i+1)\theta(j)2})$. Similarly as for the starvation time, we obtain the mean residual processing time using the probability $\gamma_{jl}(i+1)$ that station $S^d(i+1)$ is in phase l for processing job j when blocking occurs at the upstream station. Therefore, the mean processing time of job j at station $S^d(i)$ is parameterized to be

$$\mathbf{t}^d_j(i) = \mathbf{t}_j(i) + (1,1)'\boldsymbol{\gamma}_{\theta(j)}(i+1)\boldsymbol{\beta}_{\theta(j)}(i+1)\mathbf{t}^d_{\theta(j)}(i+1)p_b(i+1), \quad i = 1,\ldots,K-1, \tag{15}$$

where $\boldsymbol{\gamma}_{\theta(j)}(i+1) = (\gamma_{\theta(j)1}(i+1), \gamma_{\theta(j)2}(i+1))$. Together with the boundary conditions similar to equation (11), we obtain the parameters for the two-station submodels.

5 Experiments

We investigate the accuracy of the decomposition algorithm. A cyclic flow line model with K stations and J jobs is denoted by $M(K,J)$. We let $W(\rho_1,\ldots,\rho_K,\upsilon)$ indicate the workloads at the stations, where the workload at station i is $\rho_i \equiv \sum_{j=1}^{J} t_j(i)/\sum_{j=1}^{J} t_j(0)$. *Job variation* υ indicates the relative variations of the mean processing times of the jobs at each station, defined as the ratio of the maximum to the minimum of the mean processing times of the jobs at each station. It is chosen identical for all stations. We compute the cycle time (CT) and the mean queue lengths at the stations (L_i's), the total mean queue length in the line ($L \equiv \sum_{i=1}^{K} L_i$). Table 1 shows the performance for $M(3,3)$ models with different buffer capacities. Each station has the same buffer capacity. For such small line models, the exact performance values are computed from the continuous-time Markov chain for the whole line.

We explain the error behavior of our proposed approximation algorithm that is shown in Table 1. 'Error' in the table indicates the percent deviation from the exact value. The proposed algorithm approximates the performance of each station by

that of the corresponding decomposed two-station submodel. The submodels are not independent but interact with each other through blocking and starvation. The interactions are approximately modeled by accommodating the process times of each submodel $S(i)$ based on the starvation and blocking probabilities at the adjacent submodels $S(i-1)$ and $S(i+1)$, respectively (see equations (12) and (13)). The starvation and blocking probabilities are approximate values that are estimated from the adjacent submodels. Therefore, when the starvation or blocking probabilities are higher, the performance of each submodel is more affected by starvation or blocking at the adjacent submodels, and hence the performance estimates based on the decomposed submodels tend to have larger errors. As the traffic intensity increases, the blocking probability increases while the starvation probability decreases. Therefore, it is hard to exactly figure out how the approximation errors are affected by the traffic intensity values. They depend on the relative size of the starvation probability decrement to the blocking probability increment, which is also affected by the buffer size. When the buffer size is 10 or 20, the higher traffic intensity causes the higher blocking probability and hence tends to increase errors although the traffic intensity increment reduces the starvation probability. However, for the case of buffer size 1, we observe that the higher traffic intensity tends to decrease the errors. A conjecture for this reversed error behavior follows. Buffer size 1 implies that there is no waiting place except the machine itself and hence the blocking probability is extremely high, close to 1. Therefore, the blocking probability increment due to the traffic intensity increment is relatively small as compared to the starvation probability decrement. Consequently, the no-buffer case with higher traffic intensity has less error. Nonetheless, we observe that for a given traffic intensity, the smaller buffer tends to make larger errors. Our further experiments for longer lines such as $M(12,5)$ with $v = 3$ and high workloads indicate estimation errors within 2$\sim$3%.

Table 2 shows the performance estimates for 5-station cases with different values of job variation v. The table is visualized by Figure 4. In the figure, for better visualization, the performance values for two different job variation cases for a given buffer size are marked with a small horizontal space. The simulation estimates and 99% confidence intervals in the table are obtained by 100 simulation replications. Since there is no exact method available for these larger models, we list two types of estimates, one from simulation and another from our approximation algorithm. We observe that job variation increment from 3 to 10 causes the significant increase in the cycle time. Such increase is salient when the buffer size is small and hence the blocking probability is high. The job variation can be considered as another kind of variation to the processing times.

The primary purpose of Table 2 and Figure 4 are to show the effects of job variation on the performance. Nonetheless, the table shows the relative accuracy of our estimates in comparison to the simulation estimates. The relative errors are listed in the last two columns. In order to compare the two types of estimates, it is desirable to reduce the confidence interval so that its width is much smaller than the errors of our proposed algorithm. However, the confidence interval may not be significantly reduced by increasing the number of replications due to numerical errors and incomplete randomness of the pseudo-random number streams in the

Table 2. Effects of job variation for M(5,3)

Buffer size	W(0.5,0.5,0.5,0.5,3) CT	L	Simulation CT	L	Error CT	L
1	37.532	2.486	37.968±0.168	2.473±0.014	−1.15(%)	0.53(%)
5	30.816	4.125	30.761±0.192	4.123±0.055	0.18(%)	0.05(%)
10	30.070	4.392	30.049±0.175	4.485±0.069	−0.07(%)	−2.07(%)
20	30.000	4.506	30.005±0.161	4.521±0.065	−0.02(%)	−0.19(%)
Buffer size	W(0.5,0.5,0.5,0.5,10) CT	L	Simulation CT	L	Error CT	L
1	39.004	2.585	39.465±0.193	2.528±0.018	−1.17(%)	2.25(%)
5	31.115	4.636	31.232±0.177	4.628±0.062	−0.38(%)	0.17(%)
10	30.090	5.073	30.125±0.167	5.143±0.088	−0.12(%)	−1.36(%)
20	30.002	5.229	30.000±0.168	5.242±0.098	0.01(%)	−0.25(%)

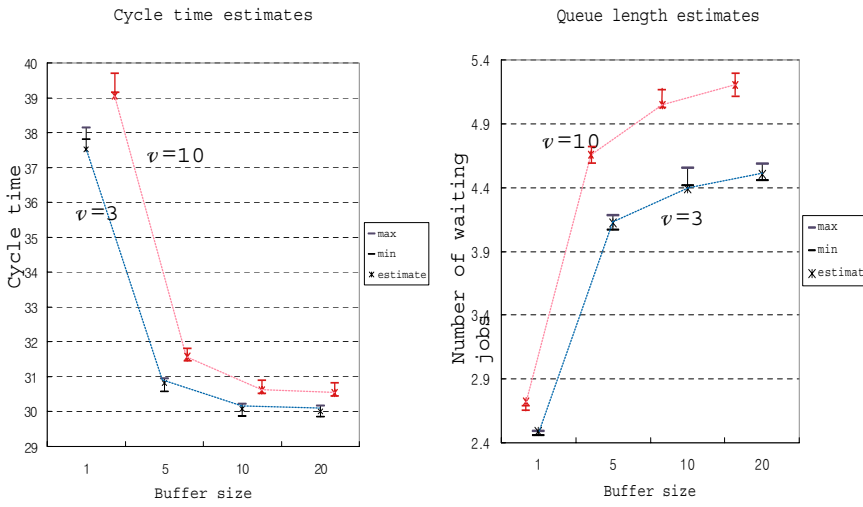


Fig. 4. Effects of job variation for $M(5,3)$

computer simulation program, which are hard to control. In fact, Figure 4 shows that the confidence intervals are not sufficiently reduced. Nonetheless, we roughly figure out the relative accuracy of the estimates by our proposed algorithm and how much job variation increment increases the errors of our estimates. For instance, among 8 comparisons in the table, 6 estimates by our approximation algorithm fall within the confidence intervals. The figure shows that the errors tend to be larger when the buffer size is smaller. It is because when the blocking probabilities are higher, the blocking propagation procedure amplifies the blocking approximation errors more. We also observe that job variation increment tends to increase the errors of our algorithm in the cycle time estimates. It is because higher job variation causes more blocking especially when the buffer size is small.

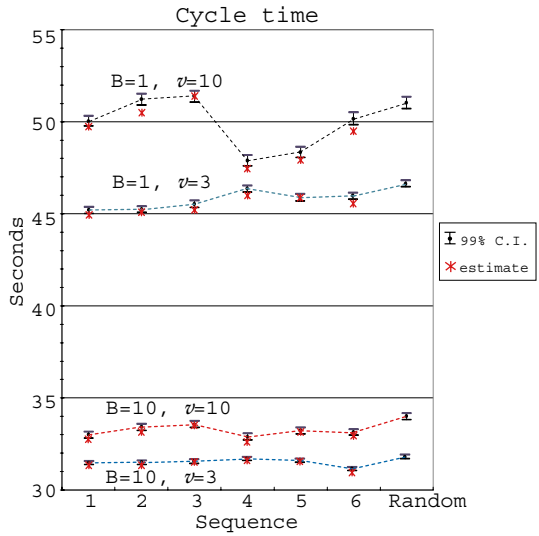


Fig. 5. Effects of job processing orders for $M(4,4)$

6 Effects of job processing sequences and comparison with random order processing

We examine the effects of the job processing sequence on the performance and compare the performance with that of random order job processing. There are $(J-1)!$ cyclic sequences of processing J types of jobs.

Since the exact method is not available for the larger cases such as $M(4,4)$, we should resort to simulation or our proposed approximation method. Although the errors of our approximation method are small, mostly within 1$\sim$2 percent, they are biased. However, simulation tends to give less biased point estimates because the point estimates are obtained by averaging out the estimates from many independent replications. Furthermore, the confidence intervals provide information on the relative accuracy of the point estimates. Therefore, we primarily use simulation estimates to have more consistent performance estimates for examining the effects of the job processing order. Table 3 shows the performance estimates for each processing sequence for line model M(4,4) with W(0.8, 0.8, 0.8, v). Figure 5 visualizes the cycle time estimates. The point estimates and the 99% confidence intervals are obtained from 100 replications of simulation. CV in the table indicates 100 times of the coefficient of variation of the performance estimates for the sequences. For random order processing, the first station that is considered as an arrival generator is modified to generate arrivals in random order while the long-run proportion of job types is maintained. From Figure 5, we observe that while the mean processing times are kept to be identical for all four cases, the relative performance between the processing sequences are different for each case. Therefore, the processing sequence should be carefully taken based on the performance estimates. Our proposed procedure can efficiently compute the approximate cycle time estimates within 1$\sim$2 percent errors. As seen in Figures 4 and 5, the estimates

Table 3. Effects of job processing orders for M(4,4) with W(0.8,0.8,0.8,υ)

	Order	B=1		B=10	
		CT	L	CT	L
υ=3	1	45.186±0.173	2.390±0.013	31.428±0.110	9.144±0.101
	2	45.235±0.163	2.431±0.015	31.444±0.103	9.247±0.097
SCFL	3	45.516±0.185	2.404±0.014	31.514±0.107	9.260±0.108
	4	46.331±0.177	2.453±0.014	31.652±0.106	9.356±0.099
	5	45.860±0.187	2.397±0.014	31.556±0.107	9.148±0.094
	6	45.952±0.185	2.421±0.014	31.099±0.101	9.228± 0.109
	CV	0.978	0.979	0.603	0.856
Random		46.615±0.179	2.573±0.013	31.758±0.107	9.453±0.106
	Order	B=1		B=10	
		CT	Q	CT	Q
υ=10	1	50.030±0.274	2.395±0.021	32.993±0.179	10.170±0.129
	2	51.204±0.301	2.494±0.018	33.404±0.186	10.329±0.128
SCFL	3	51.368±0.312	2.322±0.019	33.554±0.177	10.303±0.138
	4	47.883±0.286	2.374±0.021	32.885±0.184	10.106±0.029
	5	48.346±0.286	2.483±0.020	33.215±0.187	10.174±0.027
	6	50.164±0.342	2.387±0.020	33.129±0.174	10.219±0.137
	CV	2.894	2.764	0.755	0.834
Random		51.029±0.321	2.596±0.021	33.987±0.178	10.563±0.136

mostly fall within or are close to the confidence intervals. Further, Figure 5 shows that their changes for different sequences are consistent with those of the simulation estimates. Even though our estimates tend to have larger errors when the buffer size is smaller, the performance differences between the processing sequences become larger for such case. Therefore, our approximation procedure can be effectively used for selecting the optimal or near-optimal processing sequence.

We observe that the processing sequence significantly affects the performance, especially when job variation υ is high and the buffer sizes are small. The cyclic processing sequences outperform random order processing in most cases. When the job variation is higher and the buffer capacities are smaller, the cyclic sequences have larger performance differences, but the optimal cyclic sequence has much better performance than random order processing. A good choice of the processing sequence tends to minimize both the cycle time and the queue length.

7 Final remarks

We proposed a procedure for efficiently computing approximate performance estimates of stochastic cyclic flow line models with finite buffers, where processing times have exponential or phase-type distributions. For two-station models, we developed an exact computing procedure by making use of the matrix geometric structure. We identified that the popular method of parameterizing the decomposed submodels by propagating starvation and blocking through the adjacents submodels of a tandem queue also can be effectively extended to cyclic flow shops with blocking. We also found that the job processing sequence significantly affects the performance especially when the job variation is large and the buffer capacities are small. It was shown that cyclic production has better performance than random order production. Future topics include reversibility and buffer allocation characteristics.

References

1. Ahmadi RH, Wurgaft H (1994) Design for synchronized flow manufacturing. Management Science 40(11): 1469–1483
2. Bowman RA, Muckstadt JA (1993) Stochastic analysis of cyclic schedules. Operations Research 41(5): 947–958
3. Buzacott JA, Kostelski D (1987) Matrix-geometric and recursive algorithm solution of a two-stage unreliable flow line. IIE Transaction 19(4): 429–438
4. Dallery Y, David R, Xie XL (1988) An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers. IIE Transactions 20(3): 280–283
5. Dallery Y, David R, Xie XL (1989) Approximate analysis of transfer lines with unreliable machines and finite buffers. IEEE Transactions on Automatic Control 34(9): 943–953
6. Dallery Y, Gershwin SB (1992) Manufacturing flow lines: a review of models and analytical results. Queueing Systems 12(1): 3–94
7. Dobson G, Yano CA (1994) Cyclic scheduling to minimize inventory in a batch flow line. European Journal of Operational Research 75(2): 441–461
8. Gershwin SB (1987) An efficient decomposition method for the approximate evaluation of tandem queues with finite storage space and blocking. Operations Research 35(2): 291–305
9. Gershwin SB, Schick IC (1983) Modeling and analysis of three-stage transfer lines with unreliable machines and finite buffers. Operations Research 31(2): 354–380
10. Graves SC, Meal HC, Stefek D, Zeghmi AH (1983) Scheduling of re-entrant flow shops. Journal of Operations Management 3(4): 197–207
11. Hall NG, Lee TE, Posner ME (2002) The complexity of cyclic shop scheduling problems. Journal of Scheduling 5(4): 307–327
12. Hong Y, Glassey CR, Seong D (1992) The analysis of a production line with unreliable machines and random processing times. IIE Transactions 24(1): 77–83
13. Karabati S, Tan B (1998) Stochastic cyclic scheduling problem in synchronous assembly and production lines. The Journal of the Operational Research Society 49(11): 1173–1187
14. Lee TE, Posner ME (1997) Performance measures and schedules in periodic job shops. Operations Research 45(1): 72–91
15. Lee TE (2000) Stable earliest starting schedules for cyclic job shops: a linear system approach. International Journal of Flexible Manufacturing Systems 12(1): 59–80

16. Seo JW, Lee TE (2002) Steady state analysis of cyclic job shops with overtaking. International Journal of Flexible Manufacturing Systems 14(4): 291–318
17. Kim JH, Lee TE, Lee HY, Park DB (2003) Scheduling analysis of time-constrained dual-armed cluster tools. IEEE Transactions on Semiconductor Manufacturing 16(3): 521–534
18. McCormick ST, Pinedo ML, Shenker S, Wolf B (1989) Sequencing in an assembly line with blocking to minimize cycle time. Operations Research 37(6): 925–935
19. Neuts MF (1981) Matrix-geometric solutions in stochastic models: an algorithmic approach. The Johns Hopkins University Press, Baltimore, MD
20. Rao US, Jackson PL (1996) Estimating performance measures in repetitive manufacturing environments via stochastic cyclic scheduling. IIE Transactions 28(11): 929–939
21. Seo JW, Lee TE (1996) Stochastic cyclic flow lines: non-blocking, Markovian models. Journal of Operational Research Society 49(5): 537–548
22. Wittrock RJ (1985) Scheduling algorithm for flexible flow lines. IBM Journal of Research and Development 29(4): 401–412
23. Zhang H, Graves SC (1997) Cyclic scheduling in a stochastic environment. Operations Research 45(6): 894–903

Section III: Queueing Network Models of Manufacturing Systems

Performance analysis of multi-server tandem queues with finite buffers and blocking

Marcel van Vuuren[1], **Ivo J.B.F. Adan**[1], **and Simone A.E. Resing-Sassen**[2]

[1] Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands (e-mail: m.v.vuuren@tue.nl, i.j.b.f.adan@tue.nl)
[2] CQM BV, P.O. Box 414, 5600 AK Eindhoven, The Netherlands (e-mail: resing@cqm.nl)

Abstract. In this paper we study multi-server tandem queues with finite buffers and blocking after service. The service times are generally distributed. We develop an efficient approximation method to determine performance characteristics such as the throughput and mean sojourn times. The method is based on decomposition into two-station subsystems, the parameters of which are determined by iteration. For the analysis of the subsystems we developed a spectral expansion method. Comparison with simulation shows that the approximation method produces accurate results. So it is useful for the design and analysis of production lines.

Keywords: Approximation – Blocking – Decomposition – Finite buffers – Multi-server tandem queues – Production lines – Spectral expansion

1 Introduction

Queueing networks with finite buffers have been studied extensively in the literature; see, e.g., Dallery and Gershwin [6], Perros [17, 18], and Perros and Altiok [19], and the references therein. Most studies, however, consider *single*-server models. The few references dealing with *multi*-server models typically assume exponential service times. In this paper we focus on multi-server tandem queues with general service times, finite buffers and Blocking After Service (BAS).

Models with finite buffers and phase-type service times can be represented by finite state Markov chains. Hence, in theory, they can be analyzed exactly. However, the number of states of the Markov chain can be very large, which makes numerical solutions intractable. In practice, only small systems with one or two queues can be solved exactly; for exact methods we refer to Perros [18].

We develop an efficient method to approximate performance characteristics such as the throughput and the mean sojourn time. The method only needs the first two moments of the service time and it decomposes the tandem queue into

subsystems with one buffer. fitted on Each multi-server subsystem is approximated by a single (super) server system with state dependent arrival and departure rates, the queue length distribution of which can be efficiently computed by a spectral expansion method. The parameters of the inter-arrival and service times of each subsystem are determined by an iterative algorithm. Numerical results show that this method produces accurate estimates for important performance characteristics as the throughput and the mean sojourn time.

Decomposition techniques have also been used by, e.g., Buzacott [2], Dallery et al. [5], Perros [18], and Kerbache and MacGregor Smith [11]. These papers deal with single-server queueing networks. Methods for multi-server queueing networks with finite buffers are presented by Tahilramani et al. [21], Jain and MacGregor Smith [9], and Cruz et al. [3,4]. These methods, however, do not assume general service times. An excellent survey on the analysis of manufacturing flow lines with finite buffers is presented by Dallery and Gershwin [6].

In the analysis of queueing networks with blocking three basic approaches can be distinguished. The first approach decomposes the network into subsystems and the parameters of the inter-arrival and service times of the subsystems are determined iteratively. This is the most common approach. It involves three steps:

1. Characterize the subsystems;
2. Derive a set of equations that determine the unknown parameters of each subsystem;
3. Develop an iterative algorithm to solve these equations.

This approach is treated in Perros' book [18] and in the survey of Dallery and Gershwin [6]. The approach in this paper also involves the three steps mentioned above, as we will explain in Section 5. There are also decomposition methods available for finite buffer models with some special features, such as assembly/disassembly systems (see Gershin and Burman [7]) and systems with multiple failure modes (see Tolio et al. [23]).

The second approach is also based on decomposition of the network, but instead of iteratively determining the parameters of the inter-arrival and service times of the subsystems, holding nodes are added to represent blocking. This so-called expansion method has been introduced by Kerbache and Smith [11]. The expansion method has been successfully used to model tandem queues with the following kinds of nodes: $M/G/1/K$ [20], $M/M/C/K$ [9] and $M/G/C/C$ [3,4].

The expansion method consist of the following three stages:

1. Network reconfiguration;
2. Parameter estimation;
3. Feedback elimination.

This method is very efficient; it produces accurate results when the buffers are large.

The third approach has been introduced by Kouvatsos and Xenios [12]. They developed a method based on the maximum entropy method (MEM) to analyze single-server networks. Here, holding nodes are also used and the characteristics of the queues are determined iteratively. For each subsystem in the network the queue-length distribution is determined by using a maximum entropy method. This

algorithm is a linear program where the entropy of the queue-length distribution is maximized subject to a number of constraints. For more information we refer the reader to [12]. This method has been implemented in QNAT by Tahilramani et al. [21]; they also extended the method to multi-server networks. This method works well; the average error in the throughput is typically around 5%.

There are also several methods available for optimizing tandem queues with finite buffers. For example, Hillier and So [8] give some insight into the general form of the optimal design of tandem queues with the expected service times, the queue capacities and the number of servers at each station as the decision variables. Li et al. [13] have developed a method for optimization of tandem queues using techniques and concepts like simulation, critical path and perturbation analysis.

The paper is organized as follows. In Section 2 we introduce the tandem queue and its decomposition. In the section thereafter we elaborate on the arrivals at and departures from the subsystems. The spectral expansion method for analyzing the subsystems is discussed in Section 4. Section 5 describes the iterative algorithm. Numerical results are presented in Section 6. The results of the approximation method are compared with simulation and with QNAT. Finally, Section 7 contains some concluding remarks.

2 Model and decomposition

We consider a tandem queue (L) with M server-groups and $M-1$ buffers B_i, $i = 1, \ldots, M-1$, of size b_i in between. The server-groups are labelled M_i, $i = 0, \ldots, M-1$; server-group M_i has m_i parallel identical servers. The random variable S_i denotes the service time of a server in group M_i; S_i is generally distributed with rate $\mu_{p,i}$ (and thus with mean $1/\mu_{p,i}$) and coefficient of variation $c_{p,i}$. Each server can serve one customer at a time and the customers are served in order of arrival. The servers of M_0 are never starved and we consider the BAS blocking protocol. Figure 1 shows a tandem queue with four server groups.

The tandem queue L is decomposed into $M-1$ subsystems $L_1, L_2, \ldots, L_{M-1}$. Subsystem L_i consists of a finite buffer of size b_i, m_{i-1} so-called arrival servers in front of the buffer, and m_i so-called departure servers after the buffer. The arrival and departure servers are virtual servers who describe the arrivals to a buffer and the departures from a buffer. The decomposition of L is shown in Figure 1.

The random variable A_i denotes the service time of an arrival-server in subsystem L_i, $i = 1, \ldots, M-1$. This random variable represents the service time of a server in server-group M_{i-1} *including possible starvation* of this server. The random variable D_i denotes the service time of a departure-server in subsystem L_i; it represents the service time of a server in server-group M_i *including possible blocking* of this server. Let us indicate the rates of A_i and D_i by $\mu_{a,i}$ and $\mu_{d,i}$ and their coefficients of variation by $c_{a,i}$ and $c_{d,i}$, respectively. If these characteristics are known, we are able to approximate the queue-length distribution of each subsystem. Then, by using the queue-length distribution we can also approximate characteristics of the complete tandem queue, such as the throughput and mean sojourn time.

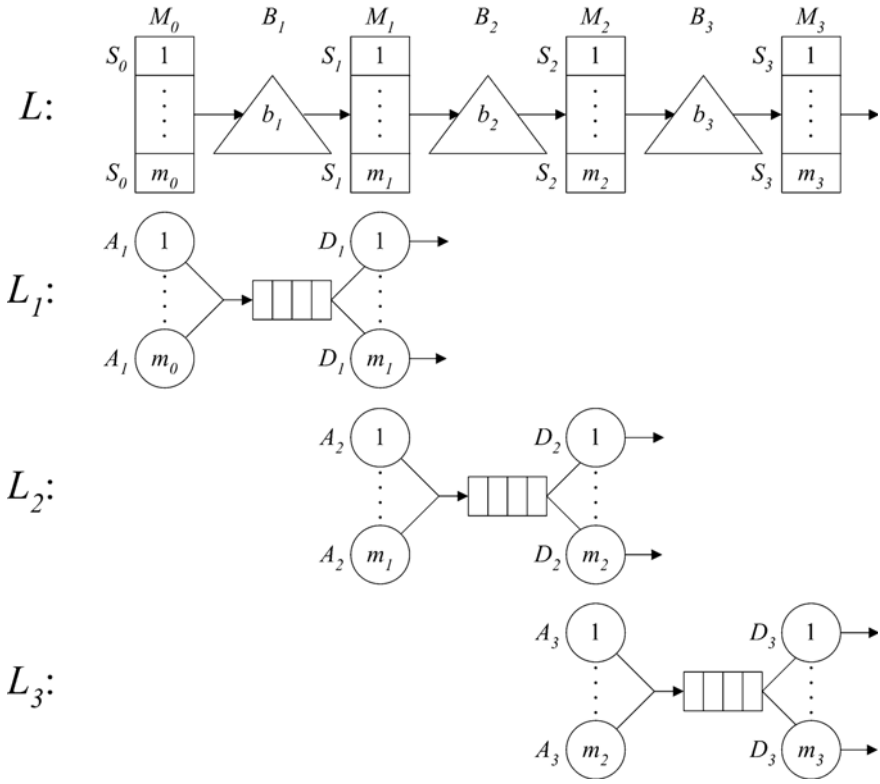


Fig. 1. The tandem queue L and its decomposition into three subsystems L_1, L_2 and L_3

3 Service times of arrival and departure servers

In this section we describe how the service times of the arrival and departure servers in subsystem L_i are modelled.

The service-time D_i of a departure-server in subsystem L_i is approximated as follows. We define $b_{i,j}$ as the probability that just after service completion of a server in server-group M_i, exactly j servers of server-group M_i are blocked. This means that, with probability $b_{i,j}$, a server in server-group M_i has to wait for one *residual* inter-departure time and $j-1$ full inter-departure times of the *next server-group* M_{i+1} before the customer can leave the server. The inter-departure times of server-group M_{i+1} are assumed to be independent and distributed as the inter-departure times of the *superposition* of m_{i+1} independent service processes, each with service times D_{i+1}; the residual inter-departure time is approximated by the equilibrium residual inter-departure time of the superposition of these service processes. Let the random variable SD_{i+1} denote the inter-departure time of server-group M_{i+1} and RSD_{i+1} the residual inter-departure time. Figure 2 displays a representation of the service time of a departure-server of subsystem L_i.

In the appendix it is explained how the rates and coefficients of variation of SD_{i+1} and RSD_{i+1} can be determined. If also the blocking probabilities $b_{i,j}$

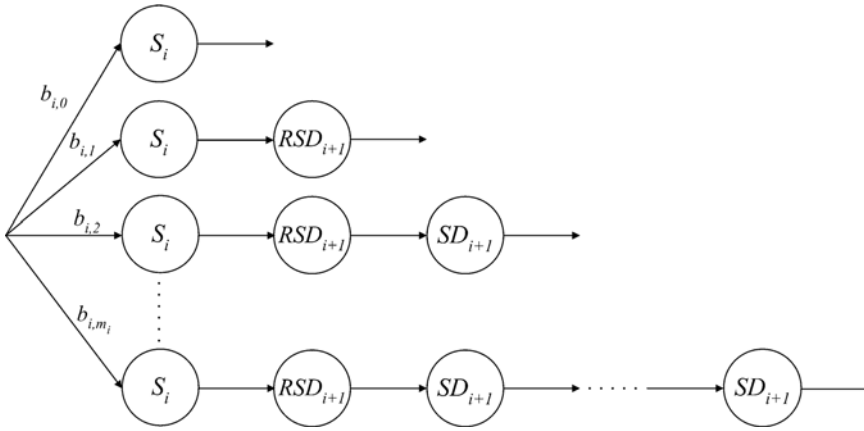


Fig. 2. Representation of the service time D_i of a departure-server of subsystem L_i

are known, then we can determine the rate $\mu_{d,i}$ and coefficient of variation $c_{d,i}$ of the service time D_i of a departure-server of subsystem L_i. The distribution of D_i is approximated by fitting an Erlang$_{k-1,k}$ or Coxian$_2$ distribution on $\mu_{d,i}$ and $c_{d,i}$, depending on whether $c_{d,i}^2$ is less or greater than $1/2$. More specifically, if $c_{d,i}^2 > 1/2$, then the rate and coefficient of variation of the Coxian$_2$ distribution with density

$$f(t) = (1-q)\mu_1 e^{-\mu_1 t} + q\frac{\mu_1\mu_2}{\mu_1-\mu_2}\left(e^{-\mu_2 t} - e^{-\mu_1 t}\right), \qquad t \geq 0,$$

matches with $\mu_{d,i}$ and $c_{d,i}$, provided the parameters μ_1, μ_2 and q are chosen as (cf. Marie [14]):

$$\mu_1 = 2\mu_{d,i}, \quad q = \frac{1}{2c_{d,i}^2}, \quad \mu_2 = \mu_1 q. \tag{1}$$

If $1/k \leq c_{d,i}^2 \leq 1/(k-1)$ for some $k > 2$, then the rate and coefficient of variation of the Erlang$_{k-1,k}$ with density

$$f(t) = p\mu^{k-1}\frac{t^{k-2}}{(k-2)!}e^{-\mu t} + (1-p)\mu^k\frac{t^{k-1}}{(k-1)!}e^{-\mu t}, \qquad t \geq 0,$$

matches with $\mu_{d,i}$ and $c_{d,i}$ if the parameters μ and p are chosen as (cf. Tijms [22]):

$$p = \frac{kc_{d,i}^2 - \sqrt{k(1+c_{d,i}^2) - k^2c_{d,i}^2}}{1+c_{d,i}^2}, \quad \mu = (k-p)\mu_{d,i}. \tag{2}$$

Of course, also other phase-type distributions may be fitted on the rate and coefficient of variation of D_i, but numerical experiments suggest that other distributions only have a minor effect on the results, as shown in [10].

The service times A_i of the arrival-servers in subsystem L_i are modelled similarly. Instead of $b_{i,j}$ we now use $s_{i,j}$ defined as the probability that just after service completion of a server in server-group M_i, exactly j servers of M_i are starved. This means that, with probability $s_{i,j}$, a server in server-group M_i has to wait one

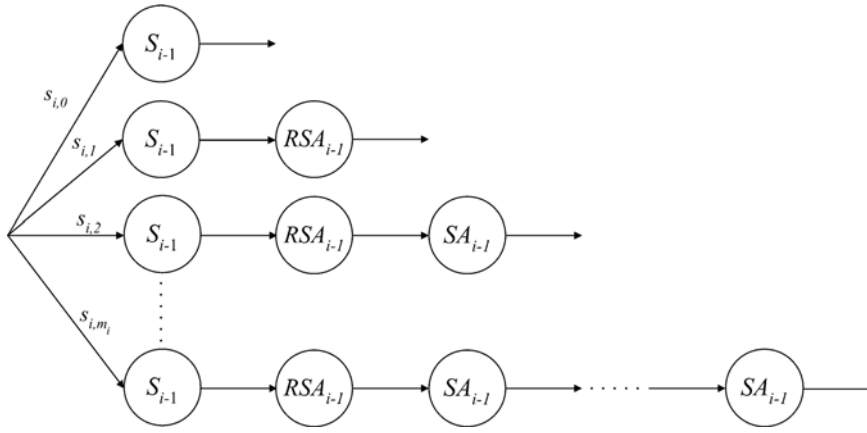


Fig. 3. Representation of the service time A_i of an arrival-server of subsystem L_i

residual inter-departure time and $j-1$ full inter-departure times from the *preceding server-group* M_{i-1}. Figure 3 displays a representation of the service time of an arrival-server of subsystem L_i.

4 Spectral analysis of a subsystem

By fitting Coxian or Erlang distributions on the service times A_i and D_i, subsystem L_i can be modelled as a finite state Markov process; below we describe this Markov process in more detail for a subsystem with m_a arrival servers, m_d departure servers and a buffer of size b.

To reduce the state space we replace the arrival and departure servers by *super servers* with *state-dependent* service times. The service time of the super arrival server is the inter-departure time of the service processes of the non-blocked arrival servers. If the buffer is not full, all arrival servers are working. In this case, the inter-departure time (or super service time) is assumed to be Coxian$_l$ distributed, where phase j ($j = 1, \ldots, l$) has parameter λ_j and p_j is the probability to proceed to the next phase (note that Erlang distributions are a special case of Coxian distributions). If the buffer is full, one or more arrival servers may be blocked. Then the super service time is Coxian distributed, the parameters of which depend on the number of active servers (and follow from the inter-departure time distribution of the active service processes). The service time of the super departure server is defined similarly. In particular, if none of the departure servers is starved, the super service time is the inter-departure time of the service processes of all m_d departure servers. This inter-departure time is assumed to be Coxian$_n$ distributed with parameters μ_j and q_j ($j = 1, \ldots, n$). So, the time spend in phase j is exponentially distributed with parameter μ_j and the probability to proceed to the next phase is q_j.

Now the subsystem can be described by a Markov process with states (i, j, k). The state variable i denotes the total number of customers in the subsystem. Clearly, i is at most equal to $m_d + b + m_a$. Note that, if $i > m_d + b$, then $i - m_d - b$ actually

indicates the number of blocked arrival servers. The state variable j (k) indicates the phase of the service time of the super arrival (departure) server. If $i \leq m_d + b$, then the service time of the super arrival server consists of l phases; the number of phases depends on i for $i > m_d + b$. Similarly, the number of phases of the service time of the super departure server is n for $i \geq m_d$, and it depends on i for $i < m_d$.

The steady-state distribution of this Markov process can be determined efficiently by using the *spectral expansion method*, see e.g. Mitrani [16]. Using the spectral expansion method, Bertsimas [1] analysed a multi-server system with an infinite buffer; we will adapt this method for finite buffer systems. The advantage of the spectral expansion method is that the time to solve a subsystem is *independent* of the size of the buffer.

Below we formulate the equilibrium equations for the equilibrium probabilities $P(i,j,k)$. Only the equations in the states (i,j,k) with $m_d < i < m_d + b$ are presented; the form of the equations in the other states appears to be of minor importance to the analysis.

So, for $m_d < i < m_d + b$ we have:

$$P(i,1,1)(\lambda_1+\mu_1) = \sum_{j=1}^{l}(1-p_j)\lambda_j P(i-1,j,1) + \sum_{k=1}^{n}(1-q_k)\mu_k P(i+1,1,k) \quad (3)$$

$$P(i,j,1)(\lambda_j+\mu_1) = p_{j-1}\lambda_{j-1}P(i,j-1,1) + \sum_{k=1}^{n}(1-q_k)\mu_k P(i+1,j,k), \quad j = 2,\ldots,l \quad (4)$$

$$P(i,1,k)(\lambda_1+\mu_k) = q_{k-1}\mu_{k-1}P(i,1,k-1) + \sum_{j=1}^{l}(1-p_j)\lambda_j P(i-1,j,k), \quad k = 2,\ldots,n \quad (5)$$

$$P(i,j,k)(\lambda_j+\mu_k) = p_{j-1}\lambda_{j-1}P(i,j-1,k) + q_{k-1}\mu_{k-1}P(i,j,k-1), \quad j = 2,\ldots,l, \quad k = 2,\ldots,n. \quad (6)$$

We are going to use the separation of variables technique presented in Mickens [15], by assuming that the equilibrium probabilities $P(i,j,k)$ are of the form

$$P(i,j,k) = D_j R_k w^i, \qquad m_d \leq i \leq m_d + b, \quad 2 \leq j \leq l, \quad 2 \leq k \leq n. \quad (7)$$

Substituting (7) in the equilibrium equations (3)–(6) and dividing by common powers of w yields:

$$D_1R_1(\lambda_1+\mu_1) = \frac{1}{w}\sum_{j=1}^{l}(1-p_j)\lambda_j D_j R_1 + w\sum_{k=1}^{n}(1-q_k)\mu_k D_1 R_k \quad (8)$$

$$D_jR_1(\lambda_j+\mu_1) = p_{j-1}\lambda_{j-1}D_{j-1}R_1 + w\sum_{k=1}^{n}(1-q_k)\mu_k D_j R_k, \quad 2\leq j\leq l \quad (9)$$

$$D_1R_k(\lambda_1+\mu_k) = \frac{1}{w}\sum_{j=1}^{l}(1-p_j)\lambda_j D_j R_k + q_{k-1}\mu_{k-1}D_1R_{k-1}, \quad 2\leq k\leq n \quad (10)$$

$$D_jR_k(\lambda_j+\mu_k) = p_{j-1}\lambda_{j-1}D_{j-1}R_k + q_{k-1}\mu_{k-1}D_jR_{k-1} \quad 2 \leq j \leq l,\ 2 \leq k \leq n \quad (11)$$

We can rewrite (11) as:

$$\frac{\lambda_j D_j - p_{j-1}\lambda_{j-1}D_{j-1}}{D_j} = \frac{-\mu_k R_k + q_{k-1}\mu_{k-1}R_{k-1}}{R_k}, \quad 2 \le j \le l, \quad 2 \le k \le n. \tag{12}$$

Since (12) holds for each combination of j and k, the left-hand side of (12) is independent of k and the right-hand side of (12) is independent of j. Hence, there exists a constant x, depending on w, such that

$$-xD_j = \lambda_j D_j - p_{j-1}\lambda_{j-1}D_{j-1}, \qquad 2 \le j \le l, \tag{13}$$

$$-xR_k = -\mu_k R_k + q_{k-1}\mu_{k-1}R_{k-1}, \qquad 2 \le k \le n. \tag{14}$$

Solving equation (13) gives

$$D_j = D_1 \prod_{r=1}^{l-1} \frac{p_r \lambda_r}{x + \lambda_{r+1}} \tag{15}$$

Substituting (15) in (10) and using equation (14) we find the following relationship between x and w,

$$w = \sum_{j=1}^{l} \frac{(1-p_j)\lambda_j}{x+\lambda_j} \prod_{r=1}^{j-1} \frac{p_r \lambda_r}{x + \lambda_r}. \tag{16}$$

Note that w is equal to the Laplace Stieltjes transform $f_A(s)$ of the service time of the super arrival server, evaluated at $s = x$. Now we do the same for (9) yielding another relationship between x and w,

$$\frac{1}{w} = \sum_{k=1}^{n} \frac{(1-q_k)\mu_k}{-x+\mu_k} \prod_{r=1}^{k-1} \frac{q_r \mu_r}{-x + \mu_r}. \tag{17}$$

Clearly, $1/w$ is equal to the Laplace Stieltjes transform $f_D(s)$ of the service time of the super departure server, evaluated at $s = -x$. Substituting (16) and (17) in (8) and using (13) and (14) we find that

$$1 = f_A(x) f_D(-x).$$

This is a polynomial equation of degree $l+n$; the roots are labeled $x_t, t = 1, \ldots, l+n$, and they are assumed to be distinct. Note that these roots may be complex-valued. Using equation (17) we can find the corresponding $l+n$ values for w_t for $t = 1, \ldots, l+n$. Summarizing, for each t, we obtain the following solution of (3)–(6),

$$P(i,j,k) = B_t \left(\prod_{r=1}^{j-1} \frac{p_r \lambda_r}{x_t + \lambda_{r+1}} \right) \left(\prod_{r=1}^{k-1} \frac{q_r \mu_r}{-x_t + \mu_{r+1}} \right) w_t^i,$$
$$m_b \le i \le m_d + b, \quad 1 \le j \le l, \quad 1 \le k \le n,$$

where $B_t = D_{1,t} R_{1,t}$ is some constant. Since the equilibrium equations are linear, any linear combination of the above solutions satisfies (3)–(6). Hence, the general solution of (3)–(6) is given by

$$P(i,j,k) = \sum_{t=1}^{l+n} B_t \left(\prod_{r=1}^{j-1} \frac{p_r \lambda_r}{x(w_t) + \lambda_{r+1}} \right) \left(\prod_{r=1}^{k-1} \frac{q_r \mu_r}{-x(w_t) + \mu_{r+1}} \right) w_t^i,$$
$$m_b \le i \le m_d + b, \quad 1 \le j \le l, \quad 1 \le k \le n.$$

Finally, the unknown coefficients B_t and the unknown equilibrium probabilities $P(i,j,k)$ for $i < m_d$ and $i > m_d + b$ can be determined from the equilibrium equations for $i \leq m_d$ and $i \geq m_d + b$ and the normalization equation.

5 Iterative algorithm

We now describe the iterative algorithm for approximating the performance characteristics of tandem queue L. The algorithm is based on the decomposition of L in $M-1$ subsystems $L_1, L_2, \ldots, L_{M-1}$. Before going into detail in Section 5.2, we present the outline of the algorithm in Section 5.1.

5.1 Outline of the algorithm

- Step 0: Determine initial characteristics of the service times D_i of the departure servers of subsystem L_i, $i = M-1, \ldots, 1$.
- Step 1: For subsystem L_i, $i = 1, \ldots, M-1$:
  1. Determine the first two moments of the service time A_i of the arrival servers, given the queue-length distribution and throughput of subsystem L_{i-1}.
  2. Determine the queue-length distribution of subsystem L_i.
  3. Determine the throughput T_i of subsystem L_i.
- Step 2: Determine the new characteristics of the service times D_i of the departure servers of subsystem L_i, $i = M-1, \ldots, 1$.
- Repeat Step 1 and 2 until the service time characteristics of the departure servers have converged.

5.2 Details of the algorithm

Step 0: Initialization: The first step of the algorithm is to set $b_{i,j} = 0$ for all i and j. This means that we initially assume that there is no blocking. This also means that the random variables D_i are initially the same as the service times S_i.

Step 1: Evaluation of subsystems: We now know the service time characteristics of the departure servers of L_i, but we also need to know the characteristics of the service times of its arrival servers, before we are able to determine the queue-length distribution of L_i.

(a) Service times of arrival servers

For the first subsystem L_1, the characteristics of A_1 are the same as those of S_0, because the servers of M_0 cannot be starved.

For the other subsystems we proceed as follows. By application of Little's law to the arrival servers, it follows that the throughput of the arrival servers multiplied with the service time of an arrival server is equal to mean number of active (i.e.

non-blocked) arrival servers. The service time of an arrival server of subsystem i is equal to $1/\mu_{a,i}$ and the mean number of active servers is equal to

$$\left(1 - \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}\right) m_{i-1} + \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}(m_{i-1} - j).$$

So, we have for the throughput T_i of subsystem L_i,

$$T_i = \left(1 - \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}\right) m_{i-1}\mu_{a,i} + \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}(m_{i-1} - j)\mu_{a,i}, \quad (18)$$

where $p_{i,j}$ denotes the probability of j customers in subsystem L_i. By substituting the estimate $T_{i-1}^{(n)}$ for T_i and $p_{i,n_i+j}^{(n-1)}$ for p_{i,n_i+j} we get as new estimate for the service rate $\mu_{a,i}$,

$$\mu_{a,i}^{(n)} = \frac{T_{i-1}^{(n)}}{(1 - \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}^{(n-1)})m_{i-1} + \sum_{j=1}^{m_{i-1}} p_{i,m_i+b_i+j}^{(n-1)}(m_{i-1} - j)},$$

where the super scripts indicate in which iteration the quantities have been calculated.

To approximate the coefficient of variation $c_{a,i}$ of A_i we use the representation for A_i as described in Section 3 (which is based on $s_{i-1,j}$, S_{i-1}, RSA_{i-1} and SA_{i-1}).

(b) Analysis of subsystem L_i

Based on the (new) characteristics of the service times of both arrival and departure servers we can determine the steady-state queue-length distribution of subsystem L_i. To do so we first fit Coxian$_2$ or Erlang$_{k-1,k}$ distributions on the first two moments of the service times of the arrival-servers and departure-servers as described in Section 3. Then we calculate the equilibrium probabilities $p_{i,j}$ by using the spectral expansion method as described in Section 4.

(c) Throughput of subsystem L_i

Once the steady-state queue length distribution is known, we can determine the new throughput $T_i^{(n)}$ according to (cf. (18))

$$T_i^{(n)} = \left(1 - \sum_{j=0}^{m_i-1} p_{i,j}^{(n)}\right) m_i\mu_{d,i}^{(n-1)} + \sum_{j=1}^{m_i-1} p_{i,j}^{(n)} j\mu_{d,i}^{(n-1)}. \quad (19)$$

We also determine new estimates for the probabilities $b_{i-1,j}$ that j servers of server-group M_{i-1} are blocked after service completion of a server in server-group M_{i-1} and the probabilities $s_{i,j}$ that j servers of server-group M_i are starved after service completion of a server in server-group M_i.

We perform Step 1 for every subsystem from L_1 up to L_{M-1}.

Step 2: Service times of departure servers: Now we have new information about the departure processes of the subsystems. So we can again calculate the first two moments of the service times of the departure-servers, starting from D_{M-2} down to D_1. Note that D_{M-1} is always the same as S_{M-1}, because the servers in server-group M_{M-1} can never be blocked.

A new estimate for the rate $\mu_{d,i}$ of D_i is determined from (cf. (18))

$$\mu_{d,i}^{(n)} = \frac{T_{i+1}^{(n)}}{(1 - \sum_{j=0}^{m_i-1} p_{i,j}^{(n)})m_i + \sum_{j=1}^{m_i-1} p_{i,j}^{(n)} j} \tag{20}$$

The calculation of a new estimate for the coefficient of variation $c_{d,i}$ of D_i is similar to the one of A_i.

Convergence criterion: After Step 1 and 2 we check whether the iterative algorithm has converged by comparing the departure rates in the $(n-1)$-th and k-th iteration. We decide to stop when the sum of the absolute values of the differences between these rates is less than ε; otherwise we repeat Step 1 and 2. So the convergence criterion is

$$\sum_{i=1}^{M-1} \left| \mu_{d,i}^{(n)} - \mu_{d,i}^{(n-1)} \right| < \varepsilon.$$

Of course, we may use other stop-criteria as well; for example, we may consider the throughput instead of the departure rates. The bottom line is that we go on until all parameters do not change anymore.

Remark. Equality of throughputs.

It is easily seen that, after convergence, the throughputs in all subsystems are equal. Let us assume that the iterative algorithm has converged, so $\mu_{d,i}^{(n)} = \mu_{d,i}^{(n-1)}$ for all $i = 1, \ldots, M-1$. From equations (19) and (20) we find the following:

$$\begin{aligned} T_i^{(n)} &= \left(1 - \sum_{j=0}^{m_i-1} p_{i,j}^{(n)}\right) m_i \mu_{d,i}^{(n-1)} + \sum_{j=1}^{m_i-1} p_{i,j}^{(n)} j \mu_{d,i}^{(n-1)} \\ &= \left(1 - \sum_{j=0}^{m_i-1} p_{i,j}^{(n)}\right) m_i \mu_{d,i}^{(n)} + \sum_{j=1}^{m_i-1} p_{i,j}^{(n)} j \mu_{d,i}^{(n)} \\ &= T_{i+1}^{(n)}. \end{aligned}$$

Hence we can conclude that the throughputs in all subsystems are the same after convergence.

Complexity analysis: The complexity of this method is as follows. Within the iterative algorithm, solving a subsystem consumes most of the time. In one iteration a subsystem is solved M times. The number of iterations needed is difficult to predict, but in practice this number is about three to seven iterations.

The time consuming part of solving a subsystem is solving the boundary equations. This can be done in $O((m_a + m_d)(k_a k_d)^3)$ time, where k_a is the number

of phases of the distribution of one arrival process and k_d is the number of phases of the distribution of one departure process. Then, the time complexity of one iteration becomes $O(M \max_i((m_i + m_{i-1})(k_i k_{i-1})^3))$. This means that the time complexity is polynomial and it doesn't depend on the sizes of the buffers.

6 Numerical results

In this section we present some numerical results. To investigate the quality of our method we compare it with discrete event simulation. After that, we compare our method with the method developed by Tahilramani et al. [21], which is implemented in QNAT [25].

6.1 Comparison with simulation

In order to investigate the quality of our method we compare the throughput and the mean sojourn time with the ones produced by discrete event simulation. We are especially interested in investigating for which set of input-parameters our method gives satisfying results. Each simulation run is sufficiently long such that the widths of the 95% confidence intervals of the throughput and the mean sojourn time are smaller than 1%.

In order to test the quality of the method we use a broad set of parameters. We test two different lengths M of tandem queues, namely with 4 and 8 server-groups. For each tandem queue we vary the number of servers m_i in the server-groups; we use tandems with 1 server per server-group, 5 servers per server-group and with the sequence $(4, 1, 2, 8)$. We also vary the level of balance in the tandem queue; every server-group has a maximum total rate of 1 and the group right after the middle can have a total rate of 1, 1.1, 1.2, 1.5 and 2. The coefficient of variation of the service times varies between 0.1, 0.2, 0.5, 1, 1.5 and 2. Finally we vary the buffer sizes between 0, 2, 5 and 10. This leads to a total of 720 test-cases. The results for each category are summarized in Table 1 up to 5. Each table lists the average error in the throughput and the mean sojourn time compared with the simulation results. Each table also gives for 4 error-ranges the percentage of the cases which fall in that range. The results for a selection of 54 cases can be found in Tables 6 and 7.

Table 1. Overall results for tandem queues with different buffer sizes

Buffer	Error in throughput					Error in mean sojourn time				
sizes (b_i)	Avg.	0–5%	5–10%	10–15%	>15%	Avg.	0–5%	5–10%	10–15%	>15%
0	5.7%	55.0%	35.0%	4.4%	5.6%	6.8%	42.8%	35.0%	14.4%	7.8%
2	3.2%	76.1%	22.8%	1.1%	0.0%	4.7%	57.2%	35.0%	7.2%	0.6%
5	2.1%	90.6%	9.4%	0.0%	0.0%	4.5%	60.6%	32.2%	7.2%	0.0%
10	1.4%	95.6%	4.4%	0.0%	0.0%	5.1%	53.3%	34.4%	12.2%	0.0%

Table 2. Overall results for tandem queues with different balancing rates

Rates	Error in throughput					Error in mean sojourn time				
unbalanced server-group ($m_i\mu_{p,i}$)	Avg.	0–5%	5–10%	10–15%	>15%	Avg.	0–5%	5–10%	10–15%	>15%
1.0	3.3%	76.4%	20.8%	1.4%	1.4%	3.4%	74.3%	22.2%	2.1%	1.4%
1.1	3.1%	78.5%	18.1%	2.1%	1.4%	4.0%	68.1%	27.1%	3.5%	1.4%
1.2	3.0%	79.2%	18.8%	0.7%	1.4%	4.6%	59.7%	34.7%	4.2%	1.4%
1.5	3.0%	81.3%	16.0%	1.4%	1.4%	6.5%	38.2%	43.1%	16.7%	2.1%
2.0	3.1%	81.3%	16.0%	1.4%	1.4%	7.9%	27.1%	43.8%	25.0%	4.2%

Table 3. Overall results for tandem queues with different coefficients of variation of the service times

Coefficients	Error in throughput					Error in mean sojourn time				
of variation ($c^2_{p,i}$)	Avg.	0–5%	5–10%	10–15%	>15%	Avg.	0–5%	5–10%	10–15%	>15%
0.1	4.4%	54.2%	44.2%	1.7%	0.0%	3.1%	77.5%	21.7%	0.8%	0.0%
0.2	2.6%	88.3%	11.7%	0.0%	0.0%	3.4%	75.8%	22.5%	1.7%	0.0%
0.5	2.2%	90.8%	9.2%	0.0%	0.0%	4.5%	60.8%	32.5%	6.7%	0.0%
1.0	1.5%	93.3%	2.5%	4.2%	0.0%	4.1%	64.2%	30.0%	5.0%	0.8%
1.5	3.0%	82.5%	13.3%	0.0%	4.2%	7.5%	25.8%	54.2%	15.0%	5.0%
2.0	4.8%	66.7%	26.7%	2.5%	4.2%	9.1%	16.7%	44.2%	32.5%	6.7%

Table 4. Overall results for tandem queues with a different number of servers per server-group

Number of	Error in throughput					Error in mean sojourn time				
servers (m_i)	Avg.	0-5%	5–10%	10–15%	>15%	Avg.	0–5%	5–10%	10–15%	>15%
All 1	2.9%	83.8%	9.2%	2.9%	4.2%	5.9%	46.3%	39.2%	10.0%	4.6%
All 5	3.8%	68.3%	30.8%	0.8%	0.0%	4.6%	60.0%	29.2%	10.8%	0.0%
Mixed	2.6%	85.8%	13.8%	0.4%	0.0%	5.3%	54.2%	34.2%	10.0%	1.7%

We may conclude the following from the above results. First, we see in Table 1 that the performance of the approximation becomes better when the buffer sizes increase. This may be due to less dependencies between the servers-groups when the buffers are large.

We also notice that the performance is better for balanced lines (Table 2); for unbalanced lines, especially the estimate for the mean sojourn time is not as good as for balanced lines. If we look at the coefficients of variation of the service times (Table 3), we get the best approximations for the throughput when the coefficients

Table 5. Overall results for tandem queues with 4 and 8 server-groups

Number of server-groups (M)	Error in throughput					Error in mean sojourn time				
	Avg.	0–5%	5–10%	10–15%	>15%	Avg.	0–5%	5–10%	10–15%	>15%
4	2.3%	87.2%	12.2%	0.6%	0.0%	4.7%	57.5%	32.8%	9.7%	0.0%
8	3.9%	71.4%	23.6%	2.2%	2.8%	5.8%	49.4%	35.6%	10.8%	4.2%

Table 6. Detailed results for balanced tandem queues

m_i	M	$c^2_{p,i}$	Buffers	T App.	T Sim.	Diff.	S App.	S Sim.	Diff.
1	4	0.1	0	0.735	0.771	−4.7%	4.70	4.63	1.5%
	8		2	0.906	0.926	−2.2%	16.14	15.99	0.9%
	4		10	0.981	0.985	−0.4%	19.22	19.03	1.0%
	8	1.0	0	0.488	0.443	10.2%	11.73	13.43	−12.7%
	4		2	0.703	0.700	0.4%	9.09	9.25	−1.7%
	8		10	0.855	0.855	0.0%	49.52	49.81	−0.6%
	4	1.5	0	0.504	0.473	6.6%	5.82	6.27	−7.2%
	8		2	0.607	0.581	4.5%	21.94	23.52	−6.7%
	4		10	0.834	0.835	−0.1%	22.38	22.31	0.3%
5	4	0.1	0	0.789	0.856	−7.8%	22.48	21.78	3.2%
	8		2	0.827	0.926	−10.7%	52.35	49.71	5.3%
	4		10	0.927	0.983	−5.7%	36.88	35.24	4.7%
	8	1.0	0	0.693	0.697	−0.6%	49.20	49.14	0.1%
	4		2	0.797	0.808	−1.4%	26.37	26.17	0.8%
	8		10	0.867	0.882	−1.7%	83.09	83.96	−1.0%
	4	1.5	0	0.742	0.724	2.5%	22.99	23.90	−3.8%
	8		2	0.759	0.737	3.0%	54.63	57.27	−4.6%
	4		10	0.867	0.874	−0.8%	37.97	38.86	−2.3%
Mixed	4	0.1	0	0.746	0.793	−5.9%	16.19	16.28	−0.6%
	8		2	0.845	0.921	−8.3%	39.90	38.96	2.4%
	4		10	0.956	0.984	−2.8%	31.61	30.05	5.2%
	8	1.0	0	0.619	0.604	2.5%	37.90	38.55	−1.7%
	4		2	0.756	0.757	−0.1%	20.15	20.14	0.0%
	8		10	0.863	0.871	−0.9%	71.67	71.74	−0.1%
	4	1.5	0	0.633	0.619	2.3%	16.78	18.01	−6.8%
	8		2	0.705	0.678	4.0%	43.38	46.32	−6.3%
	4		10	0.850	0.856	−0.7%	31.43	32.37	−2.9%

Table 7. Detailed results for unbalanced tandem queues

m_i	M	$c^2_{p,i}$	Buffers	T App.	T Sim.	Diff.	S App.	S Sim.	Diff.
1	8	0.1	0	0.718	0.751	−4.4%	8.90	9.27	−4.0%
	4		2	0.960	0.958	0.2%	6.18	6.41	−3.6%
	8		10	0.980	0.983	−0.3%	38.45	43.22	−11.0%
	4	1.0	0	0.594	0.561	5.9%	4.84	5.28	−8.3%
	8		2	0.690	0.670	3.0%	18.81	20.31	−7.4%
	4		10	0.918	0.912	0.7%	16.20	17.41	−7.0%
	8	1.5	0	0.482	0.409	17.8%	11.26	13.79	−18.3%
	4		2	0.714	0.691	3.3%	8.03	8.60	−6.6%
	8		10	0.830	0.819	1.3%	46.75	50.16	−6.8%
5	8	0.1	0	0.781	0.851	−8.2%	43.03	42.65	0.9%
	4		2	0.902	0.958	−5.8%	21.63	21.50	0.6%
	8		10	0.922	0.983	−6.2%	71.89	73.95	−2.8%
	4	1.0	0	0.801	0.794	0.9%	20.79	21.13	−1.6%
	8		2	0.789	0.787	0.3%	51.52	53.49	−3.7%
	4		10	0.927	0.929	−0.2%	30.37	32.61	−6.9%
	8	1.5	0	0.730	0.692	5.5%	44.43	47.95	−7.3%
	4		2	0.850	0.828	2.7%	21.95	23.70	−7.4%
	8		10	0.864	0.862	0.2%	74.69	81.01	−7.8%
Mixed	8	0.1	0	0.744	0.790	−5.8%	30.96	32.41	−4.5%
	4	0.1	2	0.920	0.953	−3.5%	16.72	17.14	−2.5%
	8	0.1	10	0.945	0.983	−3.9%	61.00	62.54	−2.5%
	4	1.0	0	0.714	0.702	1.7%	16.22	16.43	−1.3%
	8	1.0	2	0.750	0.742	1.1%	39.64	42.20	−6.1%
	4	1.0	10	0.926	0.919	0.8%	25.99	27.60	−5.8%
	8	1.5	0	0.628	0.588	6.8%	32.68	37.66	−13.2%
	4	1.5	2	0.787	0.773	1.8%	17.52	18.93	−7.4%
	8	1.5	10	0.844	0.843	0.1%	61.82	69.32	−10.8%

of variation are 1, and also the estimate for the mean sojourn time is better for small coefficients of variation.

The quality of the results seems to be rather insensitive to the number of servers per server-group (Table 4), in spite of the super-server approximation used for multi-server models. Finally we may conclude from Table 5 that the results are better for shorter tandem queues.

Most crucial to the quality of the approximation of the throughput appears to be the buffer-size. For the sojourn time this appears to be the coefficient of variation of the service time. In Figures 4 and 5 we present a scatter-plot of simulation results versus approximation results for the throughput and mean sojourn times; the plotted cases are the same as in Tables 6 and 7. The results of the throughput are split-up

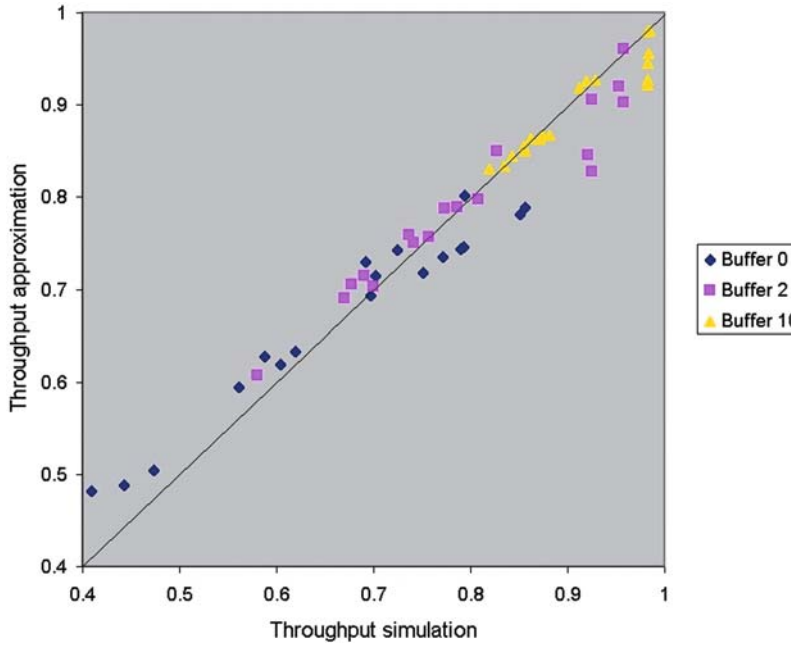


Fig. 4. Scatter-plot of the throughput of 54 cases split up by buffer-size

according to the buffer-size; the one for the sojourn times are split-up according to the squared coefficient of variation of the service times.

Overall we can say that the approximation produces accurate results in most cases. In the majority of the cases the error of the throughput is within 5% of the simulation and the error of the mean sojourn time is within 10% of the simulation (see also Tables 6 and 7). The worst performance is obtained for unbalanced lines with zero buffers and high coefficients of variation of the service times. But these cases are unlikely (and undesired) to occur in practice.

The computation times are very short. On a modern computer the computation times are much less than a second in most cases, only in cases with service times with low coefficients of variation and 1 server per server-group the computation times increase to a few seconds. Therefore, for the design of production lines, this is a very useful approximation method.

6.2 Comparison with QNAT

We also compare the present method with QNAT, a method developed by Tahilramani et al. [21]. We use a tandem queue with four server-groups. It was only possible to test cases where the first server-group consists of a single exponential server. The reason is that the two methods assume a different arrival process to the system. Both processes, however, coincide for the special case of a single exponential server at the beginning of the line. We varied the number of servers per server-group and the size of buffers. Table 8 shows the results.

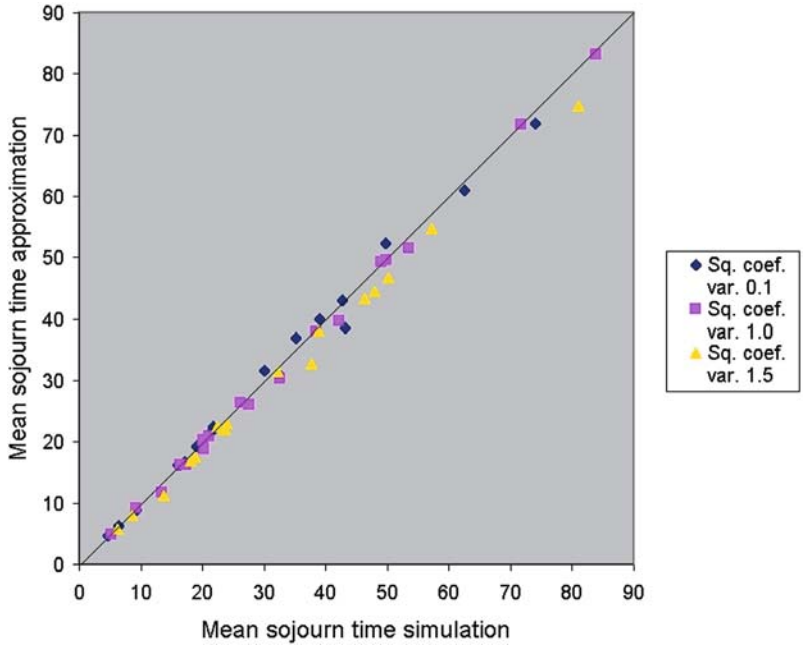


Fig. 5. Scatter-plot of the mean sojourn time of 54 cases split up by coefficient of variation

Table 8. Comparison of our method with QNAT

m_i	b_i	TP Sim.	TP App.	Our error	TP QNAT	QNAT Error	Soj. Sim.	Soj. App.	Our error	Soj. QNAT	QNAT error
(1,1,1,1)	0	0.515	0.537	−4.3%	0.500	2.9%	5.95	5.61	5.7%	–	–
(1,1,1,1)	2	0.702	0.703	−0.1%	0.750	−6.8%	9.25	9.10	1.7%	8.17	11.7%
(1,1,1,1)	10	0.879	0.876	0.3%	0.917	−4.3%	21.43	21.41	0.1%	18.55	13.5%
(1,5,5,5)	0	0.711	0.717	−0.8%	0.167	76.5%	17.87	17.67	1.1%	–	–
(1,5,5,5)	2	0.791	0.788	0.3%	0.800	−1.1%	20.53	20.45	0.4%	–	–
(1,5,5,5)	10	0.898	0.884	1.6%	0.895	0.3%	32.27	32.59	−1.0%	22.88	29.1%
(1,4,2,8)	0	0.677	0.692	−2.3%	0.200	70.5%	16.59	16.28	1.9%	–	–
(1,4,2,8)	2	0.775	0.774	0.1%	0.800	−3.2%	19.29	19.15	0.7%	–	–
(1,4,2,8)	10	0.893	0.886	0.8%	0.902	−1.0%	31.03	30.86	0.6%	23.04	25.7 %

We see that the present approximation method is much more stable than QNAT and gives in almost all cases better results. Especially the approximation of the mean sojourn time is much better; in a number of cases QNAT is not able to produce an approximation of the mean sojourn time. Of course, one should be careful with drawing conclusions from this limited set of cases. Table 8 only gives an indication of how the two methods perform.

6.3 Industrial case

To give an indication of the performance of our method in practice, we present the results of an industrial case. The case involves a production line for the production of light bulbs. The production line consists of 5 production stages with buffers in between. Each stage has a different number of machines varying between 2 and 8. The machines have deterministic service times, but they do suffer from breakdowns. In the queueing model we included the breakdowns into the coefficient of variation of the service times, yielding effective service times with coefficients of variation larger than 0. In Table 9 the parameters of the production line are shown.

Table 9. Parameters for the production line for the production of bulbs

Stage	m_i	$\mu_{p,i}$	$c_{p,i}^2$	b_i
1	2	5.73	0.96	–
2	8	1.53	0.09	21
3	4	3.43	0.80	11
4	1	32.18	0.57	34
5	4	16.12	0.96	19

We only have data of the throughput and not of the mean sojourn time of the line, so we can only test the approximation for the throughput. The output of the production line based on the measured data is 11.34 products per time unit. If we simulate this production line, we obtain a throughput of 11.41 products per time unit. The throughput given by our approximation method is 11.26, so in this case the approximation is a good prediction for the actual throughput.

7 Concluding remarks

In this paper we described a method for the approximate analysis of a multi-server tandem queue with finite buffers and general service times. We decomposed the tandem queue in subsystems. We used an iterative algorithm to approximate the arrivals and departures at the subsystems and to approximate some performance characteristics of the tandem queue. Each multi-server subsystem is approximated by a single (super) server queue with state-dependent inter-arrival and service times, the steady-state queue length distribution of which is determined by a spectral expansion method.

This method is robust and efficient; it provides a good and fast alternative to simulation methods. In most cases the errors for performance characteristics as the throughput and mean sojourn time are within 5% of the simulation results. Numerical results also give an indication of the performance of the method compared with QNAT. The method can be extended in several directions. One may think of more

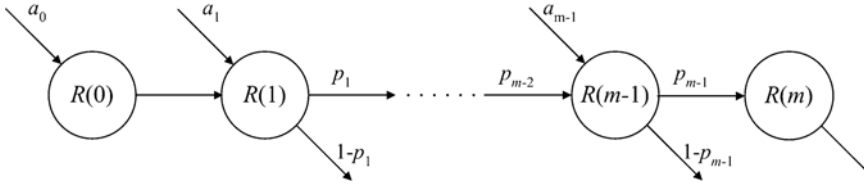


Fig. 6. Phase diagram of an arbitrary inter-departure time

general configurations, like splitting and merging of streams or the possibility of feedback. Other possibilities for extension are for example unreliable machines and assembly/disassembly (see [24]). Possibilities for improving the quality of the approximation are, for example, using a more detailed description of the arrival to and departures from the subsystems (e.g. including correlations between consecutive arrivals and departures) or improving the subsystem analysis by using a description of the service process that is more detailed than the super-server approach.

Appendix: Superposition of service processes

Let us consider m independent service processes, each of them continuously servicing customers one at a time. The service times are assumed to be independent and identically distributed. We are interested in the first two moments of an arbitrary inter-departure time of the superposition of m service processes. Below we distinguish between Coxian$_2$ service times and Erlang$_{k-1,k}$ service times.

A.1 Coxian$_2$ service times

We assume that the service times of each service process are Coxian$_2$ distributed with the same parameters. The rate of the first phase is μ_1, the rate of the second phase is μ_2 and the probability that the second phase is needed is q. The distribution of an arbitrary inter-departure time of the superposition of m service processes can be described by a phase-type distribution with $m+1$ phases, numbered $0, 1, \ldots, m$. In phase i exactly i service processes are in the second phase of the service time and $m - i$ service processes are in the first phase. A phase diagram of the phase-type distribution of an arbitrary inter-departure time is shown in Figure 6. The probability to start in phase i is denoted by a_i, $i = 0, \ldots, m - 1$. The sojourn time in phase i is exponentially distributed with rate $R(i)$, and p_i is the probability to continue with phase $i + 1$ after completion of phase i. Now we explain how to compute the parameters a_i, $R(i)$ and p_i.

The probability a_i can be interpreted as follows. It is the probability that i service processes are in phase 2 just after a departure (i.e., service completion). There is at least one process in phase 1, namely the one that generated the departure. Since the service processes are mutually independent, the number of service processes in phase 2 is binomially distributed with $m - 1$ trials and success probability p.

The success probability is equal to the fraction of time a single service process is in phase 2, so

$$p = \frac{q\mu_1}{q\mu_+\mu_2}.$$

Hence, for the initial probability a_i we get

$$a_i = \binom{m-1}{i} \left(\frac{q\mu_1}{q\mu_1 + \mu_2}\right)^i \left(\frac{\mu_2}{q\mu_1 + \mu_2}\right)^{m-1-i} \tag{21}$$

To determine the rate $R(i)$, note that in state i there are i processes in phase 2 and $m - i$ in phase 1, so the total rate at which one of the service processes completes a service phase is equal to

$$R(i) = (m-i)\mu_1 + i\mu_2 \tag{22}$$

It remains to find p_i, the probability that there is no departure after phase i. In phase i three things may happen:

- Case (i): A service process completes phase 1 and immediately continues with phase 2;
- Case (ii): A service process completes phase 1 and generates a departure;
- Case (iii): A service process completes phase 2 (and thus always generates a departure).

Clearly, p_i is the probability that case (i) happens, so

$$p_i = \frac{q(m-i)\mu_i}{R(i)} \tag{23}$$

Now the parameters of the phase-type distribution are known, we can determine its first two moments. Let X_i denote the total sojourn time, given that we start in phase i, $i = 0, 1, \ldots, m$. Starting with

$$EX_m = \frac{1}{R(m)}, \qquad EX_m^2 = \frac{2}{R(m)^2},$$

the first two moments of X_i can be calculated from $i = m - 1$ down to $i = 0$ by using

$$EX_i = \frac{1}{R(i)} + p_i EX_i, \tag{24}$$

$$EX_i^2 = \frac{2}{R(i)^2} + p_i \left(\frac{2EX_{i+1}}{R(i)} + EX_{i+1}^2\right). \tag{25}$$

Then the rate μ_s and coefficient of variation c_s of an arbitrary inter-departure time of the superposition of m service processes follow from

$$\mu_s^{-1} = \sum_{i=0}^{m} a_i EX_i \;=\; \frac{1}{m}\left(\frac{1}{\mu_1} + \frac{q}{\mu_2}\right), \tag{26}$$

$$c_s^2 = \mu_s^2 \left(\sum_{i=0}^{m} a_i EX_i^2\right) - 1 \tag{27}$$

A.2 Erlang$_{k-1,k}$ service times

Now the service times of each service process are assumed to be Erlang$_{k-1,k}$ distributed, i.e., with probability p (respectively $1-p$) a service time consists of $k-1$ (respectively k) exponential phases with parameter μ. Clearly, the time that elapses until one of the m service processes completes a service phase is exponential with parameter $m\mu$. The number of service phases completions before one of the service processes generates a departure ranges from 1 up to $m(k-1)+1$. So the distribution of an arbitrary inter-departure time of the superposition of m service processes is a mixture of Erlang distributions; with probability p_i it consists of i exponential phases with parameter $m\mu$, $i = 1, \ldots, m(k-1)+1$. Figure 7 depicts the phase diagram. Below we show how to determine the probabilities p_i.

An arbitrary inter-departure time of the superposition of m service processes is the minimum of $m-1$ equilibrium residual service times and one full service time. Both residual and full service time have a (different) mixed Erlang distribution. In particular, the residual service consists with probability r_i of i phases with parameter μ, where

$$r_i = \begin{cases} 1/(k-p), & i = 1, 2, \ldots, k-1; \\ (1-p)/(k-p), & i = k. \end{cases}$$

The minimum of two mixed Erlang service times has again a mixed Erlang distribution; below we indicate how the parameters of the distribution of the minimum can be determined. Then repeated application of this procedure yields the minimum of m mixed Erlang service times.

Let X_1 and X_2 be two independent random variables with mixed Erlang distributions, i.e., with probability $q_{k,i}$ the random variable X_k ($k = 1, 2$) consists of i exponential phases with parameter μ_k, $i = 1, \ldots, n_k$. Then the minimum of X_1

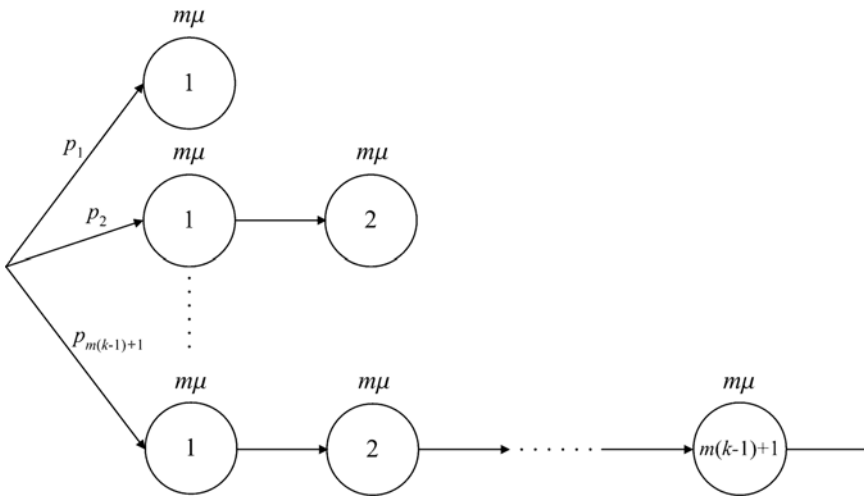


Fig. 7. Phase diagram of an arbitrary inder-departure time

and X_2 consists of at most n_1+n_2-1 exponential phases with parameter $\mu_1+\mu_2$. To find the probability q_i that the minimum consists of i phases, we proceed as follows. Define $q_i(j)$ as the probability that the minimum of X_1 and X_2 consists of i phases transitions, where $j(\leq i)$ transitions are due to X_1 and $i-j$ transitions are due to X_2. Obviously we have

$$q_i = \sum_{j=\max(0,i-n_2)}^{\min(i,n_1)} q_i(j), \qquad i = 1,2,\ldots,n_1+n_2-1.$$

To determine $q_i(j)$ note that the ith phase transition of the minimum can be due to either X_1 or X_2. If X_1 makes the last transition, then X_1 clearly consists of exactly j phases and X_2 of at least $i-j+1$ phases; the probability that X_2 makes $i-j$ transitions before the jth transition of X_1 is negative-binomially distributed with parameters j and $\mu_1/(\mu_1+\mu_2)$. The result is similar if X_2 instead of X_1 makes the last transition. Hence, we obtain

$$\begin{aligned} q_i(j) = {} & \binom{i-1}{j-1}\left(\frac{\mu_1}{\mu_1+\mu_2}\right)^j\left(\frac{\mu_2}{\mu_1+\mu_2}\right)^{i-j} q_{1,j}\left(\sum_{k=i-j+1}^{n_2} q_{2,k}\right) \\ & + \binom{i-1}{j}\left(\frac{\mu_1}{\mu_1+\mu_2}\right)^j\left(\frac{\mu_2}{\mu_1+\mu_2}\right)^{i-j}\left(\sum_{k=j+1}^{n_1} q_{1,k}\right) q_{2,i-j}, \\ & 1 \leq i \leq n_1+n_2-1, \quad 0 \leq j \leq i, \end{aligned}$$

where by convention, $q_{1,0} = q_{2,0} = 0$.

By repeated application of the above procedure we can find the probability p_i that the distribution of an arbitrary inter-departure time of the superposition of m Erlang$_{k-1,k}$ service processes consists of exactly i service phases with parameter $m\mu$, $i = 1,2,\ldots,m(k-1)+1$. It is now easy to determine the rate μ_s and coefficient of variation c_s of an arbitrary inter-departure time, yielding

$$\mu_s^{-1} = \frac{1}{m}\left(\frac{p(k-1)}{\mu} + \frac{(1-p)k}{\mu}\right) = \frac{k-p}{m\mu},$$

and, by using that the second moment of an E_k distribution with scale parameter μ is $k(k+1)/\mu^2$,

$$c_s^2 = \mu_s^2 \sum_{i=1}^{m(k-1)+1} p_i \frac{i(i+1)}{(m\mu)^2} - 1 = -1 + \frac{1}{(k-p)^2} \sum_{i=1}^{m(k-1)+1} p_i i(i+1).$$

A.3 Equilibrium residual inter-departure time

To determine the first two moments of the equilibrium residual inter-departure time of the superposition of m independent service processes we adopt the following simple approach.

Let the random variable D denote an arbitrary inter-departure time and let R denote the equilibrium residual inter-departure time. It is well known that

$$E(R) = \frac{E(D^2)}{2E(D)}, \qquad E(R^2) = \frac{E(D^3)}{3E(D)}.$$

In the previous sections we have shown how the first two moments of D can be determined in case of Coxian$_2$ and Erlang$_{k-1,k}$ service times. Its third moment is approximated by the third moment of the distribution fitted on the first two moments of D, according to the recipe in Section 3.

References

1. Bertsimas D (1990) An analytic approach to a general class of G/G/s queueing systems. Operations Rearch earch 1: 139–155
2. Buzacott JA (1967) Automatic transfer lines with buffer stock. International Journal of Production Research 5: 183–200
3. Cruz FRB, MacGregor Smith J (2004) Algorithm for analysis of generalized $M/G/C/C$ state dependent queueing networks. http://www.compscipreprints.com/comp/Preprint/fcruzfcruz/20040105/1
4. Cruz FRB, MacGregor Smith J, Queiroz DC (2004) Service and capacity allocation in $M/G/C/C$ state dependent queueing networks. Computers & Operations Research (to appear)
5. Dallery Y, David R, Xie X (1989) Approximate analysis of transfer lines with unreliable machines and finite buffers. IEEE Transactiona on Automatic Control 34(9): 943–953
6. Dallery Y, Gershwin B (1992) Manufacturing flow line systems: a review of models and analytical results. Queueing Systems 12: 3–94
7. Gershwin SB, Burman MH (2000) A decomposition method for analyzing inhomogeneous assembly/disassembly systems. Annals of Operation Research 93: 91–115
8. Hillier FS, So KC (1995) On the optimal design of tandem queueing systems with finite buffers. Queueing Systems Theory Application 21: 245–266
9. Jain S, MacGregor Smith J (1994) Open finite queueing networks with $M/M/C/K$ parallel servers. Computers Operations Research 21(3): 297–317
10. Johnson MA (1993) An empirical study of queueing approximations based on phase-type distributions. Communication Statistic-Stochastic Models 9(4): 531–561
11. Kerbache L, MacGregor Smith J (1987) The generalized expansion method for open finite queueing networks. The European Journal of Operations Research 32: 448–461
12. Kouvatsos D, Xenios NP (1989) MEM for arbitrary queueing networks with multiple general servers and repetative-service blocking. Performance Evaluation 10: 169–195
13. Li Y, Cai X, Tu F, Shao X (2004) Optimization of tandem queue systems with finite buffers. Computers & Operations Research 31: 963–984
14. Marie RA (1980) Calculating equilibrium probabilities for $\lambda(n)/C_k/1/N$ queue. Proceedings Performance '80, Toronto, pp 117–125
15. Mickens R (1987) Difference equations. Van Nostrand-Reinhold, New York
16. Mitrani I, Mitra D (1992) A spectral expansion method for random walks on semi-infinite strips. In: Beauwens R, de Groen P (eds) Iterative methods in linear algebra, pp 141–149. North-Holland, Amsterdam
17. Perros HG (1989) A bibliography of papers on queueing networks with finite capacity queues. Perf Eval 10: 255–260
18. Perros HG (1994) Queueing networks with blocking. Oxford University Press, Oxford
19. Perros HG, Altiok T (1989) Queueing networks with blocking. North-Holland, Amsterdam
20. MacGregor Smith J, Cruz FRB (2000) The buffer allocation problem for general finite buffer queueing networks. http://citeseer.nj.nec.com/smith00buffer.html
21. Tahilramani H, Manjunath D, Bose SK (1999) Approximate analysis of open network of $GE/GE/m/N$ queues with transfer blocking. MASCOTS'99, pp 164–172

22. Tijms HC (1994) Stochastic models: an algorithmic approach. Wiley, Chichester
23. Tolio T, Matta A, Gershwin SB (2002) Analysis of two-machine lines with multiple failure modes. IIE Transactions 34: 51–62
24. van Vuuren M (2003) Performance analysis of multi-server tandem queues with finite buffers. Master's Thesis, University of Technology Eindhoven, The Netherlands
25. http://poisson.ecse.rpi.edu/ hema/qnat/

An analytical method for the performance evaluation of echelon kanban control systems

Stelios Koukoumialos and George Liberopoulos

Department of Mechanical and Industrial Engineering, University of Thessaly, Volos, Greece
(e-mail: skoukoum@mie.uth.gr; glib@mie.uth.gr)

Abstract. We develop a general purpose analytical approximation method for the performance evaluation of a multi-stage, serial, echelon kanban control system. The basic principle of the method is to decompose the original system into a set of nested subsystems, each subsystem being associated with a particular echelon of stages. Each subsystem is analyzed in isolation using a product-form approximation technique. An iterative procedure is used to determine the unknown parameters of each subsystem. Numerical results show that the method is fairly accurate.

Keywords: Production/inventory control – Multi-stage system – Echelon kanban – Performance evaluation

1 Introduction

In 1960, Clark and Scarf [10] initiated the research on the coordination of multi-stage, serial, uncapacitated inventory systems with stochastic demand and constant lead times. Their work received considerable attention in the years that followed and spawned a large amount of follow-on research. Much of that research evolved around variants of the base stock control system. Research on the coordination of multi-stage, serial, production/inventory systems having networks of stations with limited capacity, on the other hand, has been directed mostly towards variants of the kanban control system. In this paper, we develop an analytical approximation method for the performance evaluation of an echelon kanban control system, used for the coordination of production in a multi-stage, serial production/inventory system. We test the behavior of this method with several numerical examples.

The term "echelon kanban" was introduced in [19]. The basic principle of the operation of the echelon kanban control system is very simple: When a part leaves

Correspondence to: G. Liberopoulos

the last stage of the system to satisfy a customer demand, a new part is demanded and authorized to be released into each stage. It is worth noting that the echelon kanban control system is equivalent to the integral control system described in [8]. The echelon kanban control system differs from the conventional kanban control system, which is referred to as installation kanban control system or policy in [19], in that in the conventional kanban control system, a new part is demanded and authorized to be released into a stage when a part leaves this particular stage and not when a part leaves the last stage, as is the case with the echelon kanban control system. This implies that in the conventional kanban control system, the placement of a demand and an authorization for the production of a new part into a stage is based on local information from this stage, whereas in the echelon kanban control system, it is based on global information from the last stage. This constitutes a potential advantage of the echelon kanban control system over the conventional kanban control system. Moreover, the echelon kanban control system, just like the conventional kanban control system, depends on only one parameter per stage, the number of echelon kanbans, as we will see later on, and is therefore simpler to optimize and implement than more complicated kanban-type control systems that depend of two parameters per stage, such as the generalized kanban control system [7] and the extended kanban control system [12]. These two apparent advantages of the echelon kanban control system motivated our effort to develop an approximation method for its performance evaluation.

Kanban-type production/inventory systems have often been modeled as queueing networks in the literature. Consequently, most of the techniques that have been developed for the analysis of kanban-type production/inventory systems are based on methods for the performance evaluation of queueing networks. Exact analytical solutions exist for a class of queueing networks known as separable, in which the steady-state joint probabilities have a product-form solution. Jackson [18] was the first to show that the steady-state joint probability of an open queueing network with Poisson arrivals, exponential service times, probabilistic routing, and first-come-first-served (FCFS) service disciplines has a product-form solution, where each station of the network can be analyzed in isolation as an M/M/1 queue. For closed queueing networks of the Jackson type, Gordon and Newell [17] showed that an analytical, product-form solution also exists. The performance parameters of such networks can be obtained using efficient algorithms, such as the mean value analysis (MVA) algorithm [22] and the convolution algorithm [9]. The BCMP theorem [1] summarizes extensions of product-form networks that incorporate alternative service disciplines and several classes of customers.

Since the class of queueing networks for which an exact solution is known (separable networks) is too restrictive for modeling and analyzing real systems, much work has been devoted to the development of approximation methods for the analysis of non-separable networks. Whitt [27] presented an approximation method for the analysis of a general open queueing network that is based on decomposing the network into a set of GI/GI/1 queues and analyzing each queue in isolation. In the case of closed queueing networks, the approximation methods are for the most part based on two approaches. The first approach relies on heuristic extensions of the MVA algorithm (e.g. [23]). The second approach relies on approximating the

performance of the original network by that of an equivalent product-form network. Spanjers et al. [24] developed a method that is based on the second approach for a closed-loop, two-indenture, repairable-item system. Interestingly, their system is equivalent to an echelon kanban control system with a finite population of external jobs. Their method aggregates several states of the underlying continuous-time Markov chain and adjusts some service rates using Norton's Theorem for closed queueing networks to obtain a product-form solution. Among the different methods that rely on the second approach, Marie's method [20] has attracted considerable attention. Extensions and comparative studies of Marie's method have been proposed for a variety of queueing networks [2–5], and [11]. Di Mascolo, Frein and Dallery [14,16] developed approximation methods based on Marie's method for the performance evaluation of the conventional kanban control system and the generalized kanban control system.

The approximation method that we develop in this paper for the performance evaluation of the echelon kanban control system relies on Marie's method. To develop our method, we first model the system as an open queueing network with synchronization stations. By exchanging the roles of jobs (parts) and resources (echelon kanbans) in the open network, we obtain an equivalent, multi-class, nested, closed queueing network, in which the population of each class is equal to the job capacity or number of echelon kanbans of the echelon of stages associated with a particular stage. The echelon of stages associated with a particular stage is the stage itself and all its downstream stages. We then decompose the closed network into a set of nested subsystems, each subsystem being associated with a particular class. This means that we have as many subsystems as the number of the stages. Each subsystem is analyzed in isolation using Marie's method. Each subsystem interacts with its neighboring subsystems in that it includes its downstream subsystem in the form of a single-server station with load-dependent, exponential service rates, and it receives external arrivals from its upstream subsystem. A fixed-point, iterative procedure is used to determine the unknown parameters of each subsystem by taking into account the interactions between neighboring subsystems.

The rest of this paper is organized as follows. In Section 2, we describe the exact operation of the echelon kanban control system by means of a simple example. In Section 3 we present the queueing network model of the echelon kanban control system and the performance measures of the system that we are interested in evaluating. In Section 4, we describe the decomposition of the original system into many subsystems. In Section 5, we present the analysis in isolation of each subsystem, and in Section 6 we develop the analysis of the entire system. In Section 7, we present numerical results on the effects and optimization of the parameters. Finally, in Section 8, we draw conclusions. The analysis of the synchronization stations that appear in the queueing network models of each subsystem is presented in Appendices A and B, and a table of the notation used in the paper is given in Appendix C.

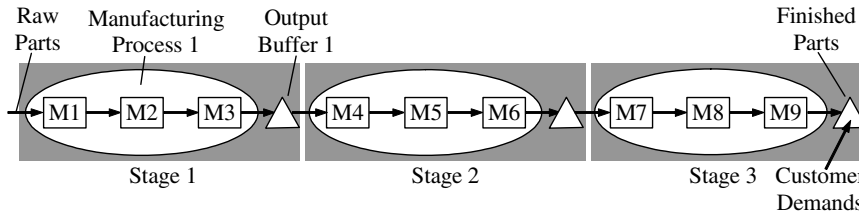


Fig. 1. A serial production system decomposed into three stages in series

2 The echelon kanban control system

In this section, we give a precise description of the operation of the echelon kanban control system by means of a simple example. In this example, we consider a production system that consists of $M = 9$ machines in series, labeled M1 to M9, produces a single part type, and does not involve any batching, reworking or scrapping of parts. Each machine has a random processing time. All parts visit successively machines M1 to M9. The production system is decomposed into $N = 3$ stages. Each stage is a production/inventory system consisting of a manufacturing process and an output buffer. The output buffer stores the finished parts of the stage. The manufacturing process consists of a subset of machines of the original manufacturing system and contains parts that are in service or waiting for service on the machines. These parts represent the work in process (WIP) of the stage and are used to supply the output buffer. In the example, each stage consists of three machines. More specifically, the sets of machines $\{$M1, M2, M3$\}$, $\{$M4, M5, M6$\}$ and $\{$M7, M8, M9$\}$ belong to stages 1, 2 and 3, respectively. The decomposition of the production system into three stages is illustrated in Figure 1.

Each stage has associated with it a number of echelon kanbans that are used to demand and authorize the release of parts into this stage. An echelon kanban of a particular stage traces a closed path through this stage and all its downstream stages. The number of echelon kanbans of stage i is fixed and equal to K_i. There must be at least one echelon kanban of stage i available in order to release a new part into this stage. If such a kanban is available, the kanban is attached onto the part and follows it through the system until the output buffer of the last stage. Since an echelon kanban of stage i is attached to every part in any stage from i to N, the number of parts in stages i to N is limited by K_i.

Parts that are in the output buffer of stage N are the finished parts of the production system. These parts are used to satisfy customer demands. When a customer demand arrives to the system, a demand for the delivery of a finished part from the output buffer of the last stage to the customer is placed. If there are no finished parts in the output buffer of the last stage, the demand cannot be immediately satisfied and is backordered until a finished part becomes available. If there is at least one finished part in the output buffer of the last stage, this part is delivered to the customer after releasing the kanbans of all the stages (1, 2, and 3, in the example) that were attached to it, hence the demand is immediately satisfied. The released kanbans are immediately transferred upstream to their corresponding stages. The kanban of stage i carries with it a demand for the production of a new

stage$-i$ finished part and an authorization to release a finished part from the output buffer of stage $i-1$ into stage i. When a finished part of stage $i-1$ is transferred to stage i, the stage-i kanban is attached to it on top of the kanbans of stages 1 to $i-1$, which have already been attached to the part at previous stages. With this in mind, we can just as well assume that

$$K_i \geq K_{i+1}, i = 1, \ldots, N-1. \tag{1}$$

3 Queueing network model of the echelon kanban control system

In order to develop the approximation method for the performance evaluation of the echelon kanban control system, we first model the system as an open queueing network with synchronization stations. Figure 2 shows the queueing network model of the echelon kanban control system with three stages in series, considered in Section 2. The manufacturing process of each stage is modeled as a subnetwork in which the machines of the manufacturing process are represented by single-server stations. The subnetwork associated with the manufacturing process of stage i is denoted by L_i, and the single-server stations representing machines M1,..., M9 are denoted by S_1,..., S_9, respectively. The number of stations of subnetwork L_i is denoted by m_i. In the example, m_i = 3, i = 1, 2, 3. The echelon kanban control mechanism is modeled via three synchronization stations, denoted by J_i, at the output of each stage i, i = 1, 2, 3.

A synchronization station is a modeling element that is often used to model assembly operations in queueing networks. It can be thought of as a server with instant service times. This server is fed by two or more queues (in our case by two). When there is at least one customer in each of the queues that feed the server, these customers move instantly through and out of the server. This implies that, at any time, at least one of the queues that feed the server is empty. Customers that enter the server, exit the server after possibly having been split into more or merged into fewer customers. In our case, the queues in each synchronization station contain either parts or demands combined with kanbans.

To illustrate the operation of the synchronization stations, let us first focus on any synchronization station J_i, except that of the last stage. This synchronization station represents the synchronization between a stage-i finished part and a stage-$(i+1)$ free kanban. Let PA_i and DA_{i+1} denote the two queues of J_i. PA_i represents

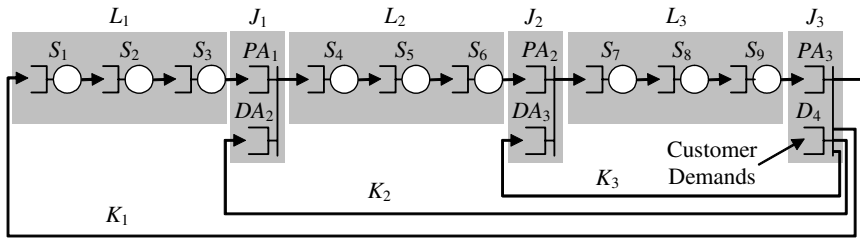


Fig. 2. Queueing network model of the echolon kanban control system of Figure 1

the output buffer of stage i and contains stage-i finished parts, each of which has attached to it a kanban from each stage from 1 to i. DA_{i+1} contains demands for the production of new stage-$(i+1)$ parts, each of which has attached to it a stage-$(i+1)$ kanban. The synchronization station operates as follows. As soon as there is one entity in each queue PA_i and DA_{i+1}, the stage-i finished part engages the stage-$(i+1)$ kanban without releasing the kanbans from stages 1 to i that were already attached to it, and joins the first station of stage $i+1$. Note that at stage 1, as soon as a stage-1 kanban is available, a new part is immediately released into stage 1 since there are always raw parts at the input of the system.

Let us now consider the last synchronization station J_N (J_3 in the example). J_N synchronizes queues PA_N, and D_{N+1}. PA_N represents the output buffer of stage N and contains stage-N finished parts, each of which has attached to it a kanban from each stage from 1 to N. D_{N+1} contains customer demands. When a customer demand arrives to the system, it joins D_{N+1}, thereby demanding the release of a finished part from PA_N to the customer. If there is a finished part in queue PA_N, it is released to the customer and the demand is satisfied. In this case, the finished part in PA_N releases the kanbans that were attached to it, and these kanbans are transferred upstream to queues DA_i $(i = 1, \ldots, N)$. The kanban of stage i carries along with it a demand for the production of a new stage-$i(i = 1, \ldots, N)$ finished part and an authorization for the release of a finished part from queue PA_{i-1} into stage i. If there are no finished parts in queue PA_N, the customer demand remains on hold in D_{N+1} as a backordered demand.

An important special case of the echelon kanban control system in the case where there are always customer demands for finished parts. This case is known as the *saturated* echelon kanban control system. Its importance lies in the fact that its throughput determines the maximum capacity of the system. In the saturated system, when there are finished parts at stage N, they are immediately consumed and an equal number of parts enter the system. As far as the queueing network corresponding to this model is concerned, the synchronization station J_N can be eliminated since queue D_{N+1} is never empty and can therefore be ignored. In the saturated echelon kanban control system, when the processing of a part is completed at stage N, this part is immediately consumed after releasing the kanbans of stages 1,..., N that were attached to it and sending them back to queues $DA_i(i = 1, \ldots, N)$.

It is worth noting that the echelon kanban control system contains the make-to-stock CONWIP system [23] as a special case. In the make-to-stock CONWIP system, as soon as a finished part leaves the production system to be delivered to a customer, a new part enters the system to begin its processing. An echelon kanban control system with $K_1 \leq K_i, i \neq 1$, behaves exactly like the make-to-stock CONWIP system.

The dynamic behavior of the echelon kanban control system depends on the manufacturing processes, the arrival process of customer external demands, and the number of echelon kanbans of each stage. Among the performance measures that are of particular interest are the average work in process (WIP) and the average number of finished parts in each stage, the average number of backordered (not immediately satisfied) demands, and the average waiting time and percentage of

backordered demands. In the case of the saturated echelon kanban control system, the main performance measure of interest is its production rate, P_r, i.e. the average number of finished parts leaving the output buffer of stage N per unit of time. P_r represents the maximum rate at which customer demands can be satisfied. With this in mind, the average arrival rate of external customer demands in the unsaturated system, say λ_D, must be strictly less than P_r in order for the system to meet all the demands in the long run. In other words, the stability condition for the unsaturated system is

$$\lambda_D < P_r. \tag{2}$$

4 Decomposition of the echelon kanban control system

To evaluate the performance of the multi-stage, serial, echelon kanban control system, we decompose the system into many nested, single-stage subsystems and analyze each system in isolation. The susbsystems are nested in each other in such a way that each subsystem includes its downstream subsystem in the form of a single-server station and receives external arrivals from its upstream subsystem. The first subsystem mimics the original system. To analyze each subsystem, we view it as a closed queueing network and we approximate each station of this network by an exponential-service station with load-dependent service rates. The resulting network is a product-form network. A fixed-point iterative procedure is then used to determine the unknown parameters of each subsystem by taking into account the interactions between neighboring subsystems. A detailed description of the decomposition follows.

Consider the queueing network model of an echelon kanban control system consisting of N stages in series as described in Section 3 (see Fig. 2 for $N = 3$). Let us denote the queueing network of the system by R. Our goal is to analyze R by decomposing it into a set of N nested subsystems, $R^i, i = 1, \ldots, N$. This is done as follows (see Fig. 3 for $N = 3$).

Subsystem R^N (R^3 in the example) is an open queueing network with restricted capacity consisting of 1) an upstream synchronization station, denoted by I^N, representing J_{N-1} in the original system, 2) the subnetwork of stations L_N of the original system, and 3) a downstream synchronization station, denoted by O^N, representing J_N in the original system. Each subsystem R^i, $i = 2, \ldots, N-1$, is an open queueing network with restricted capacity consisting of 1) an upstream synchronization station, denoted by I^i, representing J_{i-1} in the original system, 2) the subnetwork of stations L_i of the original system, and 3) a downstream single-server pseudo-station, denoted by $\hat{S}_i$, representing the part of the system downstream of L_i in the original system. Finally, subsystem R^1 is a closed queueing network consisting of 1) the subnetwork of stations L_1 of the original system, and 2) a downstream single-server pseudo-station, denoted by $\hat{S}_1$, representing the part of the system downstream of L_1 in the original system. Note that pseudo-station $\hat{S}_i$ in subsystem R^i, $i = 1, \ldots, N-1$, is an aggregate representation of subsystem R^{i+1}.

The number of echelon kanbans of subsystem R^i is K_i. Subsystem R^N is synchronized with two external arrival processes, one at synchronization station I^N

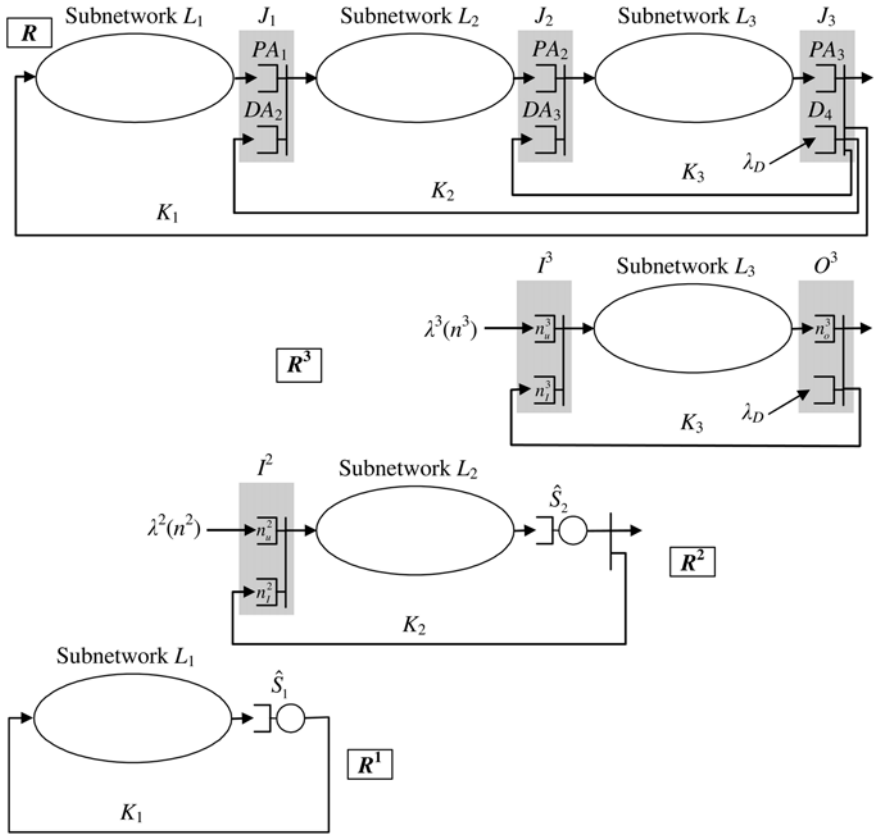


Fig. 3. Illsutration of the decomposition of a 3-stage echolon kanban control system

concerning parts that arrive from subnetwork L_{N-1}, and the other at synchronization station O^N concerning customer demands. Subsystem R^i, $i = 2, \ldots, N-1$, is synchronized with only one external arrival process at synchronization station I^i concerning parts that arrive from subnetwork L_{i-1}. Subsystem R^1 is a closed network; therefore it is not synchronized with any external arrival processes. As can be seen from Table 3, each synchronization station J_i of the original network R, linking stage i to stage $i+1$, is represented only once in the decomposition.

To completely characterize each subsystem R^i, $i = 2, \ldots, N$, we assume that each of the external arrival processes to R^i is a state-dependent, continuous-time Markov process. Let $\lambda^i(n^i)$ denote the state-dependent arrival rate of stage-i raw parts at the upstream synchronization station I^i of subsystem R^i, where n^i is the state of subsystem R^i and is defined as the number of parts in this subsystem. Let Q_u^i and Q_I^i be the two queues of synchronization station I^i, containing n_u^i and n_I^i customers, respectively, where n_u^i is the number of finished parts of stage i-1 waiting to enter subnetwork L_i, and n_I^i is the number of free stage-i kanbans waiting to authorize the release of stage-($i-1$ <) finished parts into subnetwork L_i. Then, it is clear that the only possible states of the synchronization station are

the states $(n_I^i, 0)$, for $n_I^i = 0, \ldots, K_i$, and $(0, n_u^i)$, for $n_u^i = 0, \ldots, K_{i-1} - K_i$; therefore, the state n^i of subsystem R^i can be simply obtained from n_u^i and n_I^i using the following relation:

$$n^i = \begin{cases} K_i - n_I^i \text{ if } n_I^i \neq 0, \\ K_i + n_u^i \text{ if } n_I^i = 0. \end{cases} \tag{3}$$

The above relation implies that $0 \leq n^i \leq K_{i-1}$. Also, since the number of raw parts at the input of stage i cannot be more than the number of stage-$(i-1)$ kanbans, $\lambda^i(K_{i-1}) = 0$. In subsystem R^N, besides the arrival rate of stage-N raw parts at I^N, $\lambda^N(n^N)$, there is also the external arrival rate of customer demands at O^N, λ_D. Subsystem R^1, as was mentioned above, is a closed network and therefore has no external arrival processes to define.

To obtain the performance of the original network R, the following two problems must be addressed: 1) How to analyze each subsystem R^i, $i = 1, \ldots, N$, assuming that the external arrival rates are known (except in the case of the first subsystem R^1, where there are no external arrivals), and 2) how to determine the unknown external arrival rates. These two problems are addressed in Sections 5 and 6, respectively. Once these two problems have been solved, the performance of each stage of the original network R can be obtained from the performances of subsystems R^i, $i = 1, \ldots, N$.

5 Analysis of each subsystem in isolation

In this section, we describe how to analyze each subsystem in isolation using Marie's approximate analysis of general closed queueing networks [20]. Throughout this analysis, the state-dependent rates of the external arrival processes, $\lambda^i(n^i), 0 \leq n^i \leq K_{i-1}, i = 2, \ldots, N$, are assumed to be known. To analyze each subsystem using Marie's method, we first view the subsystem as a closed queueing network. For subsystems $R^i, i = 2, \ldots, N$, this is done by considering the kanbans of stage i as the customers of the closed network, and the parts and demands (in the case of the last subsystem R^N) as external resources. Note that the queueing network associated with subsystem R^1 is already being modeled as a closed queueing network in the decomposition. Its customers are the kanbans of stage 1.

The closed queueing network associated with subsystem R^N is partitioned into m_N + 2 stations, namely, the synchronization stations I^N and O^N and the m_N stations of subnetwork L_N. Similarly, the closed queueing network associated with each subsystem R^i is partitioned into $m_i + 2$ stations, namely, the synchronization station I^i, the m_i stations of subnetwork L_i, and station $\hat{S}_i$. Finally, the closed queueing network associated with subsystem R^1 is partitioned into m_1+1 stations, namely, the m_1 stations of subnetwork L_1, and station $\hat{S}_1$. Each station is approximated by an exponential-service station with load-dependent service rates. The resulting network associated with each subsystem is a Gordon-Newell, product-form network [17] consisting of K_i customers and $m_i + 2$ stations for subsystems $R^i, i = 2, \ldots, N$, and $m_1 + 1$ stations for subsystem R^1. The stations within each subsystem R^i, $i = 1, \ldots, N$, will be denoted by the index $k \in M_i$,

where $M_1 = \{1, \ldots, m_1, \hat{S}\}, M_i = \{I, 1, \ldots, m_i, \hat{S}\}$ for $i = 2, \ldots, N-1$, and $M_N = \{I, 1, \ldots, m_N, O\}$.

Let $\mu_k^i(n_k^i)$ denote the load-dependent service rate of station k in the product-form network of subsystem R^i when there are n_k^i customers in that station. We will show how to determine $\mu_k^i(n_k^i)$, $n_k^i = 1, \ldots, K_i$, for each station $k \in M_i$ within a particular subsystem R^i, $i = 1, \ldots, N$. The method for doing this is the same for all subsystems R^i, $i = 1, \ldots, N$; therefore, for the sake of notational simplicity we will drop index i that denotes variables associated with subsystem R^i.

Let vector $\mathbf{n} = (n_k, k \in M)$ be the state of the closed, product-form network, where n_k denotes the number of customers present at station k. Then, the probability of being in stage $\mathbf{n}$, $P(\mathbf{n})$, is given by the following product-form solution [12]:

$$P(\mathbf{n}) = \frac{1}{G(K)} \prod_{k \in M} \left[\prod_{n=1}^{n_k} \frac{V_k}{\mu_k(n)} \right], \tag{4}$$

where V_k is the average visit ratio of station k in the original system and is given from the routing matrix of the original system, and $G(K)$ is the normalization constant.

To determine the unknown parameters $\mu_k(n_k), n_k = 1, \ldots, K$, for each station $k \in M$, in the product-form solution (4), each station is analyzed in isolation as an open system with a state-dependent, Poisson arrival process, whose rate $\lambda_k(n_k)$ depends on the total number of customers, n_k, present in the station. Let T_k denote this open system. Assume that the rates $\lambda_k(n_k)$ are known for $n_k = 1, \ldots, K-1$. The open queueing system T_k can then be analyzed in isolation using any appropriate technique to obtain the steady-state probabilities of having n_k customers in the isolated system, say $P_k(n_k)$. The issue of analyzing each queueing system T_k in isolation will be discussed immediately after Algorithm 1, below. Once the probabilities $P_k(n_k)$ are known, the conditional throughput of T_k when its population is n_k, which is denoted by $v_k(n_k)$, can be derived using the relation [12],

$$v_k(n_k) = \lambda_k(n_k - 1) \frac{P_k(n_k - 1)}{P_k(n_k)}, \quad \text{for } n_k = 1, \ldots, K. \tag{5}$$

The load-dependent service rates of the k-th station of the closed product-form network are then set equal to the conditional throughputs of the corresponding open station in isolation, i.e.:

$$\mu_k(n_k) = v_k(n_k), \quad \text{for } n_k = 1, \ldots, K. \tag{6}$$

Once the rates $\mu_k(n_k)$ have been obtained, the state-dependent arrival rates $\lambda_k(n_k)$ can be obtained from the generalized, product-form solution as [6,12]:

$$\lambda_k(n_k) = V_k \frac{G_k(K - n_k - 1)}{G_k(K - n_k)}, \quad \text{for } n_k = 1, \ldots, K-1, \text{ and } \lambda_k(\mathrm{K}) = 0, \tag{7}$$

where $G_k(n)$ is the normalization constant of the closed, product-form network with station k removed (complementary network) and population n. $G_k(n)$ is a function of the parameters $\mu_{k'}(n_{k'})$ for all $k' \neq k$ and $n_{k'} = 1, \ldots, K$, and can be computed efficiently using any computational algorithm for product-form networks [6,9]. An

iterative procedure can then be used to determine these unknown quantities. This procedure is described by the following algorithm.

Algorithm 1: Analysis of a Subsystem in Isolation.

Step 0. (Initialization) Set $\mu_k(n_k)$ to some initial value, for $k \in M$ and $n_k = 1, \ldots, K$.

Step 1. For $k \in M$:

Calculate the state-dependent arrival rates $\lambda_k(n_k)$, for $n_k = 0, \ldots, K-1$, using (7).

Step 2. For $k \in M$:

1. Analyze the open queueing system T_k.
2. Derive the steady state probabilities $P_k(n_k)$ of having n_k customers, for $n_k = 1, \ldots, K$.
3. Calculate the conditional throughputs $v_k(n_k)$, for $n_k = 1, \ldots, K$, using (5).

Step 3. For $k \in M$:

Set the load-dependent service rates $\mu_k(n_k)$, for $n_k = 1, \ldots, K$, in the closed, product-form network using (6).

Step 4. Go to Step 1 until convergence of the parameters $\mu_k(n_k)$.

Next, we show how to analyze each open queueing system T_k. To do this, we reintroduce index i to denote subsystem R^i. Step 2.1 of Algorithm 1 above requires the analysis of the open queueing systems T_k^i for $k \in M_i$ and $i = 1, \ldots, N$. There are four different types of queueing systems: 1) the synchronization station O^N in subsystem R^N, 2) the synchronization stations I^i in subsystems $R^i, i = 2, \ldots, N$, 3) the m_i stations in each subnetwork $L_i, i = 1, \ldots, N$, and 4) the pseudo-stations $\hat{S}_i$ in subsystems R^i, $i = 1, \ldots, N-1$.

First, consider the analysis of synchronization station O^N in subsystem R^N. O^N is a synchronization station fed by a continuous-time Markov arrival process with state-dependent rates, $\lambda_O^N(n_O^N)$, $0 \leq n_O^N \leq K_N$, and an external Poisson process with fixed rate λ_D. An exact solution for this system is easy to obtain by solving the underlying continuous-time Markov chain. Namely, the steady-state probabilities $P_O^N(n_O^N)$ of having n_O^N customers in subsystem O^N can be derived, and the conditional throughput $v_O^N(n_O^N)$ can be estimated using (5) (see [11] and Appendix A).

The synchronization station I^i in each subsystem $R^i, i = 2, \ldots, N$, is a synchronization station fed by two continuous-time Markov arrival processes with state-dependent rates, $\lambda_I^i(n_I^i)$, $0 \leq n_I^i \leq K_i$, and $\lambda^i(n^i), 0 \leq n^i \leq K_{i-1}$. An exact solution for this system is also easy to obtain by solving the underlying continuous-time Markov chain. (see [14] and Appendix B).

The analysis in isolation of any station $k \in \{1, \ldots, m_i\}$ in each subnetwork $L_i, i = 1, \ldots, N$, reduces to the analysis of a $\lambda_k^i(n_k^i)/G_i/1/N$ queue. Classical methods can be used to analyze this queue to obtain the steady-state probabilities $P_k^i(n_k^i)$. For instance, if the service time distribution is Coxian, the algorithms given in [21] may be used. For multiple-server stations, we can use the numerical

technique presented in [26]. The conditional throughput $v_k^i(n_k^i)$ can then be derived from the state probabilities using (5). In the special case where the service time is exponentially distributed, the conditional throughput $v_k^i(n_k^i)$ is simply equal to the load-dependent service rate $\mu_k^i(n_k^i)$ [12].

Finally, as was mentioned earlier, pseudo-station $\hat{S}_i$ in subsystem R^i, $i = 1, \ldots, N-1$, is an aggregate representation of subsystem R^{i+1}, which is nested inside subsystem R^i. Therefore, the conditional throughput of pseudo-station $\hat{S}_i$, $v_{\hat{S}}^i(n_{\hat{S}}^i)$, is set equal to the conditional throughput of subsystem R^{i+1}. The conditional throughput of any subsystem R^i, $i = 2, \ldots, N$, is denoted by $v^i(n^i)$ and can be estimated by the following simple expression [3]:

$$v^i(n^i) = \begin{cases} \lambda_I^i(K_i - n^i) & \text{for } 1 \le n^i \le K_i, \\ \lambda_I^i(0) & \text{for } K_i \le n^i \le K_{i-1}. \end{cases} \tag{8}$$

6 Analysis of the entire echelon kanban control system

In Section 5 we analyzed each subsystem of the decomposition in isolation, given that the arrival rates of the external arrival processes were known. In this section, we show how to determine these arrival rates.

Consider again the queueing network of the original system,R, which was decomposed into N subsystems (see Fig. 3 for $N = 3$). In each subsystem R^i, $i = 2, \ldots, N$, the unknown parameters involved in the decomposition are the arrival rates of raw parts at each upstream synchronization station I^i, $\lambda^i(n^i), 0 \le n^i \le K_{i-1}$. Recall that pseudo-station $\hat{S}_{i-1}$ in subsystem R^{i-1} represents subsystem $R^i, i = 2, \ldots, N$; therefore, the external arrival process of raw parts at synchronization station I^i in subsystem R^i should be identical to the arrival process of parts at pseudo-station $\hat{S}_{i-1}$ in subsystem R^{i-1}. The latter process was involved in the analysis of subsystem R^{i-1} in isolation and was modeled as a state-dependent Poisson arrival process with rate $\lambda_{\hat{S}}^{i-1}(n_{\hat{S}}^{i-1})$, $0 \le n^{i-1} \le K_{i-1}$. As a result, the following set of equations holds:

$$\lambda^i(n^i) = \lambda_{\hat{S}}^{i-1}(n_{\hat{S}}^{i-1}) \text{ for } 0 \le n^i = n_{\hat{S}}^{i-1} \le K_{i-1} \text{ and } i = 2, \ldots, N. \tag{9}$$

Equation (9) implies that the unknown parameters $\lambda^i(n^i)$ are the solutions of a fixed-point problem. To determine these quantities we use an iterative procedure. This procedure is given by Algorithm 2 below. Algorithm 2 consists of several forward and backward steps. A forward step from subsystem R^{i-1} to R^i uses new estimates of the arrival rates to the upstream synchronization station I^i of subsystem R^i, $\lambda^i(n^i)$, to resolve R^i using Algorithm 1. A backward step from R^i to R^{i-1} solves R^{i-1} using Algorithm 1, given that the arrival rates $\lambda^i(n^i)$ to the upstream synchronization station I^i of each subsystem R^j, $j = i, \ldots, N$, have converged. The procedure starts with subsystem R^N and moves backwards until it reaches subsystem R^1. Subsystem R^N is analyzed first using Algorithm 1 and current estimates of $\lambda^N(n^N)$. This yields the conditional throughput of R^N, $v^N(n^N)$, which is needed to analyze subsystem R^{N-1}, since it determines the load-dependent exponential-service rates of pseudo-station $\hat{S}_{N-1}$. Subsystem R^{N-1}

is analyzed next using Algorithm 1 and current estimates of $\lambda^{N-1}(n^{N-1})$. This yields the conditional throughput of R^{N-1}, $v^{N-1}(n^{N-1})$, and the arrival rates to the pseudo-station $\hat{S}_{N-1}$, $\lambda^{N-1}_{\hat{S}}(n^{N-1}_{\hat{S}})$. If these arrival rates are not equal to the current estimates of the arrival rates $\lambda^{N}(n^{N})$, then the latter rates have not converged. In this case, the current estimates of $\lambda^{N}(n^{N})$ are updated to $\lambda^{N-1}_{\hat{S}}(n^{N-1}_{\hat{S}})$ and subsystem R^N is analyzed again using Algorithm 1 with the new estimates. Otherwise, the arrival rates $\lambda^{N}(n^{N})$ have converged and the procedure moves on to the analysis of subsystem R^{N-2} using Algorithm 1, where the load-dependent exponential-service rates of pseudo-station $\hat{S}_{N-2}$ are set equal to $v^{N-1}(n^{N-1})$. This procedure is repeated for subsystems R^{N-2},R^{N-3},..., until the first subsystem, R^1, is reached and all the arrival rates $\lambda^i(n^i)$, $i = 2, ..., N$, have converged. All the performance parameters of interest can then be derived.

Algorithm 2: Analysis of a multi-stage echelon kanban control system.

Step 0. (Initialization) Set the unknown arrival rates of each subsystem R^i to some initial values, e.g., $\lambda^i(n^i) = \lambda_D, 0 \leq n^i \leq K_{i-1}$, and $i = 2, \ldots, N$.

Step 1. Computation and convergence of the arrival rates, $\lambda^i(n^i)$, $i = 2, \ldots, N$.
Set $i = N$
While $i \geq 1$
  If $i = N$
    Solve subsystem R^N using Algorithm 1 and calculate the throughput $v^N(n^N)$, $n^N = 1, \ldots, K_{N-1}$, from (8).
    Set $i = i - 1$.
  Else
    Solve subsystem R^i using Algorithm 1 and calculate the arrival rate $\lambda^i_{\hat{S}}(n^i_{\hat{S}})$, $n^i_{\hat{S}} = 0, \ldots, K_i$, and the throughput $v^i(n^i)$, $n^i = 1, \ldots, K_{i-1}$, from (8).
    If $\lambda^i_{\hat{S}}(n^{i+1}) = \lambda^{i+1}(n^{i+1})$, $n^{i+1} = 0, \ldots, K_i$,
      Set $i = i - 1$
    Else
      Set $\lambda^{i+1}(n^{i+1}) = \lambda^i_{\hat{S}}(n^{i+1})$, $n^{i+1} = 0, \ldots, K_i$, and set $i = i + 1$
    Endif
  Endif
Endwhile

In the case of the saturated echelon kanban control system, we can use the same algorithm. The only difference is in the analysis of subsystem R^N in Algorithm 1, where there is no downstream synchronization station O^N. As far as the convergence properties of Algorithms 1 and 2 are concerned, in all of the numerical examples that we examined (see Sect. 7), both algorithms converged. The convergence criterion was that the relative difference between the values of every unknown parameter at two consecutive iterations should be less that 10^{-4}.

Once Algorithm 2 has converged, all the performance parameters of the system can be calculated. Indeed, from the analysis of each subsystem R^i using Algorithm 1, it is possible to derive the performance parameters of stage i in the original network R, especially the throughput and the average length of each queue, including the queues of the synchronization stations. Thus, in the case of the saturated

echelon kanban system, we can derive the throughput, the average WIP, the average number of finished parts, and the average number of free echelon kanbans for each stage. In the case of the echelon kanban control system with external demands, some other important performance measures can be derived from the analysis of subsystem R^N, namely, the proportion of backordered demands, p_B, the average number of backordered demands, Q_D, and the average waiting time of backordered demands, W_B. These performance measures can be derived as follows [11,14]:

$$p_B = P_O^N(0), \quad Q_D = P_O^N(0)\frac{1}{\frac{\lambda_O^N(0)}{\lambda_D} - 1}, \quad W_B = \frac{Q_D}{p_B \lambda_D},$$

where $\lambda_O^N(0)$ is the arrival rate of finished parts at synchronization station O^N when there are no finished parts at that station and $P_O^N(0)$ is the steady-state probability of having no finished parts at synchronization station O^N.

7 Numerical results

In this section, we test the approximation method for the performance evaluation of the echelon kanban control system that we developed in Sections 4–6 on several numerical examples. The approximation method was implemented on an Intel Celeron PC @ 433 MHz, and its results are compared to simulation results obtained using the simulation software ARENA on an AMD Athlon PC @ 1400 MHz. For each simulation experiment we run a single replication. The length of this replication was set equal to the time needed for the system to produce 68 million parts. The initial condition of the system at the beginning of the replication was set to a typical regenerative state, namely the state where all customer demands and demands for the production of new parts at all stages have been satisfied. This permitted us to set the warm-up period at the beginning of the replication equal to zero. In all simulation experiments we used 95% confidence intervals. The numerical examples are organized into Sections 7.1 and 7.2. In Section 7.1, we study the accuracy and rapidity of the approximation method as well as the influence of some key parameters of the echelon kanban control system on system performance. In Section 7.2, we use the approximation method to optimize the design parameters (echelon kanbans) of the system.

7.1 Influence of parameters

In this section, we test the accuracy and rapidity of the approximation method with two numerical examples in which we vary the number of stages, the number of kanbans in each stage, and the service-time distributions of the manufacturing process of each stage. For each example, we consider first the case of the saturated system and then the case of the system with external demands. In each example, we compare the performance of the system obtained by the approximation method to that obtained by simulation. We also compare the performance of the echelon kanban control system obtained by the approximation method and by simulation to

Table 1. Production capacity of the saturated echelon kanban control system (Example 1)

Configuration	Simulation		Approximation		
	Production capacity	Confidence interval	Production capacity	Relative error	Iterations
1.1: $N = 3; K = 1$	0.581	$\pm 0.1\%$	0.571	-1.8%	7
1.2: $N = 3; K = 3$	0.809	$\pm 0.1\%$	0.804	-0.6%	7
1.3: $N = 3; K = 5$	0.877	$\pm 0.2\%$	0.873	-0.5%	7
1.4: $N = 3; K = 10$	0.934	$\pm 0.5\%$	0.933	-0.1%	7
1.5: $N = 3; K = 15$	0.955	$\pm 0.6\%$	0.954	-0.1%	7
1.6: $N = 5; K = 1$	0.522	$\pm 0.0009\%$	0.502	-4%	16
1.7: $N = 5; K = 3$	0.772	$\pm 0.1\%$	0.761	-1.4%	16
1.8: $N = 5; K = 5$	0.850	$\pm 0.1\%$	0.843	-0.8%	16
1.9: $N = 5; K = 10$	0.919	$\pm 0.2\%$	0.916	-0.3%	16
1.10: $N = 5; K = 15$	0.945	$\pm 0.0009\%$	0.942	-0.3%	16
1.11: $N = 10; K = 1$	0.485	$\pm 0.0007\%$	0.456	-6.4%	56
1.12: $N = 10; K = 3$	0.745	$\pm 0.5\%$	0.730	-2.1%	56
1.13: $N = 10; K = 5$	0.831	$\pm 0.7\%$	0.820	-1.3%	56
1.14: $N = 10; K = 10$	0.908	$\pm 0.1\%$	0.902	-0.7%	56
1.15: $N = 10; K = 15$	0.937	$\pm 0.1\%$	0.933	-0.4%	56

the performance of the conventional or installation kanban control system obtained by a similar approximation method developed in [14] and by simulation.

Example 1. In Example 1, we consider an echelon kanban system composed of N identical stages, where each stage contains a single machine with exponentially distributed service times with mean equal to 1. In order to compare the echelon kanban control system to the conventional kanban control system, we first set the number of installation kanbans of each stage i in the conventional kanban control system, say K_i^c, equal to some constant, K, i.e. $K_i^c = K$. Then, we set the number of echelon kanbans of each stage i in the echelon kanban control system, say K_i^e, equal to the sum of the installation kanbans of stages $i, \ldots, N$, in the conventional kanban control system, i.e. $K_i^e = \sum_{j=i}^{N} K_j^c = (N + 1 - i)K$.

For the case of the saturated system, the main performance parameter of interest is the throughput of the system, which determines the production capacity of the system. Table 1 shows the throughput of the saturated echelon kanban control system obtained by the approximation method and by simulation, for different values of N and K. The same table also shows the 95% confidence interval for the simulation results, the percentage of relative error of the approximation method with respect to simulation, and the number of iterations of Algorithm 2 that are needed to reach convergence. Table 2 shows the same results for the conventional kanban control system obtained in [14].

From the results in Table 1, we note that the number of iterations of Algorithm 2 of the approximation method increases with the number of stages, as is expected.

Table 2. Production capacity of the saturated conventional kanban control system (Example 1)

Configuration	Simulation		Approximation		
	Production capacity	Confidence interval	Production capacity	Relative error	Iterations
1.1: $N = 3; K = 1$	0.562	$\pm 0.5\%$	0.547	-2.7%	2
1.2: $N = 3; K = 3$	0.800	$\pm 0.7\%$	0.792	-1.0%	2
1.3: $N = 3; K = 5$	0.869	$\pm 1.3\%$	0.865	-0.5%	2
1.4: $N = 3; K = 10$	0.926	$\pm 0.8\%$	0.928	$+0.2\%$	2
1.5: $N = 3; K = 15$	0.952	$\pm 1.2\%$	0.951	-0.1%	2
1.6: $N = 5; K = 1$	0.484	$\pm 0.6\%$	0.449	-7.0%	4
1.7: $N = 5; K = 3$	0.746	$\pm 0.8\%$	0.731	-2.0%	4
1.8: $N = 5; K = 5$	0.833	$\pm 0.8\%$	0.822	-1.3%	4
1.9: $N = 5; K = 10$	0.901	$\pm 1.2\%$	0.904	$+0.3\%$	4
1.10: $N = 5; K = 15$	0.943	$\pm 1.1\%$	0.934	-0.9%	4
1.11: $N = 10; K = 1$	0.429	$\pm 0.5\%$	0.379	-11.6%	7
1.12: $N = 10; K = 3$	0.704	$\pm 0.7\%$	0.680	-3.4%	6
1.13: $N = 10; K = 5$	0.806	$\pm 0.9\%$	0.786	-2.6%	5
1.14: $N = 10; K = 10$	0.855	$\pm 0.5\%$	0.883	-3.2%	5
1.15: $N = 10; K = 15$	0.917	$\pm 1.3\%$	0.919	$+0.2\%$	5

Specifically, for $N = 3, 5$, and 10, we have 7, 16, and 56 iterations of Algorithm 2, respectively. As far as the convergence of Algorithm 1 is concerned, we also note that subsystem R^N requires two iterations of Algorithm 1, subsystem R^1 requires one iteration, and all other subsystems require three iterations, irrespectively of the number of stages N, for all the configurations tested. The simulation time is extremely long (over two hours) compared to the time required for the approximation method, which is approximately 1–10 seconds. From Table 1, we see that as the number of echelon kanbans increases, for a given number of stages N, the throughput also increases and asymptotically tends to the production rate of each machine in isolation. Moreover, the throughput seems to be decreasing in the number of stages. The results obtained by the approximation method are fairly accurate when compared to the simulation results. The relative error is very small in general except for the cases where $K = 1$, where we observe somewhat significant errors. This happens because when the number of echelon kanbans is small, there are strong dependence phenomena among stations and these phenomena are not captured well by the state-dependent, continuous-time, Markov arrival processes assumed in the decomposition method. Comparing the results between Tables 1 and 2, we note that the production capacity of the echelon kanban control system is always higher than that of the conventional kanban control system, given that the two systems have the same value of K.

For the system with backordered demands, the main performance parameters of interest are the proportion of backordered demands, p_B, the average number of backordered demands, Q_D, and the average waiting time of backordered demands, W_B, as defined at the end of Section 6. Table 3 shows these performance parameters obtained by the approximation method and by simulation, for the configurations of parameters 1.3, 1.8, and 1.13 of Table 1, i.e. for $K = 5$, and different values of the customer demand rate, λ_D. The same table also shows the 95% confidence interval for the simulation results and the number of iterations of Algorithm 2 that are needed to reach convergence. Table 4 shows the same results for the conventional kanban control system obtained in [14].

From the results in Table 3, we note that as the customer demand arrival rate increases, the number of iterations of Algorithm 2 also increases, though not dramatically. As far as the average number of backordered demands, Q_D, is concerned, we note that the analytical method is fairly accurate. This is not true for the average waiting time of backordered demands, W_B, where in some cases the difference between the approximation method and simulation are significant. Comparing the results between Tables 3 and 4, we note that the echelon kanban control system always has a smaller average number of backordered demands, Q_D, than the conventional kanban control system, given that the two systems have the same value of K. The difference in the average number of backordered demands is more pronounced when the two systems are highly loaded, i.e. when λ_D is close to the production capacity.

Table 5 shows the results for the average number of finished parts (FP) and the average work-in-process (WIP) at each stage for the configurations of parameters 1.17 and 1.19 in Table 3. Table 6 shows the same results for the conventional kanban control system.

Comparing the results between Tables 5 and 6, we note that the echelon kanban control system has slightly higher average WIP and lower FP than the conventional kanban control system, when the two systems are highly loaded (i.e. λ_D is close to P_r), and given that the two systems have the same value of K. When the two systems are not highly loaded, the difference in average WIP and FP between the two systems is very small. Finally, it appears that the difference in average WIP and FP between the echelon kanban control system and the conventional kanban control system is higher in upstream stages than in downstream stages.

Although the above observations hold for the particular configurations of parameters examined, we expect that they should also hold for the other configurations of Table 1 and different values of the customer demand rate, λ_D, because to a large extent they are due to the fact that the echelon kanban control system always responds faster to customer demands than the conventional kanban control system, given that the two systems have the same value of K.

Finally, we should note that the approximation method for the performance evaluation of the conventional kanban control system developed in [14] is also based on decomposing a system of N stages into N subsystems. The total number of the unknown parameter sets (the arrival rates of the external arrival processes to the subsystems) that must be determined for the conventional kanban control system, however, is twice as big as that which must be determined for the echelon

Table 3. Average number of backordered demands, average waiting time of backordered demands, and proportion of backordered demands for the echelon kanban system (Example 1)

Configuration	Q_D	W_B	p_B (%)	Iterations
1.16: $N = 3; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	6
Simulation	0.0	0.0	0.0	
1.17: $N = 3; K = 5; \lambda_D = 0.5$				
Approximation	0.035	4.069	1.729	7
Simulation	0.034 ($\pm$0.9%)	2.066 ($\pm$1.2%)	3.337	
1.18: $N = 3; K = 5; \lambda_D = 0.625$				
Approximation	0.221	4.594	7.687	7
Simulation	0.213 ($\pm$0.1%)	3.014 ($\pm$14.2%)	11.32	
1.19: $N = 3; K = 5; \lambda_D = 0.8$				
Approximation	4.176	10.791	48.38	8
Simulation	4.095 ($\pm$3.6%)	9.755 ($\pm$7%)	52.47	
1.20: $N = 5; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	16
Simulation	0.0	0.0	0.0	
1.21: $N = 5; K = 5; \lambda_D = 0.5$				
Approximation	0.035	4.070	1.71	16
Simulation	0.032 ($\pm$0.007%)	3.189 ($\pm$0.003%)	2.03	
1.22: $N = 5; K = 5; \lambda_D = 0.8$				
Approximation	6.774	14.440	58.69	22
Simulation	6.5686 ($\pm$0.08%)	12.895 ($\pm$0.02%)	63.67	
1.23: $N = 10; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	20
Simulation	0.0	0.0	0.0	
1.24: $N = 10; K = 5; \lambda_D = 0.5$				
Approximation	0.035	4.070	1.72	39
Simulation	0.023 ($\pm$0.005%)	3.512 ($\pm$0.002%)	1.28	
1.25: $N = 10; K = 5; \lambda_D = 0.77$				
Approximation	3.817	10.709	46.3	61
Simulation	3.131 ($\pm$0.003%)	9.064 ($\pm$0.001%)	49.3	

kanban control system (namely, there are $2(N-1)$ external arrival rates for the conventional kanban control system compared to $N-1$ external arrival rates for the echelon kanban control system). Yet, for both examples examined, the number of iterations needed for the convergence of the parameters is significantly lower for the conventional kanban control system than for the echelon kanban control system, given the same convergence criterion for the two systems, as can be seen from Tables 1–4. This is due to the fact that the coordination of production is decentralized in

Table 4. Average number of backordered demands, average waiting time of backordered demands, amd proportion of backordered demands for the conventional kanban control system (Example 1)

Configuration	Q_D	W_B	p_B (%)	Iterations
1.16: $N = 3; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	1
Simulation	0.0	0.0	0.0	
1.17: $N = 3; K = 5; \lambda_D = 0.5$				
Approximation	0.035	2.06	3.4	2
Simulation	0.033 ($\pm$30%)	2.16 ($\pm$17%)	3.1	
1.18: $N = 3; K = 5;$				
$\lambda_D = 0.625$ Approximation	0.222	3.00	11.82	3
Simulation	0.230 ($\pm$17%)	3.26 ($\pm$15%)	11.78	
1.19: $N = 3; K = 5; \lambda_D = 0.8$				
Approximation	4.56	10.1	56.3	4
Simulation	4.26 ($\pm$19%)	10.3 ($\pm$13%)	52.1	
1.20: $N = 5; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	1
Simulation	0.0	0.0	0.0	
1.21: $N = 5; K = 5; \lambda_D = 0.5$				
Approximation	0.0353	2.07	3.40	2
Simulation	0.038 ($\pm$30%)	2.16 ($\pm$9%)	3.58	
1.22: $N = 5; K = 5; \lambda_D = 0.8$				
Approximation	11.26	19.3	73.0	7
Simulation	8.93 ($\pm$22%)	17.2 ($\pm$15%)	65.2	
1.23: $N = 10; K = 5; \lambda_D = 0.1$				
Approximation	0.0	0.0	0.0	1
Simulation	0.0	0.0	0.0	
1.24: $N = 10; K = 5; \lambda_D = 0.5$				
Approximation	0.0353	2.07	3.40	2
Simulation	0.0368 ($\pm$30%)	2.18 ($\pm$17%)	3.38	
1.25: $N = 10; K = 5; \lambda_D = 0.77$				
Approximation	6.89	13.9	64.2	11
Simulation	5.95 ($\pm$22%)	13.7 ($\pm$14%)	56.9	

the conventional kanban control system, whereas it is centralized in the echelon kanban control system. Nonetheless, this does not seem to constitute a noticeable disadvantage of the approximation method for the echelon kanban control system, since for all the cases examined, the method converges in a matter of 1–10 seconds.

Example 2. In Example 2, we consider an echelon kanban control system consisting of N = 3 identical stages, where each stage contains a single machine with

Table 5. Average work in progress (WIP) and average number of finished parts (FP) in each stage for the echelon kanban control system (Example 1)

Configuration	Stage 1		Stage 2		Stage 3	
	WIP	FP	WIP	FP	WIP	FP
1.17: $N = 3; K = 5; \lambda_D = 0.5$						
Simulation	0.988	4.039	0.978	4.022	0.961	4.011
	($\pm$0.1%)	($\pm$0.09%)	($\pm$0.1%)	($\pm$0.1%)	($\pm$0.1%)	($\pm$0.1%)
Approximation	0.999	4.031	0.995	4.005	0.969	4.000
Error	+1.1%	−0.2%	+1.7%	−0.4%	+0.8%	−0.3%
1.19: $N = 3; K = 5; \lambda_D = 0.8$						
Simulation	3.363	2.392	3.068	2.018	2.589	1.569
	($\pm$0.5%)	($\pm$0.3%)	($\pm$0.3%)	($\pm$0.3%)	($\pm$0.3%)	($\pm$0.5%)
Approximation	3.479	2.349	3.159	1.902	2.655	1.455
Error	+3.3%	−1.8%	+2.9%	−6.1%	+2.5%	−7.8%

Table 6. Average work in progress (WIP) and average number of finished products (FP) in each stage for the conventional kanban control system (Example 1)

Configuration	Stage 1		Stage 2		Stage 3	
	WIP	FP	WIP	FP	WIP	FP
1.17: $N = 3; K = 5; \lambda_D = 0.5$						
Simulation	0.94	4.06	0.95	4.02	0.94	4.04
	($\pm$3.2%)	($\pm$0.7%)	($\pm$3.1%)	($\pm$0.7%)	($\pm$3.2%)	($\pm$0.8%)
Approximation	0.97	4.03	0.97	4.01	0.97	4.00
Error	+3%	−0.7%	+2%	−0.2%	+3%	−1%
1.19: $N = 3; K = 5; \lambda_D = 0.8$						
Simulation	2.54	2.47	2.52	1.98	2.55	1.58
	($\pm$3.0%)	($\pm$4.0%)	($\pm$3.2%)	($\pm$5.0%)	($\pm$3.1%)	($\pm$6.3%)
Approximation	2.61	2.38	2.58	1.85	2.66	1.40
Error	+2.7%	−3.6%	+2.4%	−6.5%	+4%	−11%

mean service-time equal to 1. The number of echelon kanbans at each stage is $K_1 = 15, K_2 = 10$, and $K_3 = 5$. Our goal is to investigate the influence of the variability of the service time on the performance of the above system. To this end, we consider three different service-time distributions: a Coxian-2 distribution with squared coefficient of variation $\text{cv}^2 = 2.0$, an Erlang-2 distribution with $\text{cv}^2 = 0.5$, and an exponential distribution with $\text{cv}^2 = 1.0$. Table 7 shows the production capacity for the saturated echelon kanban control system obtained by the approximation method and by simulation, for the three different distributions. Table 8 shows the same results for the conventional kanban control system obtained in [14].

Table 7. Production capacity of the echelon kanban control system (Example 2)

Configuration	Simulation		Approximation		
	Production capacity	Confidence interval	Production capacity	Relative error	Iterations
2.1: $N = 3; K = 5; \text{cv}^2 = 0.5$	0.929	$\pm 0.1\%$	0.934	$+0.5\%$	11
2.2: $N = 3; K = 5; \text{cv}^2 = 1$	0.876	$\pm 0.2\%$	0.873	-0.3%	7
2.3: $N = 3; K = 5; \text{cv}^2 = 2$	0.813	$\pm 0.3\%$	0.808	-0.6%	13

Table 8. Production capacity of the conventional kanban control system (Example 2)

Configuration	Simulation		Approximation		
	Production capacity	Confidence interval	Production capacity	Relative error	Iterations
2.1: $N = 3; K = 5; \text{cv}^2 = 0.5$	0.926	$\pm 0.2\%$	0.932	$+0.6\%$	2
2.2: $N = 3; K = 5; \text{cv}^2 = 1$	0.870	$\pm 0.1\%$	0.865	-0.6%	2
2.3: $N = 3; K = 5; \text{cv}^2 = 2$	0.787	$\pm 0.5\%$	0.786	-0.2%	2

From the results in Table 7, we note that when the variability of the service time distribution increases, the production capacity decreases, as is expected. The results obtained by the approximation method are fairly accurate when compared to the simulation results. Comparing the results between Tables 7 and 8, we note that for all the service-time distributions, the production capacity of the echelon kanban control system is higher than that of the conventional kanban control system. The results for the analytical solution and simulation for the case of the echelon kanban system with backordered demands is shown in Figure 4. More specifically, Figure 4 depicts the proportion of backordered demands, p_B, as a function of the arrival rate of demands, λ_D, for the three different service time distributions. It appears that as the cv^2 of the service time distribution increases, the difference between simulation and analytical results tends to increase.

7.2 Optimization of parameters

The main purpose of developing an approximation method for the performance evaluation of the echelon kanban control system is to use it to optimize the design parameters of the system. The design parameters of the echelon kanban control system are the number of echelon kanbans for each stage. In order to optimize these parameters, we must define a performance measure of the system. Typical performance measures are those that include the cost of not being able to satisfy the demands on time (i.e. quality of service) and the cost of producing parts ahead of

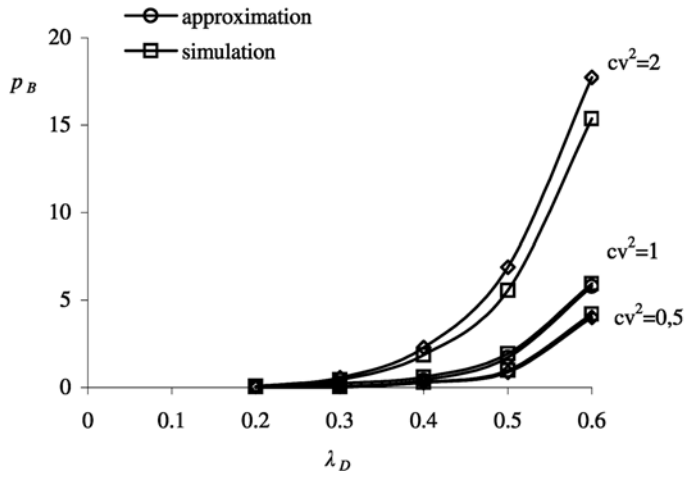


Fig. 4. Proportion of backordered demands versus the average arrival rate of demands for different values of the service-time squared coefficient of variation (Example 2)

time and, therefore, building up inventory (inventory holding cost). In this paper, we consider an optimization problem where the objective is to meet a certain quality of service constraint with minimum inventory holding cost.

We examine two quality-of-service measures as in [15]. The first measure is the probability that when a customer demand arrives, it is backordered. The second measure is the probability that when a customer demand arrives, it sees more than $n$waiting demands, excluding itself. The first measure is denoted by P_{rupt} and concerns the situation where the demands must be immediately satisfied. The second measure is denoted by $P(Q > n)$ and concerns the situation where we have the prerogative to introduce a delay in filling orders, which is equivalent to authorizing demands to wait. Specifically, P_{rupt} is the marginal stationary probability of having no finished parts in the last synchronization station, which is given by equation (18) in Appendix A. Similarly, $P(Q > n)$ is the stationary probability of having more than n customers waiting and can be computed from the following expression:

$$P(Q > n) = \sum_{x=n+1}^{\infty} P(Q = x) = 1 - \sum_{y=0}^{n} P(Q = y), \tag{10}$$

where $P(Q = n)$ is given by (see Appendix A):

$$P(Q = n) = p_O^N(0, n) = p_O^N(0, 0) \left(\frac{\lambda_D}{\lambda_O^N(0)} \right)^n. \tag{11}$$

The stationary distribution $p_O^N(0, 0)$ that is needed to evaluate both P_{rupt} and $P(Q > n)$ is given by the following expression:

$$p_O^N(0, 0) = \frac{1}{\frac{1}{1 - \frac{\lambda_D}{\lambda_O^N(0)}} + \sum_{x=1}^{K_N} \left(\frac{1}{\lambda_D^x} \prod_{i=0}^{x-1} \lambda_O^N(i) \right)}. \tag{12}$$

The cost function that we want to minimize is the long-run, expected, average cost of holding inventory,

$$C_{total} = \sum_{i=1}^{N} h_i E\left[WIP_i + FP_i\right], \tag{13}$$

where h_i is the unit cost of holding WIP_i + FP_i inventory per unit time in stage i.

In the remaining of this section, we optimize the echelon kanbans of an echelon kanban control system consisting of $N = 5$ stages, where each stage contains a single machine with exponentially distributed service times with mean equal to 1, for different combinations of inventory holding cost rates, h_i, $i = 1, \ldots, 5$, and demand arrival rate $\lambda_D = 0.5$. In all cases we assume that there is value added to the parts at every stage so that the inventory holding cost increases as the stage increases, i.e. $h_1 < h_2 < \ldots < h_5$. If this were not the case, i.e. if $h_1 = h_2 = \ldots = h_5$, then clearly it would make no sense to block the passage of parts from one stage to another via the use of echelon kanbans, because this would not lower the inventory holding cost but would worsen the quality of service. This implies that if $h_1 = h_2 = \ldots = h_5$, the optimal echelon kanbans satisfy $K_1 \leq K_i$, $i = 2, \ldots, 5$, in which case the echelon kanban control system is equivalent to the make-to-stock CONWIP system [23] with a WIP-cap on the total number of parts in the system equal to K_1.

Table 9 shows the optimal design parameters $(K_1, \ldots, K_5)$ and associated minimum, long-run, expected, average cost of holding inventory, for $\lambda_D = 0.5$ and different quality of service constraints and inventory holding cost rates $h_1, \ldots, h_5$, where $h_1 < h_2 < \ldots < h_5$. The quality of service constraints that we use are $P_{rupt} \leq 0.02$ and $P(Q > n) \leq 0.02$, for $n = 2, 5, 10$.

From the results in Table 9, we see that the higher the number of backordered demands n in the quality of service definition, $P(Q > n)$, the lower the optimal number of echelon kanbans, and hence the inventory holding cost. As the difference between the holding cost rates $h_i, i = 1, \ldots, 5$, increases, the difference between the optimal values of $K_i, i = 1, \ldots, 5$, also increases, since the behavior of the echelon kanban control system diverts further from that of the make-to-stock CONWIP system. When the relative difference between the holding cost rates $h_i, i = 1, \ldots, 5$, is low, the behavior of the echelon kanban control system tends to that of the make-to-stock CONWIP system.

Table 10 shows the optimal design parameter K_1 and associated minimum inventory holding cost for $\lambda_D = 0.5$ and different quality of service constraints and inventory holding cost rates $h_1, \ldots, h_5$, for the make-to-stock CONWIP system. The last column of Table 10 shows the relative increase in cost of the optimal make-to-stock CONWIP system compared to the optimal echelon kanban control system. Comparing the results between Tables 9 and 10, we note that the optimal make-to-stock CONWIP system performs considerably worse than the optimal echelon kanban control system, particularly when the relative difference between the holding cost rates h_i, $i = 1, \ldots, 5$, is high and/or the number of backordered demands n in the quality of service definition, $P(Q > n)$, is high, indicating that the quality of service is low.

Table 9. Opimal configuration and associated costss for $\lambda_D = 0.5$ and different values of $h_1, \ldots, h_5$, for the echelon kanban control system

Design criterion	K_1	K_2	K_3	K_4	K_5	Cost
$h_1 = 1, h_2 = 2, h_3 = 3, h_4 = 4, h_5 = 5$						
$P_{rupt} \leq 0.02$	15	13	12	10	8	55.885
$P(Q > 2) \leq 0.02$	13	11	10	8	7	46.555
$P(Q > 5) \leq 0.02$	10	8	7	6	2	31.120
$P(Q > 10) \leq 0.02$	7	6	5	3	1	20.253
$h_1 = 3, h_2 = 8, h_3 = 9, h_4 = 10, h_5 = 12$						
$P_{rupt} \leq 0.02$	15	13	12	10	8	144.314
$P(Q > 2) \leq 0.02$	13	11	10	9	6	121.161
$P(Q > 5) \leq 0.02$	10	8	7	6	2	84.074
$P(Q > 10) \leq 0.02$	7	6	5	3	1	57.360
$h_1 = 1, h_2 = 2, h_3 = 4, h_4 = 11, h_5 = 12$						
$P_{rupt} \leq 0.02$	15	14	13	9	8	121.288
$P(Q > 2) \leq 0.02$	14	13	10	7	6	98.890
$P(Q > 5) \leq 0.02$	10	9	8	5	2	67.383
$P(Q > 10) \leq 0.02$	8	6	4	3	1	39.483
$h_1 = 1, h_2 = 6, h_3 = 11, h_4 = 16, h_5 = 21$						
$P_{rupt} \leq 0.02$	17	13	11	10	8	218.702
$P(Q > 2) \leq 0.02$	15	11	10	8	5	178.162
$P(Q > 5) \leq 0.02$	10	8	7	6	2	115.601
$P(Q > 10) \leq 0.02$	8	6	5	3	1	76.523
$h_1 = 1, h_2 = 11, h_3 = 21, h_4 = 31, h_5 = 41$						
$P_{rupt} \leq 0.02$	17	13	11	10	8	420.405
$P(Q > 2) \leq 0.02$	15	11	10	8	5	341.324
$P(Q > 5) \leq 0.02$	10	8	7	6	2	221.203
$P(Q > 10) \leq 0.02$	8	6	5	3	1	145.047
$h_1 = 1, h_2 = 2, h_3 = 4, h_4 = 8, h_5 = 16$						
$P_{rupt} \leq 0.02$	17	15	12	9	7	143.879
$P(Q > 2) \leq 0.02$	14	13	11	7	5	112.442
$P(Q > 5) \leq 0.02$	10	8	7	6	2	65.843
$P(Q > 10) \leq 0.02$	8	6	5	3	1	39.934
$h_1 = 1, h_2 = 3, h_3 = 9, h_4 = 27, h_5 = 81$						
$P_{rupt} \leq 0.02$	19	17	14	10	6	633.178
$P(Q > 2) \leq 0.02$	17	15	12	8	4	471.867
$P(Q > 5) \leq 0.02$	12	10	8	6	1	231.446
$P(Q > 10) \leq 0.02$	8	6	5	3	1	139.066

Table 10. Optimal configuration and associated costs for $\lambda_D = 0.5$ and different values of $h_1, \ldots, h_5$, for the CONWIP system

Design criterion	K_1	Cost	Relative cost increase
$h_1 = 1, h_2 = 6, h_3 = 11, h_4 = 16, h_5 = 21$			
$P_{rupt} \leq 0.02$	14	244.163	10.43%
$P(Q > 2) \leq 0.02$	12	202.415	11.98%
$P(Q > 5) \leq 0.02$	10	161.006	28.2%
$P(Q > 10) \leq 0.02$	8	120.307	36.39%
$h_1 = 1, h_2 = 11, h_3 = 21, h_4 = 31, h_5 = 41$			
$P_{rupt} \leq 0.02$	14	474.326	11.37%
$P(Q > 2) \leq 0.02$	12	392.830	13.11%
$P(Q > 5) \leq 0.02$	10	312.012	29.1%
$P(Q > 10) \leq 0.02$	8	232.613	37.64%
$h_1 = 1, h_2 = 2, h_3 = 4, h_4 = 8, h_5 = 16$			
$P_{rupt} \leq 0.02$	14	175.160	17.86%
$P(Q > 2) \leq 0.02$	12	143.407	21.59%
$P(Q > 5) \leq 0.02$	10	111.986	41.2%
$P(Q > 10) \leq 0.02$	8	81.260	50.86%
$h_1 = 1, h_2 = 3, h_3 = 9, h_4 = 27, h_5 = 81$			
$P_{rupt} \leq 0.02$	14	850.927	25.59%
$P(Q > 2) \leq 0.02$	12	690.358	31.65%
$P(Q > 5) \leq 0.02$	10	531.715	56.47%
$P(Q > 10) \leq 0.02$	8	377.102	63.12%

8 Conclusions

We developed an analytical, decomposition-based approximation method for the performance evaluation of the echelon kanban control system and tested it on several numerical examples. The numerical examples showed that the method is quite accurate in most cases. They also showed that the echelon kanban control system has some advantages over the conventional kanban control system. Specifically, when the two systems have the same value of K, the echelon kanban control system has higher production capacity, lower average number of backordered demands, but only slightly higher average WIP and either slightly higher or slightly lower FP than the conventional kanban control system. The numerical results also showed that as the variability of the service time distribution increases, the production capacity of the echelon kanban control system and the accuracy of the approximation method decrease. Finally, we know that the optimized echelon kanban control system always performs at least as well as the optimized make-to-stock CONWIP system since the latter system is a special case of the first system. The numerical results showed that in fact the superiority in performance of the echelon kanban control system over that of the make-to-stock CONWIP system can be quite significant, particularly when the relative increase in inventory holding costs from one stage to the next downstream stage is high and/or the quality of service is low.

Appendix A – Analysis of synchronization station O^N

O^N is a synchronization station fed by a continuous-time Markov arrival process with state-dependent arrival rate $\lambda_O^N(n_O^N), 0 \leq n_O^N < K_N$, and an external Poisson process with rate λ_D. The underlying continuous-time Markov chain is shown in Figure 5. The state of this Markov chain is (n_O^N, n_D), where is the number of engaged kanbans and n_D, $n_D \geq 0$, is the number of external resources (customer demands) currently present in subsystem O^N. Let $p_O^N(n_O^N, n_D)$ be the steady-state probabilities of the Markov chain. These probabilities are solution of the following balance equations:

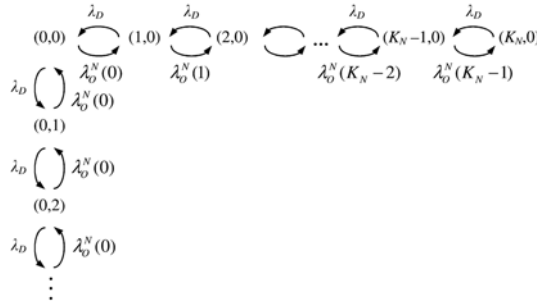


Fig. 5. Continuous-time Markov chain describing the state (n_0^N, n_D) of synchronization station O^N

$$p_O^N(n_O^N, 0)\lambda_D = p_O^N(n_O^N - 1, 0)\lambda_O^N(n_O^N - 1) \qquad \text{for } n_O^N = 1, ..., K_N \tag{14}$$

$$p_O^N(0, n_D)\lambda_O^N(0) = p_O^N(0, n_D - 1)\lambda_D \qquad \text{for } n_D > 0 \tag{15}$$

The marginal probabilities $P_O^N(n_O^N)$ are then simply given by

$$P_O^N(n_O^N) = p_O^N(n_O^N, 0) \qquad \text{for } n_O^N = 1, ..., K_N, \tag{16}$$

$$P_O^N(0) = \sum_{n_D=0}^{\infty} p_O^N(0, n_D). \tag{17}$$

From (15) and (17) we get

$$P_O^N(0) = \sum_{n_D=0}^{\infty} p_O^N(0,0) \left(\frac{\lambda_D}{\lambda_O^N(0)} \right)^{n_D} = p_O^N(0,0) \frac{1}{1 - \frac{\lambda_D}{\lambda_O^N(0)}}. \tag{18}$$

The conditional throughputs of subsystem O^N are obtained from (5), (14) and (16), as follows:

$$v_O^N(n_O^N) = \lambda_D \qquad \text{for } n_O^N = 2, ..., K_N \tag{19}$$

From (5), (14), (16) and (18), we also get

$$v_O^N(1) = \frac{1}{\frac{1}{\lambda_D} - \frac{1}{\lambda_O^N(0)}}. \tag{20}$$

Appendix B – Analysis of synchronization station $\mathbf{I}^i$

I^i, $i = 2, \ldots, N$, is a synchronization station fed by two continuous-time Markov arrival processes with state-dependent arrival rates: $\lambda_I^i(n_I^i)$, $0 \leq n_I^i \leq K_i$, and $\lambda^i(n^i)$, $0 \leq n^i \leq K_{i-1}$. The underlying continuous-time Markov chain is shown in Figure 6. The state of this Markov chain is (n_I^i, n_u^i), where n_I^i is the number of free kanbans and n_u^i is the number of external resources (finished parts of stage $i-1$) currently present in subsystem I^i. Recall that n^i can be obtained from n_u^i and n_I^i using (3). The steady-state probabilities $p_I^i(n_I^i, n_u^i)$ can be derived as solutions of the underlying balance equations and are given by:

$$p_I^i(n_I^i, 0) = \left[\prod_{n=1}^{n_I^i} \frac{\lambda_I^i(n-1)}{\lambda^i(K_i - n)} \right] p_I^i(0,0), \tag{21}$$

$$p_I^i(0, n_u^i) = \frac{\prod_{n=1}^{n_u^i} \lambda^i(K_i + n - 1)}{\left[\lambda_I^i(0)\right]^{n_u^i}} p_I^i(0,0). \tag{22}$$

The marginal probabilities, $P_I^i(n_I^i)$, can then be derived by summing up the probabilities above as follows:

$$P_I^i(n_I^i) = \left[\prod_{n=1}^{n_I^i} \frac{\lambda_I^i(n-1)}{\lambda^i(K_i - n)} \right] p_I^i(0,0) \quad \text{for} \quad n_I^i = 1, \ldots, K_i, \tag{23}$$

$$P_I^i(0) = \left[1 + \sum_{n_u^i=1}^{K_{i-1}-K_i} \frac{\prod_{n=1}^{n_u^i} \lambda^i(K_i + n - 1)}{\left[\lambda_I^i(0)\right]^{n_u^i}} \right] p_I^i(0,0). \tag{24}$$

The estimation of the conditional throughputs of subsystem I^i can then be obtained by substituting the above probabilities into (5), as follows:

$$v_I^i(n_I^i) = \lambda^i(K_i - n_I^i) \qquad \text{for } n_I^i = 2, ..., K_i, \tag{25}$$

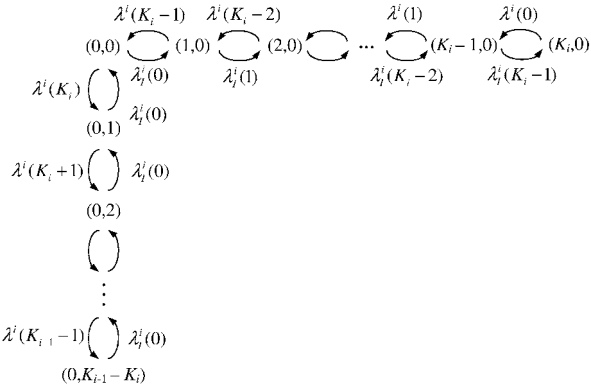


Fig. 6. Continuous-time Markov chain describing the state (n_I^i, n_u^i) of synchronization station I^i

$$v_I^i(1) = \lambda^i(K_i - 1)\left[1 + \sum_{n_u^i=1}^{K_{i-1}-K_i} \frac{\prod_{n=1}^{n_u^i} \lambda^i(K_i + n - 1)}{\left[\lambda_I^i(0)\right]^{n_u^i}}\right]. \tag{26}$$

Appendix C – Table of notation

N	Number of stages
K_i	Number of echelon kanbans of stage i
L_i	Subnetwork associated with the manufacturing process of stage i
m_i	Number of stations of subnetwork L_i
J_i	Synchronization station at the output of stage i
λ_D	Average arrival rate of external customer demands in the unsaturated system
P_r	Maximum rate at which customer demands can be satisfied
R	Queueing network of the echelon kanban control system
R^i	Subsystem associated with stage i
I^i	Upstream synchronization station of subsystem R^i
O^N	Downstream synchronization station of subsystem R^N
$\hat{S}_i$	Downstream single-server pseudo-station of subsystem R^i
n^i	State of subsystem R^i
$\lambda^i(n^i)$	State-dependent arrival rate of stage-i raw parts at the upstream synchronization station I^i of subsystem R^i
$v^i(n^i)$	Conditional throughput of subsystem R^i
$k \in M_i$	Index denoting the stations within subsystem R^i, where $M_1 = \{1, \ldots, m_1, \hat{S}\}$, $M_i = \{I, 1, \ldots, m_i, \hat{S}\}$ for $i = 2, \ldots, N-1$, and $M_N = \{I, 1, \ldots, m_N, O\}$
n_k^i	State of station k in subsystem R^i
$\mu_k^i(n_k^i)$	Load-dependent service rate of station k in subsystem R^i
$\mu_k(n_k)$	Same as $\mu_k^i(n_k^i)$ with index i dropped
T_k^i	Open system representing station k in subsystem R^i
T_k	Same as T_k^i with index i dropped
$\lambda_k^i(n_k^i)$	Rate of state-dependent Poisson arrival process at T_k^i
$\lambda_k(n_k)$	Same as $\lambda_k^i(n_k^i)$ with index i dropped
$v_k^i(n_k^i)$	Conditional throughput of T_k^i
$v_k(n_k)$	Same as $v_k^i(n_k^i)$ with index i dropped
$P_k^i(n_k^i)$	Steady-state probability of T_k^i
p_B	Proportion of backordered demands
Q_D	Average number of backordered demands
W_B	Average waiting time of backordered demands

References

1. Baskett F, Chandy KM, Muntz RR, Palacios-Gomez F (1975) Open, closed and mixed networks of queues with different classes of customers. Journal of ACM 22: 248–260
2. Baynat B, Dallery Y (1993) A unified view of product-form approximation techniques for general closed queueing networks. Performance Evaluation 18(3): 205–224
3. Baynat B, Dallery Y (1993) Approximate techniques for general closed queueing networks with subnetworks having population constraints. European Journal of Operational Research 69: 250–264
4. Baynat B, Dallery Y (1996) A product-form approximation method for general closed queueing networks with several classes of customers. Performance Evaluation 24: 165–188
5. Baynat B, Dallery Y, Ross K (1994) A decomposition approximation method for multiclass BCMP queueing networks with multiple-server stations. Annals of Operations Research 48: 273–294
6. Bruell SC, Balbo G (1980) Computational algorithms for closed queueing networks. Elsevier North-Holland, Amsterdam
7. Buzacott JA (1989) Queueing models of kanban and MRP controlled production systems. Engineering Costs and Production Economics 17: 3–20
8. Buzacott JA, Shanthikumar JG (1993) Stochastic models of manufacturing systems. Prentice-Hall, Englewood Cliffs, NJ
9. Buzen JP (1973) Computational algorithms for closed queueing networks with exponential servers. Comm. ACM 16(9): 527–531
10. Clark A, Scarf H (1960) Optimal policies for a multi-echelon inventory problem. Management Science 6: 475–490
11. Dallery Y (1990) Approximate analysis of general open queueing networks with restricted capacity. Performance Evaluation 11(3): 209–222
12. Dallery Y, Cao X (1992) Operational analysis of stochastic closed queueing networks. Performance Evaluation 14(1): 43–61
13. Dallery Y, Liberopoulos G (2000) Extended kanban control system: combining kanban and base stock. IIE Transactions 32(4): 369–386
14. Di Mascolo M, Frein Y, Dallery Y (1996) An analytical method for performance evaluation of kanban controlled production systems. Operations Research 44(1): 50–64
15. Duri C, Frein Y, Di Mascolo M (2000) Comparison among three pull control policies: kanban, base stock and generalized kanban. Annals of Operations Research 93: 41–69
16. Frein Y, Di Mascolo M, Dallery Y (1995) On the design of generalized kanban control systems. International Journal of Operations and Production Management 15(9): 158–184
17. Gordon WJ, Newell GF (1967) Closed queueing networks with exponential servers. Operations Research 15: 252–267
18. Jackson JR (1963) Jobshop-like queueing systems. Management Science 10(1): 131–142
19. Liberopoulos G, Dallery Y (2002) Comparative modeling of multi-stage production-inventory control policies with lot sizing. International Journal of Production Research 41(6): 1273–1298
20. Marie R (1979) An approximate analytical method for general queueing networks. IEEE Transactions on Software Engineering 5(5): 530–538
21. Marie R (1980) Calculating equilibrium probabilities for $\lambda(n)/C_k/1/N$ queues. Performance Evaluation Review 9: 117–125
22. Reiser M, Lavenberg SS (1980) Mean value analysis of closed multichain queueing networks. Journal of ACM 27(2): 313–322

23. Schweitzer PJ (1979) Approximate analysis of multiclass closed networks of queues. Proceedings of the International Conference on Stochastic Control and Optimization, Amsterdam
24. Spanjers L, van Ommeren JCW, Zijm WHM (2005) Closed loop two-echelon reparable item systems. OR Spectrum 27(2–3): 369–398
25. Spearman ML, Woodruff DL, Hopp WJ (1990) CONWIP: a pull alternative to kanban. International Journal of Production Research 28: 879–894
26. Stewart WJ, Marie R (1980) A numerical solution for the λ(n)/C_k/r/N queue. European Journal of Operational Research 5: 56–68
27. Whitt W (1983) The queueing network analyser. Bell Systems Technology Journal 62(9): 2779–2815

Closed loop two-echelon repairable item systems

L. Spanjers, J.C.W. van Ommeren, and W.H.M. Zijm

Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
(e-mail: w.h.m.zijm@utwente.nl)

Abstract. In this paper we consider closed loop two-echelon repairable item systems with repair facilities both at a number of local service centers (called bases) and at a central location (the depot). The goal of the system is to maintain a number of production facilities (one at each base) in optimal operational condition. Each production facility consists of a number of identical machines which may fail incidentally. Each repair facility may be considered to be a multi-server station, while any transport from the depot to the bases is modeled as an ample server. At all bases as well as at the depot, ready-for-use spare parts (machines) are kept in stock. Once a machine in the production cell of a certain base fails, it is replaced by a ready-for-use machine from that base's stock, if available. The failed machine is either repaired at the base or repaired at the central repair facility. In the case of local repair, the machine is added to the local spare parts stock as a ready-for-use machine after repair. If a repair at the depot is needed, the base orders a machine from the central spare parts stock to replenish its local stock, while the failed machine is added to the central stock after repair. Orders are satisfied on a first-come-first-served basis while any requirement that cannot be satisfied immediately either at the bases or at the depot is backlogged. In case of a backlog at a certain base, that base's production cell performs worse.

To determine the steady state probabilities of the system, we develop a slightly aggregated system model and propose a special near-product-form solution that provides excellent approximations of relevant performance measures. The depot repair shop is modeled as a server with state-dependent service rates, of which the parameters follow from an application of Norton's theorem for Closed Queuing Networks. A special adaptation to a general Multi-Class Marginal Distribution Analysis (MDA) algorithm is proposed, on which the approximations are based. All relevant performance measures can be calculated with errors which are generally

Correspondence to: W.H.M. Zijm

less than one percent, when compared to simulation results. The approximations are used to find the stock levels which maximize the availibility given a fixed configuration of machines and servers and a certain budget for storing items.

Keywords: Multi-echelon systems – Repairable items – Spare parts inventory – Closed queueing networks – Near-product form solutions

1 Introduction

Repairable inventory theory involves designing inventory systems for items which are repaired and returned to use rather than discarded. The items are less expensive to repair than to replace. Such items can for example be found in the military, aviation, copying machines, transportation equipment and electronics. The repairable inventory problem is typically concerned with the optimal stocking of parts at bases and a central depot facility which repairs failed units returned from bases while providing some predetermined level of service. Different performance measures may be used, such as cost, backorders and availability.

Over the past 30 years there has been considerable interest in multi-echelon inventory theory. Much of this work originates from a model called METRIC, which was first reported in the literature by Sherbrooke [9]. The model was developed for the US Air Force at the Rand Corporation for a multi-echelon repairable-item inventory system. In this model an item at failure is replaced by a spare if one is available. If none are available a spare is backordered. Of the failed items a certain proportion is repaired at the base and the rest at a repair depot, thereby creating a two-echelon repairable-item system. Items are returned from the depot using a one-for-one reordering policy. The METRIC model determines the optimal level of spares to be maintained at each of the bases and at the depot.

A shortfall of the METRIC model is that it assumes that failures are Poisson from an infinite source and that the repair capacity is unlimited. Therefore, others have continued the research to gain results more useful for real life applications. Gross, Kioussis and Miller [5], Albright and Soni [1] and Albright [2] focused their attention on closed queuing network models, thereby dropping the assumption of Poisson failures from an infinite source. The intensity by which machines enter the repair shops depends on the number of machines operating in the production cell. In case of a backlog at a base, this intensity is therefore smaller than in the optimal case where the maximum number of machines is operating in the production cell. Also the assumption of unlimited repair capacity is dropped in Gross et al. [5] and Albright [2].

This paper deals with similar models. It handles closed queuing network models with limited repair. However, the approximation method differs considerably. The approximation method builds on the method by Avsar and Zijm [3]. Avsar and Zijm considered an open queuing network model with limited repair. By a small aggregation step, the system is changed into a system with a special near-product-form solution that provides an approximation for the steady state distribution. From the steady state distribution all relevant performance measures can be computed.

We will perform a similar aggregation step in this paper and again a special near-product-form solution will be obtained. However, as opposed to open systems, in a system with finite sources, the demand rates to the depot also become state dependent; moreover, these demand rates are clearly influenced by the efficiency of the base repair stations. Nevertheless, we are able to develop relatively simple approximation algorithms to obtain the relevant performance measures. These performance measures can ultimately be used within an optimization model to determine such quantities as the optimal repair capacities and the optimal inventory levels.

The organization of this paper is as follows: In the next section we consider a very simple two-echelon system, consisting of one base, a base repair shop and a central repair shop. The repair shops are modeled as single servers. This model mainly serves to explain the essential elements of the aggregation step. We present the modified system with near-product-form solution and numerical results to show the accuracy of the approximation. Next, in Section 3, we turn to more general repairable item network structures, containing multiple bases and transport lines from the depot to the bases. The repair shops are modeled as multi-servers. The approximation method leading to an adapted Multi-Class MDA algorithm is presented and some numerical results are discussed. In Section 4, an optimization algorithm based on this approximation method, is given which finds the stock levels that maximize the (weighted) availibility under a given cost constraint. In the last section, we summarize our results and discuss a number of extensions that are currently being investigated.

2 Analysis of a simple two-echelon system with single server facilities

In this section a simplified repairable item system is discussed to explain how a slight modification turns this system into a near-product form network that can be analyzed completely. In the next section we turn to more complex systems.

2.1 The single base model without transportation

Consider the system as depicted in Figure 1. The system consists of a single base and a depot. At the base a maximum of J_1 machines can be operational in the production cell.

Operational machines fail at exponential rate λ_1 and are replaced by a machine from the base stock (if available). Both at the base and at the depot there is a repair shop. Failed machines are base-repairable with probability p_1 and consequently depot-repairable with probability $1 - p_1$. The repair shops are modeled as single servers with exponential service rate μ_0 for the depot and exponential service rate μ_1 for the base. In addition to the J_1 machines another group of S_1 machines is dedicated to the base to act as spares. When a machine fails, the failed machine goes to a repair shop while at the same time a request is sent to place a spare machine from the base stock in the production cell. This request is carried out immediately, if possible. In case no spare machines are at the base, a backlog occurs. As soon as there is a repaired machine available, it becomes operational. A number of S_0

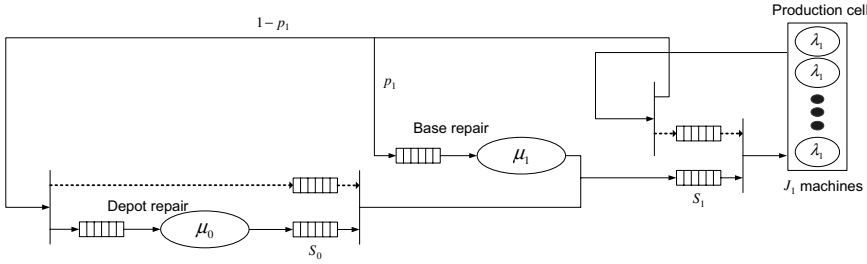


Fig. 1. The single base repairable item system

machines is dedicated to the depot to act as spares. When a failed machine cannot be repaired at the base and hence is sent to the depot, a spare machine is shipped from the depot to the base to replenish the base stock, or - in case of a backlog - to become operational immediately. When no spares are available at the depot, a backorder is created. In that case, as soon as a machine is repaired at the depot repair shop, it is sent to the base. In this simple model, transport times from the base to the depot and vice versa are not taken into account.

In Figure 1 (and subsequent figures), requests are indicated by dotted lines. The matching of a request and a ready-for-use machine is modeled as a synchronization queue, both at the base and at the depot. At the base however, some reflection reveals that the synchronization queue can be seen as a normal queue where machines are waiting to be moved into the production cell. This is only possible when the production cell does not contain the maximum number of machines, that is, if a machine in the production cell has failed. This leads to the model in Figure 2.

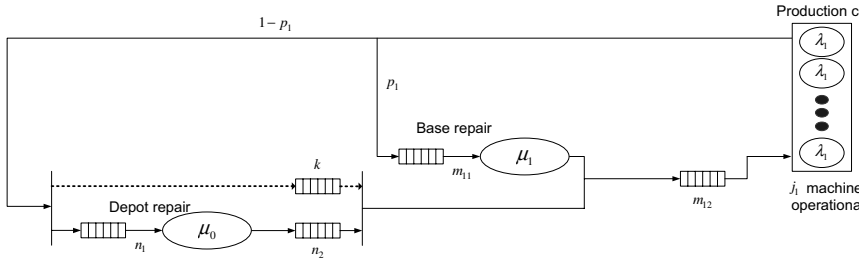


Fig. 2. The modified single base system

In this figure the variables n_1, n_2, k, m_{11} and m_{12} indicate the lengths of the various queues in the system. The number of machines in (or awaiting) depot repair is denoted by the random variable $\underline{n}_1$, the number of spare machines at the depot is denoted by the random variable $\underline{n}_2$ and the backlog of machines at the depot is denoted by $\underline{k}$. At the base there are $\underline{m}_{11}$ machines waiting for repair or being repaired and $\underline{m}_{12}$ machines are acting as spares. In the production cell $\underline{j}_1$ machines are operational. As a result of the operating inventory control policies, for $\underline{n}_1 = n_1$, $\underline{n}_2 = n_2$, $\underline{k} = k$, $\underline{m}_{11} = m_{11}$, $\underline{m}_{12} = m_{12}$ and $\underline{j}_1 = j_1$ the following equations

must hold:

$$n_1 + n_2 - k = S_0, \tag{1}$$

$$n_2 \cdot k = 0, \tag{2}$$

$$k + m_{11} + m_{12} + j_1 = S_1 + J_1, \tag{3}$$

$$m_{12} \cdot (J_1 - j_1) = 0, \tag{4}$$

where Equations (2) and (4) follow from the fact that it is impossible to have a backlog and to have spare machines available at the same time. If spare machines are available, a request is satisfied immediately. In case of a backlog, a request is not satisfied until a repair completion. The repaired machine is merged with the longest waiting request.

From these relations it follows immediately that n_1 and m_{11} completely determine the state of the system, including the values of n_2, k, m_{12} and j_1. Therefore, the system can be modeled as a continuous time Markov chain with state description (n_1, m_{11}). The corresponding transition diagram is displayed in Figure 3.

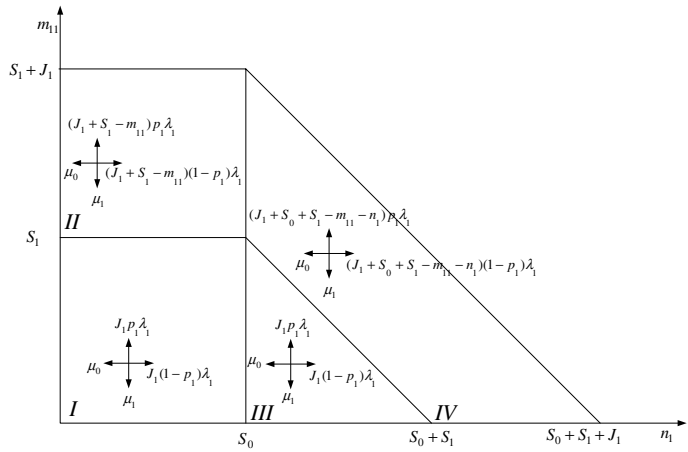


Fig. 3. Transition diagram for state description (n_1, m_{11})

Let $P(n_1, m_{11}) = P(\underline{n}_1 = n_1, \underline{m}_{11} = m_{11})$ be the steady state probability of being in state (n_1, m_{11}). This steady state probability can be found by solving the global balance equations of the system. These can be deduced from the transition diagram. Nevertheless, it is not possible to find an algebraic expression for the steady state probabilities. Moreover, for larger systems with e.g. multiple bases, the computational effort becomes prohibitive. Therefore the system will be slightly adjusted in the next subsection, in order to arrive at a near-product form network. Note that the analysis presented in this paper, is partly similar to the one given in Avsar and Zijm [3], where the equivalent open two-echelon network is considered. For this open network, an algebraic and easily computable product form approximation is found. In the current paper, a closed network is considered, and an easily computable algebraic approximation could not be found. However, the aggregated

network has a product form steady state distribution, and we can use MDA-like algorithms to find numerical approximations for performance measures.

An alternative approach is to model the number of machines at the depot and the bases as a level dependent quasi birth death process. This method may yield an algebraic solution but, here too, the finite state space makes the analysis more complex. Moreover, the transition rates in a given state, do not only depend on the phase but also on the level. Together, this makes the alternative method computationally very demanding, if not intractable.

2.2 Approximation

A first step towards an approximation for the steady state probabilities is to aggregate the state space. The most difficult parts of the transition diagram are regions I and II, that is, the parts with $n_1 \leq S_0$ or, equivalently, the parts with $k = 0$. The parts with $k > 0$ are equivalent to the states with $n_1 = k + S_0$. A natural aggregation of the system is a description through the states (k, m_{11}). The states (n_1, m_{11}) with $n_1 = 0, 1, \ldots, S_0$ are then aggregated into one state $(0, m_{11})$. Denote the steady state probabilities for the new model by $\tilde{P}$ then the following holds for any m_{11}:

$$\tilde{P}(\underline{k} = 0, \underline{m}_{11} = m_{11}) = \sum_{n_1=0}^{S_0} P(\underline{n}_1 = n_1, \underline{m}_{11} = m_{11}), \tag{5}$$

$$\tilde{P}(\underline{k} = k, \underline{m}_{11} = m_{11}) = P(\underline{n}_1 = S_0 + k, \underline{m}_{11} = m_{11}). \tag{6}$$

The transition diagram corresponding to the alternative state space description is displayed in Figure 4.

The rates only differ from the transition diagram in Figure 3 for the case $k = 0$. Let $q(m_{11})$ be the steady state probability that an arriving request for a machine at the depot has to wait, given that it finds no other waiting requests in front of it ($k = 0$) and $\underline{m}_{11} = m_{11}$. Given the (aggregated) state $(0, m_{11})$, the state does not change in case of an arriving request with probability $1-q(m_{11})$, because spares are available. With probability $q(m_{11})$ no spares are available and the state changes into $(1, m_{11})$. The transition rate from $(0, m_{11})$ to $(1, m_{11})$ equals $j_1(1-p_1)\lambda_1 q(m_{11})$. To determine $q(m_{11})$ one needs

$$q(m_{11}) = P(\underline{n}_1 = S_0 | \underline{n}_1 \leq S_0, \underline{m}_{11} = m_{11}). \tag{7}$$

However, to compute this, one needs to know the steady state distribution of the original system, which is exactly what we attempt to approximate. Therefore, we approximate the $q(m_{11})$'s by their weighted average, i.e. we focus on the conditional probability q defined by

$$q = \sum_{m_{11}} q(m_{11}) P(\underline{m}_{11} = m_{11} | \underline{n}_1 \leq S_0) = P(\underline{n}_1 = S_0 | \underline{n}_1 \leq S_0) \tag{8}$$

and for every m_{11} we replace $q(m_{11})$ in the transition diagram by this q. In the next section it will be explained how a reasonable approximation for this q can be obtained by means of an application of Norton's theorem.

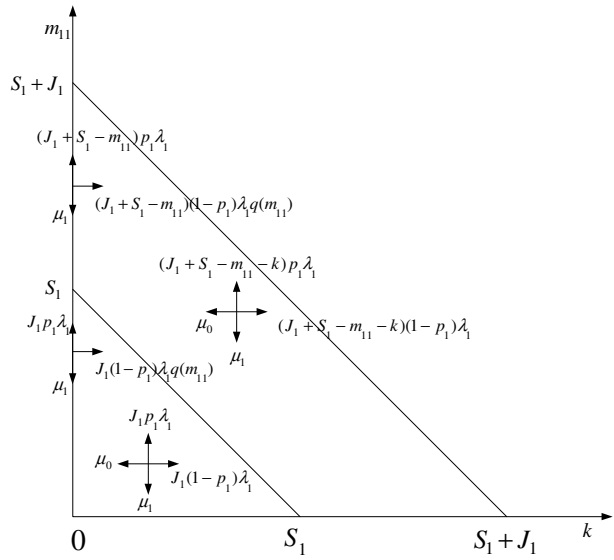


Fig. 4. Transition diagram for state description (k, m_{11})

Lemma 1 *The steady state probabilities for the model with state description* (k, m_{11}) *and transition rates as denoted in Figure 4 with* $q(m_{11})$ *replaced by arbitrary* q *have a product form.*

Proof. To find the steady state probabilities, consider both the original model in Figure 2 and the alternative model in Figure 5.

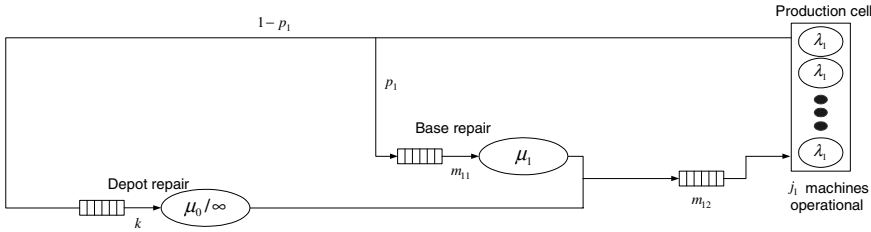


Fig. 5. Typical-server Closed Queuing Network (TCQN)

In Figure 5 the depot repair shop with synchronization queue is replaced by a typical server. For jobs that find the server idle the server has infinite service rate with probability $1 - q$ (the case spares are available) and service rate μ_0 with probability q (the case no spares are available). Let $\underline{b}_1$ be the random variable equal to $\underline{m}_{12} + \underline{j}_1$, then by looking at the system with the typical server, and conditioning on the fact that the network contains exactly $J_1 + S_1$ jobs, it is easily verified that the following expression for $\tilde{P}(\underline{k} = k, \underline{m}_{11} = m_{11}, \underline{b}_1 = b_1)$ satisfies the balance

equations of the TCQN:

$$\tilde{P}(k,m_{11},b_1)=\begin{cases}\tilde{G}q\left(\dfrac{p_1}{\mu_1}\right)^{m_{11}}\left(\dfrac{1-p_1}{\mu_0}\right)^{k}\dfrac{\left(\frac{1}{\lambda_1}\right)^{b_1}}{J_1!J_1^{b_1-J_1}}, & b_1>J_1,k>0\\ \tilde{G}q\left(\dfrac{p_1}{\mu_1}\right)^{m_{11}}\left(\dfrac{1-p_1}{\mu_0}\right)^{k}\dfrac{\left(\frac{1}{\lambda_1}\right)^{b_1}}{b_1!}, & b_1\le J_1,k>0\\ \tilde{G}\left(\dfrac{p_1}{\mu_1}\right)^{m_{11}}\dfrac{\left(\frac{1}{\lambda_1}\right)^{b_1}}{J_1!J_1^{b_1-J_1}}, & b_1>J_1,k=0\\ \tilde{G}\left(\dfrac{p_1}{\mu_1}\right)^{m_{11}}\dfrac{\left(\frac{1}{\lambda_1}\right)^{b_1}}{b_1!}, & b_1\le J_1,k=0\end{cases} \tag{9}$$

with $k+m_{11}+b_1=J_1+S_1$ and $\tilde{G}$ the normalization constant. □

Expressed in terms of the state variables (k,m_{11}), this result immediately leads to:

Lemma 2 *The steady state distribution for the aggregate model is given by*

$$\tilde{P}(k,m_{11})=\begin{cases}\dfrac{Gq}{J_1!J_1^{S_1-k-m_{11}}}\left(\dfrac{p_1\lambda_1}{\mu_1}\right)^{m_{11}}\left(\dfrac{(1-p_1)\lambda_1}{\mu_0}\right)^{k}, & k+m_{11}\le S_1,k>0\\ \dfrac{Gq}{(S_1+J_1-k-m_{11})!}\left(\dfrac{p_1\lambda_1}{\mu_1}\right)^{m_{11}}\left(\dfrac{(1-p_1)\lambda_1}{\mu_0}\right)^{k}, & k+m_{11}>S_1,k>0\\ \dfrac{G}{J_1!J_1^{S_1-m_{11}}}\left(\dfrac{p_1\lambda_1}{\mu_1}\right)^{m_{11}}, & m_{11}\le S_1,k=0\\ \dfrac{G}{(S_1+J_1-m_{11})!}\left(\dfrac{p_1\lambda_1}{\mu_1}\right)^{m_{11}}, & m_{11}>S_1,k=0\end{cases} \tag{10}$$

with $G=\tilde{G}\lambda_1^{-(J_1+S_1)}$ *the normalization constant.*

The previous lemma gives an explicit expression for the steady state probabilities. For large systems it may be difficult to calculate the normalization constant G. However, since we are dealing with a product form network, Marginal Distribution Analysis (see e.g. Buzacott and Shanthikumar [4]) can be used to calculate the appropriate performance measures directly.

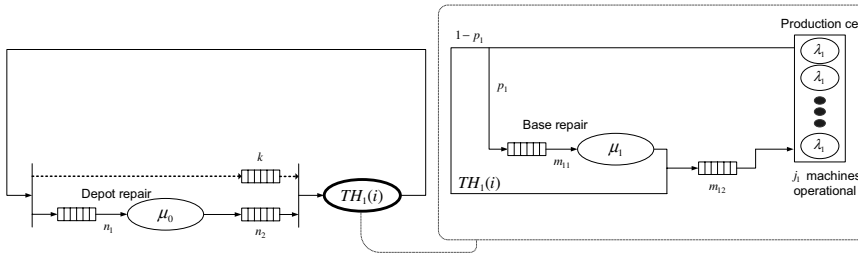
The results presented so far hold true for any value of $q\in[0,1]$. In the derivation of the lemmas above the interpretation of q as the conditional probability that a

request at the depot has to wait given that it finds no other requests in front of it (see (8)), has not been used. Therefore any $q \in [0, 1]$ will do, but it is expected that a good approximation will be obtained by using a q that does correspond to this interpretation. In the next subsection Norton's theorem will be used to find a q with a meaningful interpretation that gives good results.

2.3 Applying Norton's theorem to approximate q

Although we have stated in the previous section that the product form does not depend on q, it is still needed to find a q that gives a good approximation for the performance measures. In this section, the basic idea of Norton's theorem (see Harrison and Patel [6] for an overview) is used to find an approximation for q that gives good results. This basic idea is that a product form network can be analyzed by replacing subnetworks by state dependent servers. Norton's theorem states that the joint distributions for the numbers of customers in the subnetworks and the queue lengths at the replacing state dependent servers are the same.

To use this idea, first recall the original model as shown in Figure 2. We want to find q, the conditional probability that a request corresponding with a machine failure finds no spare parts in stock at the depot, although there was no backlog so far. The base, consisting of the production cell and the base repair shop, is taken apart and replaced by a state dependent server.



a **b**

Fig. 6. a The new network with state dependent server. **b** The short circuited network

The new network with the state dependent server is displayed in Figure 6a. In order to find the service rates for this state dependent server, the original network is short circuited by setting the service rate at the depot repair facility to infinity. This short circuited network is also depicted in Figure 6b. The service rate for the new state dependent server with i jobs present is equal to the throughput of the short circuited network with i jobs present, denoted by $TH_1(i)$.

The evolution of $\underline{n}_1 = n_1$, the number of machines in or awaiting depot repair, can be described as a birth-death process. The transition diagram is shown in Figure 7.

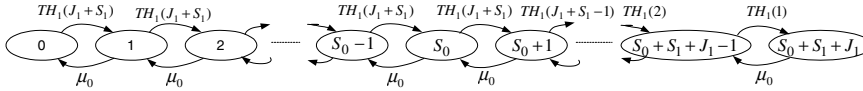


Fig. 7. Transition diagram for n_1

Note that this is just an approximation due to the fact that Norton's theorem is only valid for product form networks. In case $S_0 = 0$, we would have a product form network and the results would be exact. From the diagram one can observe that

$$P(\underline{n}_1 = n_1)\, TH_1(J_1 + S_1 - (n_1 - S_0)^+) = P(\underline{n}_1 = n_1 + 1)\, \mu_0 \tag{11}$$

for $n_1 = 0, \ldots, J_1 + S_1 + S_0 - 1$. In principle one can derive an approximation of the distribution of $\underline{n}_1$ from this. However, by the definition of q (see (8)), we only need to study the behavior for $\underline{n}_1 \leq S_0$. For these states, the service rate of the state dependent server is equal to $TH_1(J_1 + S_1)$. Let $\delta = TH_1(J_1 + S_1)/\mu_0$. From (11) we observe that $P(\underline{n}_1 = n_1) = \delta^{n_1} P(\underline{n}_1 = 0)$ for $n_1 = 0, \ldots, S_0$ so

$$\begin{aligned} q &= \frac{P(\underline{n}_1{=}S_0)}{P(\underline{n}_1 \leq S_0)} = \frac{\delta^{S_0} P(\underline{n}_1{=}0)}{\sum_{n_1=0}^{S_0} P(\underline{n}_1{=}n_1)} = \frac{\delta^{S_0} P(\underline{n}_1{=}0)}{\sum_{n_1=0}^{S_0} \delta^{n_1} P(\underline{n}_1{=}0)} = \frac{\delta^{S_0}}{\frac{1-\delta^{S_0+1}}{1-\delta}} \\ &= \delta^{S_0} \frac{1-\delta}{1-\delta^{S_0+1}}. \end{aligned} \tag{12}$$

It remains to find the throughput of the short circuited network in Figure 6b with $J_1 + S_1$ jobs present. A simple observation reveals that $P(\underline{b}_1 = b_1)\ min(b_1, J_1)\ \lambda_1\ p_1 = P(\underline{b}_1 = b_1 - 1)\mu_1$ for $b_1 = 1, \ldots, J_1 + S_1$ from which the steady state probabilities of $\underline{b}_1$ are immediately deduced. Moreover, the throughput satisfies

$$\begin{aligned} TH_1(J_1 + S_1) &= (1{-}p_1) \sum_{b_1=1}^{J_1+S_1} P(\underline{b}_1{=}b_1)\ min(b_1, J_1)\lambda_1 \\ &= \frac{1-p_1}{p_1} \mu_1 (1 - P(\underline{b}_1{=}J_1{+}S_1)). \end{aligned} \tag{13}$$

We can determine q with (12) and (13). This q can be used to approximate the steady state distribution using (10) or using Marginal Distribution Analysis. Results of this approximation are presented in the next section.

2.4 Results

In this section numerical results obtained by the approximation described above will be presented. To be able to judge the approximation the results are compared to exact results. The exact results are obtained by solving the balance equations for the original model.

The performance measures we are interested in are the availability, i.e. the probability that the maximum number of machines is working in the production

cell, denoted by A, and the expected number of machines operating in the production cell ($E\underline{j}_1$). These are defined as follows:

$$A = P(\underline{j}_1 = J_1) = P(\underline{b}_1 \geq J_1) = P(\underline{k} + \underline{m}_{11} \leq S_1), \tag{14}$$

$$E\underline{j}_1 = E(J_1 - [\underline{k} + \underline{m}_{11} - S_1]^+) = \sum_{k,m_{11}} (J_1 - [k + m_{11} - S_1]^+) P(k, m_{11}). \tag{15}$$

The performance measures are computed for several values of $J_1, S_0, S_1, p_1, \lambda_1, \mu_0$ and μ_1. The results are given in Table 1 and in Tables 5 and 6 in the Appendix. Also, the percentage deviation is given.

The numbers reveal that in these systems, the approximation gives an error of at most 1 %. In all other cases that we tested, we got similar results. The largest errors are attained in the cases with only a small number of spares ($S_0 > 0$) in the system. For the case $S_0 = 0$ the results are exact.

3 General two-echelon repairable item systems

In this section the simple system from Section 2 will be extended to a more realistic one. The system will contain multiple bases and transport lines. Furthermore, the single servers that are used in the repair shops are replaced by multiple parallel servers. These adjustments will make the analysis of the system more complicated. Nevertheless, the basic idea of the aggregation step will be the same.

3.1 The multi-base model with transportation

The system in this section consists of multiple bases, where the number of bases is denoted by L. A graphical representation of the system is given in Figure 8 for the case $L = 2$.

As in the simple system described before, at base $\ell = 1, \ldots, L$ at most J_ℓ machines are operating in the production cell. The machines fail at exponential rate λ_ℓ and are always replaced by a machine from the corresponding base stock (if available). Failed machines from base ℓ are base-repairable with probability p_ℓ and depot-repairable with probability $1 - p_\ell$. In contrast to the simple model described before, the repair shops are modeled as multi-servers. That is, at the repair shop of base $\ell = 1, \ldots, L$ R_ℓ repairmen are working, each at exponential rate μ_ℓ. At the depot repair shop R_0 repairmen are working at exponential rate μ_0. Consistent with the simple model S_ℓ machines are dedicated to base ℓ to act as spares and S_0 spare machines are dedicated to the depot. Broken machines at a certain base ℓ that are base-repairable are sent to the base ℓ repair shop. After repair they fill up the spares buffer at base ℓ or, in case of a backlog at that base, become operational immediately. Broken machines from base ℓ that are considered depot-repairable are sent to the depot repair shop. When depot spares are available, a spare is immediately sent to the stock of base ℓ. In case there are no spares available a backlog occurs. Machines that have completed repair are sent to the base that has been waiting the longest. That is, an FCFS return policy is used. In this model the transportation from the depot to

Table 1. Results for the simple single base model, $p_1 = 0.5, \lambda_1 = 1, \mu_0 = 2J_1, \mu_1 = J_1$

J_1	S_0	S_1	A_{exact}	A_{appr}	% dev	$Ej_{\underline{1}}exact$	$Ej_{\underline{1}}appr$	% dev
3	1	0	0.5651	0.5674	0.4185	2.4225	2.4246	0.0853
3	3	0	0.5889	0.5892	0.0543	2.4572	2.4576	0.0145
3	5	0	0.5901	0.5901	0.0041	2.4589	2.4590	0.0012
3	1	1	0.7945	0.7952	0.0934	2.7283	2.7286	0.0098
3	3	1	0.8110	0.8111	0.0154	2.7506	2.7507	0.0036
3	5	1	0.8120	0.8120	0.0014	2.7518	2.7518	0.0004
3	1	3	0.9506	0.9506	0.0012	2.9349	2.9348	0.0057
3	3	3	0.9554	0.9554	0.0000	2.9412	2.9412	0.0006
3	5	3	0.9557	0.9557	0.0000	2.9416	2.9416	0.0000
3	1	4	0.9755	0.9754	0.0012	2.9677	2.9676	0.0036
3	3	4	0.9779	0.9779	0.0004	2.9709	2.9709	0.0005
3	5	4	0.9781	0.9781	0.0000	2.9711	2.9711	0.0000
5	1	0	0.5369	0.5387	0.3314	4.3147	4.3160	0.0318
5	3	0	0.5625	0.5628	0.0461	4.3581	4.3584	0.0064
5	5	0	0.5639	0.5639	0.0037	4.3604	4.3604	0.0006
5	1	1	0.7759	0.7765	0.0761	4.6703	4.6704	0.0006
5	3	1	0.7940	0.7941	0.0127	4.6978	4.6979	0.0012
5	5	1	0.7950	0.7950	0.0012	4.6994	4.6994	0.0002
5	1	3	0.9453	0.9453	0.0012	4.9198	4.9196	0.0041
5	3	3	0.9506	0.9506	0.0000	4.9276	4.9276	0.0005
5	5	3	0.9510	0.0000	0.0000	4.9281	4.9281	0.0000
5	1	4	0.9727	0.9727	0.0009	4.9601	4.9600	0.0025
5	3	4	0.9755	0.9755	0.0003	4.9641	4.9640	0.0004
5	5	4	0.9757	0.9757	0.0000	4.9643	4.9643	0.0000
10	1	0	0.5091	0.5102	0.2178	9.1830	9.1837	0.0073
10	3	0	0.5363	0.5365	0.0328	9.2375	9.2377	0.0017
10	5	0	0.5379	0.5379	0.0028	9.2406	9.2406	0.0002
10	1	1	0.7565	0.7569	0.0507	9.5979	9.5977	0.0016
10	3	1	0.7762	0.7762	0.0087	9.6321	9.6321	0.0000
10	5	1	0.7774	0.7774	0.0008	9.6341	9.6341	0.0000
10	1	3	0.9395	0.9395	0.0006	9.9006	9.9004	0.0020
10	3	3	0.9455	0.9455	0.0001	9.9104	9.9104	0.0003
10	5	3	0.9458	0.9458	0.0000	9.9110	9.9110	0.0000
10	1	4	0.9698	0.9698	0.0007	9.9504	9.9503	0.0012
10	3	4	0.9728	0.9728	0.0002	9.9554	9.9554	0.0002
10	5	4	0.9730	0.9730	0.0000	9.9557	9.9557	0.0000

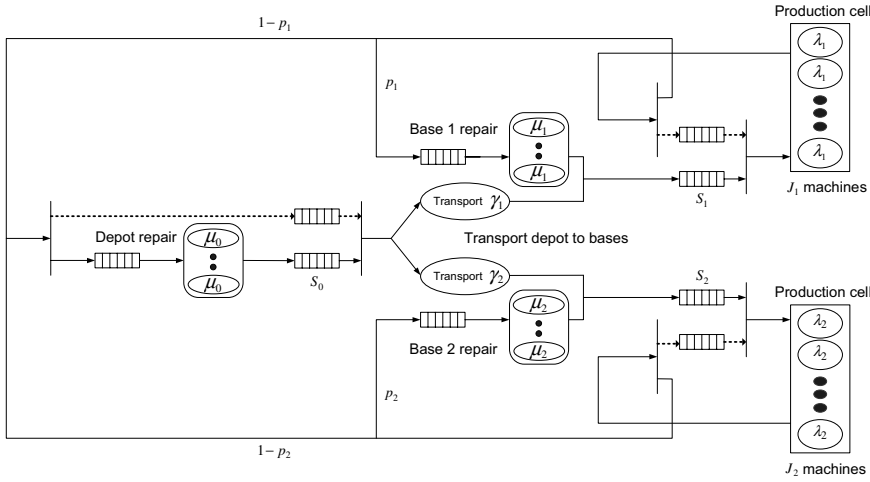


Fig. 8. The multi-base repairable item system for $L = 2$

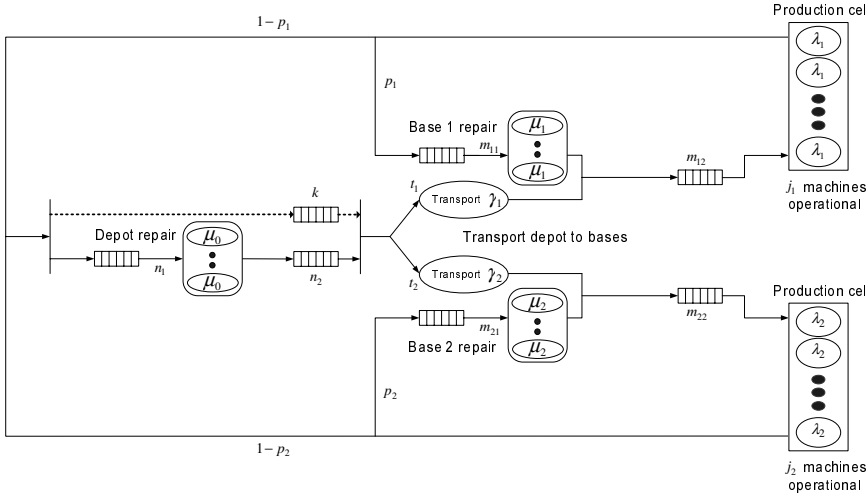


Fig. 9. The modified multi-base system for $L = 2$

the bases is taken into account explicitly. The transport lines are modeled as ample servers with exponential service rate γ_ℓ for the transport to base $\ell = 1, \ldots, L$. The number of machines in transport to base ℓ is denoted by the random variable $\underline{t}_\ell$. The transport from the bases to the depot is not taken into account.

As in the simple model, the synchronization queues at the bases can be replaced by ordinary queues as is depicted in Figure 9.

The vector $\mathbf{m}_1 = (m_{11}, m_{21}, \ldots, m_{L1})$ denotes the number of machines in base repair ($\ell = 1, \ldots, L$) and the vector $\mathbf{m}_2 = (m_{12}, m_{22}, \ldots, m_{L2})$ denotes the number of spares at the bases ($\ell = 1, \ldots, L$). The variable n_1 stands for the number of machines in depot repair and n_2 is the number of spare machines at the

depot. The vector $\mathbf{k}_0 = (k_{01}, k_{02}, \ldots, k_{0L})$ denotes the backorders at the depot, originating from base ℓ ($\ell = 1, \ldots, L$). The total number of backorders at the depot equals $k = \sum_{\ell=1}^{L} k_{0\ell}$. The machines in transit to the bases are given by the vector $\mathbf{t} = (t_1, t_2, \ldots, t_L)$ and the numbers of machines operating in the production cells are expressed in vector $\mathbf{j} = (j_1, j_2, \ldots, j_L)$. The sum of the number of machines in base stock and the number of machines operating in the production cell is denoted in the vector $\mathbf{b} = (b_1, b_2, \ldots, b_L)$, where $b_\ell = m_{\ell 2} + j_\ell$.

As a result of the operating inventory control policies, for $\underline{n}_1 = n_1$, $\underline{n}_2 = n_2$, $\underline{\mathbf{k}}_0 = \mathbf{k}_0$, $\underline{\mathbf{t}} = \mathbf{t}$, $\underline{\mathbf{m}}_1 = \mathbf{m}_1$, $\underline{\mathbf{m}}_2 = \mathbf{m}_2$ and $\underline{\mathbf{j}} = \mathbf{j}$ the following equations must hold:

$$n_1 + n_2 - k = S_0, \tag{16}$$

$$n_2 \cdot k = 0, \tag{17}$$

and for $\ell = 1, 2, \ldots, L$:

$$k_{0\ell} + t_\ell + m_{\ell 1} + m_{\ell 2} + j_\ell = S_\ell + J_\ell, \tag{18}$$

$$m_{\ell 2} \cdot (J_\ell - j_\ell) = 0. \tag{19}$$

From these relations it follows immediately that $\mathbf{k}_0$, n_1, $\mathbf{t}$ and $\mathbf{m}_1$ completely determine the state of the system. Therefore, the system can be modeled as a continuous time Markov chain with state description $(\mathbf{k}_0, n_1, \mathbf{t}, \mathbf{m}_1)$.

Remark 3 In the vector that denotes the number of backorders originating from the bases, $\mathbf{k}_0 = (k_{01}, k_{02}, \ldots, k_{0L})$, it is not taken into account that the order of the backorders matters. Since an FCFS return policy is assumed, this order should be known. Nevertheless, in this model all states with similar numbers of backorders per base, are aggregated into one state. This aggregation step will not have a big influence on the results, but it will considerably simplify the analysis.

3.2 Approximation

In correspondence with the simple model as described in Section 2 a similar aggregation step is performed to tackle this extended model. Once more, all states with $0 \leq n_1 \leq S_0$ are aggregated into one state. The aggregation step is performed as follows

$$\begin{aligned} &P(\underline{\mathbf{k}}_0 = \mathbf{0}, \underline{k} = 0, \underline{\mathbf{t}} = \mathbf{t}, \underline{\mathbf{m}}_1 = \mathbf{m}_1) \\ &\quad = \sum_{n_1=0}^{S_0} P(\underline{\mathbf{k}}_0 = \mathbf{0}, \underline{n}_1 = n_1, \underline{\mathbf{t}} = \mathbf{t}, \underline{\mathbf{m}}_1 = \mathbf{m}_1) \end{aligned} \tag{20}$$

$$\begin{aligned} &P(\underline{\mathbf{k}}_0 = \mathbf{k}_0, \underline{k} = k, \underline{\mathbf{t}} = \mathbf{t}, \underline{\mathbf{m}}_1 = \mathbf{m}_1) \\ &\quad = P(\underline{\mathbf{k}}_0 = \mathbf{k}_0, \underline{n}_1 = S_0 + k, \underline{\mathbf{t}} = \mathbf{t}, \underline{\mathbf{m}}_1 = \mathbf{m}_1) \end{aligned} \tag{21}$$

The aggregated system can be described by $(\mathbf{k}_0, k, \mathbf{t}, \mathbf{m}_1)$. Furthermore, because $k = \sum_{\ell=1}^{L} k_{0\ell}$ the state space can also be described by $(\mathbf{k}_0, \mathbf{t}, \mathbf{m}_1)$.

Define q as before, that is q is the conditional probability that an arriving request at the depot cannot be fulfilled immediately, given that there are no other requests

waiting. In a formula it says $q = P(\underline{n}_1 = S_0 | \underline{n}_1 \leq S_0)$. So, given there is no backlog at the depot, an arriving request has to wait with probability q. The waiting time depends on the number of spares already in the queue.

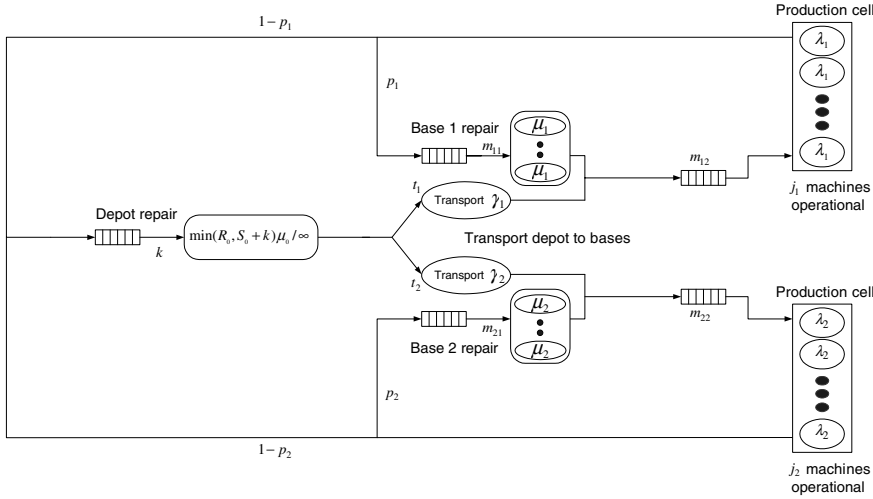


Fig. 10. The Typical-server Closed Queuing Network

The first spare that finishes repair will fulfill the just arrived request. With probability $1 - q$ spares are available and the arriving request does not have to wait. This aggregated network is depicted as a Typical-server Closed Queuing Network in Figure 10. The depot repair shop is modeled as a typical server. In case of no backlog ($k = 0$) the service rate equals infinity with probability $1 - q$ and equals $min(S_0, R_0)\mu_0$ with probability q. In all other cases ($k > 0$) the service rate equals $min(k + S_0, R_0)\mu_0$.

To determine q Norton's theorem is used once more. As in Subsection 2.3 each base (the transport line, the base repair shop and the production cell) is replaced by a state dependent server. To determine the transition rate of this state dependent server, each base-part of the network is short circuited and its throughput is calculated. This throughput operates as the service rate of the state dependent server. The new network with the state dependent servers and the short circuited networks are depicted in Figure 11.

Once again the evolution of $\underline{n}_1$ can be described as a birth-death process. The (approximated) transition diagram for $n_1 = 0, \ldots, S_0$ is given in Figure 12.

Let $TH_\ell(i)$ be the throughput of the subnetwork replacing base ℓ ($\ell = 1, \ldots, L$) with i jobs present. As in the simple model only the behavior for $\underline{n}_1 \leq S_0$ needs to

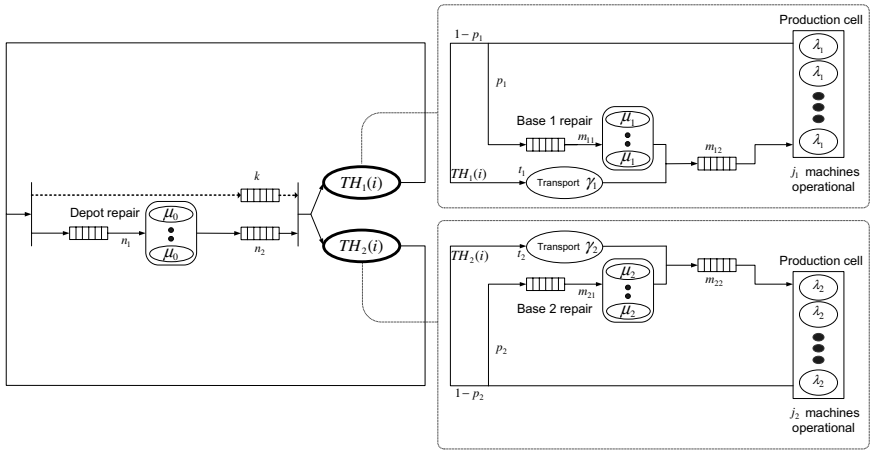


Fig. 11. a The new network with state dependent servers. **b** The short circuited networks

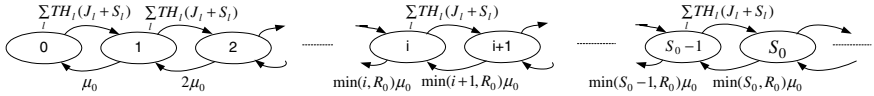


Fig. 12. Transition diagram for n_1

be studied to determine q. Take $\delta = \sum_\ell TH_\ell(J_\ell + S_\ell)/\mu_0$, then

$$P(\underline{n}_1 = n_1) = \delta^{n_1} \frac{1}{\prod_{k=1}^{n_1} min(k, R_0)} P(\underline{n}_1 = 0) \quad \text{for} \quad n_1 = 0, \ldots, S_0 \tag{22}$$

and

$$\begin{aligned} q &= \frac{P(\underline{n}_1 = S_0)}{P(\underline{n}_1 \le S_0)} = \frac{P(\underline{n}_1 = S_0)}{\sum_{n_1=0}^{S_0} P(\underline{n}_1 = n_1)} \\ &= \frac{\delta^{S_0} \frac{1}{\prod_{k=1}^{S_0} min(k,R_0)} P(\underline{n}_1 = 0)}{\sum_{n_1=0}^{S_0} \delta^{n_1} \frac{1}{\prod_{k=1}^{n_1} min(k,R_0)} P(\underline{n}_1 = 0)} \\ &= \frac{\delta^{S_0} \frac{1}{\prod_{k=1}^{S_0} min(k,R_0)}}{\sum_{n_1=0}^{S_0} \delta^{n_1} \frac{1}{\prod_{k=1}^{n_1} min(k,R_0)}}. \end{aligned} \tag{23}$$

The throughputs can be obtained by applying a standard MDA algorithm (see [4]) on the short circuited product form networks as shown in Figure 11.

The steady state marginal probabilities as well as the main performance measures for the aggregated system can be found by using an adapted Multi-Class Marginal Distribution Analysis algorithm (see Buzacott and Shanthikumar [4] for ordinary Multi-Class MDA). To see this, introduce tokens of class ℓ with

$\ell = 1, \ldots, L$ that either represent machines present at base ℓ (in the production cell, in the base repair shop, in the base stock or in transit to this base) or represent requests to the depot stock emerging from a failure of a machine at base ℓ that cannot be repaired locally. Recall that machines that have to be repaired in the depot repair shop, in fact lose their identity, i.e. after completion they are placed in the depot stock, from which they can in principle be shipped to any arbitrary base. However, the request arriving jointly with that broken machine at the depot, maintains its identity, meaning that it is matched with the first spare machine available, after which the combination is transported to the base the request originated from. Therefore, a token can be seen as connected to a machine as long as that machine is at the base (in any status) and connected with the corresponding request as soon as the machine is sent to the depot. This request matches with an available machine from stock (which generally is different from the one sent to the depot, unless $S_0 = 0$) and the combination returns to the base that generated the request. Hence, in this way, a multi-class network arises in a natural way.

The adapted algorithm is given below. An important aspect of an MDA algorithm is the computation of the expected sojourn time in the stations. Since the depot repair shop is modeled as a typical server, the standard sojourn time as described in [4] will not do for this station. As denoted before, in case of no backlog ($k = 0$) the service rate equals infinity with probability $1 - q$ and equals $min(S_0, R_0)\mu_0$ with probability q. In all other cases ($k > 0$) the service rate equals $min(k + S_0, R_0)\mu_0$. The expected sojourn time of an arriving request is the time it takes until all requests in front of it (k) are fulfilled and the request itself is fulfilled. That is, the time until $k + 1$ machines come out of repair. In case $k = 0$ with probability $1 - q$ the sojourn time equals 0 because a spare fulfills the request. The adaptations to the sojourn time reveal themselves in the algorithm in step 4. Another adaptation to the ordinary algorithm is found in step 6. The transition rates from the states with 0 machines in depot repair to the states with 1 machine in depot repair now equal q times the throughput, instead of just the throughput.

Algorithm 4 *The depot repair shop is defined as station* 0 *and all other stations are defined as station ℓi, where ℓ denotes the number of the base ($\ell = 1, \ldots, L$) and i denotes the specific station associated with that base. The production cell is denoted by $i = b$, the base repair shop by $i = m$ and the transport line from the depot to the base by $i = t$.*

Let $V_j^{(r)}$ be the visit ratio of station j for class r type machines. Let z denote the number of machines in the system and $\mathbf{z} = (z_1, \ldots, z_r, \ldots, z_L)$ the vector denoting the state that indicates the number of machines per class. The steady state probability that y machines are in station j, given vector $\mathbf{z}$ is denoted by $p_j(y|\mathbf{z})$. The expected sojourn time for type r machines arriving at station j given that $\mathbf{z}$ machines are wandering through the system is given by $EW_j^{(r)}(\mathbf{z})$ and $TH_j^{(r)}(\mathbf{z})$ denotes the throughput of type r machines given state $\mathbf{z}$. The algorithm is executed as follows:

1. *(Initialization) For $\ell = 1, \ldots, L$ set $V_0^{(\ell)} = 1$, $V_{lb}^{(\ell)} = \frac{1}{1-p_\ell}$, $V_{lm}^{(\ell)} = \frac{p_\ell}{1-p_\ell}$ and $V_{lt}^{(\ell)} = 1$. For $\ell = 1, \ldots, L$, $r = 1, \ldots, L$, $r \neq \ell$, $i \in \{b, m, t\}$ set $V_{\ell i}^{(r)} = 0$. Set $z = 0$ and $p_j(0|\mathbf{0}) = 1$ for $j \in \bigcup_\ell \{lb, lm, lt\} \cup \{0\}$.*

2. *z:=z+1.*
3. *For all states* $\mathbf{z} \in \{\mathbf{z} | \sum_{\ell=1}^{L} z^{(\ell)} = z \text{ and } z^{(\ell)} \leq J_\ell + S_\ell\}$ *execute steps 4 through 6.*
4. *Compute the sojourn times for* $\ell = 1, \ldots, L$ *for which* $z^{(\ell)} > 0$ *from:*

$$EW_0^{(\ell)}(\mathbf{z}) = \sum_{k=1}^{z-1} \frac{k+1}{min(R_0, S_0+k+1)\mu_0} p_0(k|\mathbf{z}-\mathbf{e}_\ell) + \frac{q}{min(R_0, S_0+1)\mu_0} p_0(0|\mathbf{z}-\mathbf{e}_\ell),$$

$$EW_{lb}^{(\ell)}(\mathbf{z}) = \sum_{b_\ell=J_\ell}^{z-1} \frac{b_\ell - J_\ell + 1}{J_\ell \lambda_\ell} p_{lb}(b_\ell|\mathbf{z}-\mathbf{e}_\ell) + \frac{1}{\lambda_\ell},$$

$$EW_{lm}^{(\ell)}(\mathbf{z}) = \sum_{m_{\ell 1}=R_\ell}^{z-1} \frac{m_{\ell 1} - R_\ell + 1}{R_\ell \mu_\ell} p_{lm}(m_{\ell 1}|\mathbf{z}-\mathbf{e}_\ell) + \frac{1}{\mu_\ell},$$

$$EW_{lt}^{(\ell)}(\mathbf{z}) = \frac{1}{\gamma_\ell}.$$

5. *Compute* $TH_0^{(\ell)}(\mathbf{z})$ *for* $\ell = 1, \ldots, L$ *if* $z^{(\ell)} > 0$ *from:*

$$TH_0^{(\ell)}(\mathbf{z}) = \frac{z^{(\ell)}}{V_0^{(\ell)} EW_0^{(\ell)} + \sum_{i \in \{b,m,t\}} V_{\ell i}^{(\ell)} EW_{\ell i}^{(\ell)}},$$

and if $z^{(\ell)} = 0$ *then* $TH_0^{(\ell)}(\mathbf{z}) = 0$. *Compute* $TH_{\ell i}^{(\ell)}(\mathbf{z})$ *for* $\ell = 1, \ldots, L$ *and* $i \in \{b, m, t\}$ *from:*

$$TH_{\ell i}^{(\ell)}(\mathbf{z}) = V_{\ell i}^{(\ell)} TH_0^{(\ell)}(\mathbf{z}).$$

6. *Compute the marginal probabilities for all stations from:*

$$\mu_0\ min(R_0, S_0+1)\ p_0(1|\mathbf{z}) = \sum_{\ell=1}^{L} TH_0^{(\ell)}(\mathbf{z})\ q\ p_0(0|\mathbf{z}-\mathbf{e}_\ell),$$

$$\mu_0\ min(R_0, S_0+k)\ p_0(k|\mathbf{z}) = \sum_{\ell=1}^{L} TH_0^{(\ell)}(\mathbf{z}) p_0(k-1|\mathbf{z}-\mathbf{e}_\ell) \text{ for } k=2, \ldots, z,$$

and for $\ell = 1, \ldots, L$ *from:*

$$\lambda_\ell min(J_\ell, b_\ell) p_{lb}(b_\ell|\mathbf{z}) = TH_{lb}^{(\ell)}(\mathbf{z}) p_{lb}(b_\ell - 1|\mathbf{z}-\mathbf{e}_\ell) \quad \text{for } b_\ell = 1, \ldots, z,$$

$$\mu_\ell\ min(R_\ell, m_{\ell 1})\ p_{lm}(m_{\ell 1}|\mathbf{z}) = TH_{lm}^{(\ell)}(\mathbf{z})\ p_{lm}(m_{\ell 1} - 1|\mathbf{z}-\mathbf{e}_\ell) \quad \text{for } m_{\ell 1} = 1, \ldots, z,$$

$$\gamma_\ell\ t_\ell\ p_{lt}\ (t_\ell|\mathbf{z}) = TH_{lt}^{(\ell)}(\mathbf{z})\ p_{lt}(t_\ell - 1|\mathbf{z}-\mathbf{e}_\ell) \quad \text{for } t_\ell = 1, \ldots, z.$$

Compute $p_j(0|\mathbf{z})$ *for* $j \in \bigcup_\ell \{lb, lm, lt\} \cup \{0\}$ *from:*

$$p_j(0|\mathbf{z}) = 1 - \sum_{y=1}^{z} p_j(y|\mathbf{z}).$$

7. If $z = \sum_{\ell=1}^{L} J_\ell + S_\ell$ then stop; else go to step 2.

With the adapted Multi-Class MDA algorithm presented above, the marginal probabilities of the system as well as the throughputs and the sojourn times can be approximated. From these, various performance measures can be computed. In the next section some results obtained by the algorithm will be compared with results from simulation.

Remark 5 Opposite to the simple problem discussed in Section 2 (which merely served to illustrate the basic steps of the aggregation procedure), an exact solution approach for the current extended problem already proves to be computationally intractable, due to the curse of dimensionality. The aggregation procedure, on the other hand, yields no essential computational difficulties. This is due to two reasons. First of all, the aggregation and subsequent small changes on some border transition rates allow us to come up with a near-product form solution for the approximated system. Second, as a result of that, we are able to apply Norton's theorem, which allows for an exact decomposition of the remaining approximated model. Although for large problems the adapted Multi-Class MDA algorithm becomes slower, standard approximation techniques for multi-class systems are available to speed up these algorithms further, without losing much accuracy (see also our final remarks in Sect. 5).

3.3 Results

In this section results obtained by the adapted Multi-Class MDA algorithm from the previous section will be presented. They will be compared to results obtained by simulation. For each base we are interested in the availability, that is the probability that the maximum number of machines is operating in the production cell. For base ℓ this is denoted by A_ℓ for $\ell = 1, \ldots, L$. Furthermore we are interested in the expected number of machines operating in the production cell, denoted by $E\underline{j}_\ell$ for base ℓ=1, ... , L. For ℓ=1, ... , L the performance measures can be computed by

$$A_\ell = P(\underline{j}_\ell = J_\ell) = P(\underline{b}_\ell \geq J_\ell) = P(\underline{k}_{0\ell} + \underline{m}_{\ell 1} \leq S_\ell), \tag{24}$$

$$\begin{aligned} E\underline{j}_\ell &= E(J_\ell - [\underline{k}_{0\ell} + \underline{m}_{\ell 1} - S_\ell]^+) \\ &= \sum_{k_{0\ell}, m_{\ell 1}} (J_\ell - [k_{0\ell} + m_{\ell 1} - S_\ell]^+) P(k_{0\ell}, m_{\ell 1}). \end{aligned} \tag{25}$$

In Table 2 and Table 7 in the Appendix, the parameter settings for some representative test problems are given. In this section, we consider dual base systems ($L = 2$); in the appendix we also have examples of systems with three ($L = 3$) and four bases ($L = 4$). The other parameters in this case are given in Table 2 with

J_ℓ, the maximum number of working machines at base ℓ,
S_ℓ, the maximum number of stored items at base ℓ (or at the depot),
λ_ℓ, the breakdown rate of individual machines at base ℓ,
μ_ℓ, the repair rate of individual machines at base ℓ (or at the depot),
R_ℓ, the number of repairmen at base ℓ (or at the depot),

p_ℓ, the probability that a machine can be repaired at base ℓ,
γ_ℓ, the transportation rate to base ℓ and
ρ_ℓ, the traffic intensity at the base ℓ (or at the depot).

The first 10 models are symmetric, that is the same parameter values apply to both bases. The other 10 problems concern asymmetric cases. It is obvious that a large number of input parameters is required to specify a given problem. This makes it difficult to vary these parameters in a totally systematic manner. In Albright [2] it is shown that traffic intensities are good indicators of whether a system will work well (minimal backorders) and are better indicators than the stock levels. Therefore we selected most of the test problem parameter settings by selecting values of the traffic intensities, usually well less than 1, and then selecting parameters to achieve these traffic intensities. For the base ℓ repair facility, the traffic intensity ρ_ℓ is defined as

$$\rho_\ell = J_\ell \lambda_\ell p_\ell / R_\ell \mu_\ell, \tag{26}$$

the maximum failure rate divided by the maximum repair rate. Similarly, the depot traffic intensity ρ_0 is defined as

$$\rho_0 = \sum_{\ell=1}^{L} J_\ell \lambda_\ell (1 - p_\ell) / R_0 \mu_0. \tag{27}$$

The results are given in Table 3 and Table 8 in the Appendix. The simulation leads to 95 % confidence intervals. The simulation method was the so called replication deletion method where the warmup period was found by Welch's graphical procedure (cf. Law and Kelton [7]). To compare the approximations with the simulation results, the deviation from the approximation to the midpoint of the confidence interval is calculated. These percentage deviations are given as well.

From the results it can be concluded that the approximations are very accurate. The maximum deviation for the availability as well as the relative deviation for the expected number of working machines, is well less than 1% and all approximating values lie within the confidence intervals. Furthermore, all types of problems exhibited similar levels of accuracy.

4 Optimization

In the preceeding sections an accurate approximation for several performance measures of closed two-echelon repairable item systems has been obtained. These approximation methods can be used to find an optimal allocation of spares in the system, in order to achieve the best performance. In this section we give algoritms to find the optimal allocation.
At first, we formulate the optimization problem. Subsequently, we present a fast but reliable greedy approximation scheme for the optimization problem. The section is concluded with some numerical results.

Table 2. Parameter settings for test problems multi-base model with transportation (1)

Problem	depot S_0	depot μ_0	depot R_0	depot ρ_0	base	J_ℓ	S_ℓ	λ_ℓ	μ_ℓ	R_ℓ	p_ℓ	γ_ℓ	ρ_ℓ
1	1	20	1	0.5	1/2	10	2	1	10	1	0.5	∞	0.5
2	1	10	1	0.5	1/2	5	2	1	5	1	0.5	∞	0.5
3	1	10	2	0.25	1/2	5	2	1	5	2	0.5	∞	0.25
4	1	10	1	0.5	1/2	5	2	1	5	1	0.5	10	0.5
5	1	10	1	0.5	1/2	5	2	1	5	1	0.5	2	0.5
6	1	10	2	0.25	1/2	5	2	1	5	2	0.5	2	0.25
7	1	2	5	0.5	1/2	5	2	1	1	5	0.5	2	0.5
8	7	2	5	0.5	1/2	5	2	1	1	5	0.5	2	0.5
9	5	6	1	0.83	1/2	5	5	1	3	1	0.5	∞	0.83
10	5	10	1	0.5	1/2	5	5	1	5	1	0.5	∞	0.5
11	1	20	1	0.5	1	10	2	1	10	1	0.5	∞	0.5
					2	10	2	1	3	1	0.5	∞	1.67
12	1	20	1	0.38	1	10	2	1	10	1	0.5	∞	0.5
					2	10	2	1	3	1	0.75	∞	2.5
13	1	20	1	0.38	1	10	2	1	10	1	0.5	1	0.5
					2	10	2	1	20	1	0.75	∞	0.375
14	2	20	1	0.25	1	10	5	1	10	1	0.5	2	0.5
					2	10	2	1	20	1	0.5	2	0.25
15	2	10	1	0.5	1	10	1	1	12	1	0.5	∞	0.42
					2	10	4	1	3	4	0.5	∞	0.42
16	1	10	1	0.5	1	5	2	1	5	1	0.5	∞	0.5
					2	5	2	1	5	1	0.5	300	0.5
17	1	10	1	0.5	1	5	2	1	5	1	0.5	∞	0.5
					2	5	2	1	5	1	0.5	5	0.5
18	1	10	1	0.5	1	5	2	1	5	1	0.5	∞	0.5
					2	5	2	1	5	1	0.5	2	0.5
19	1	6	1	1.67	1	10	2	1	5	1	0.5	∞	1
					2	10	2	1	5	1	0.5	2	1
20	1	10	1	1	1	10	2	1	5	1	0.5	∞	1
					2	10	2	1	5	1	0.5	2	1

4.1 The optimization problem

The aim is to maximize the overall performance of the system under a budget constraint for stocking costs. For the overall performance of the two-echelon repairable item system, the total availability A_{tot}, defined by

$$A_{tot} = \frac{\sum_{\ell=1}^{L} J_\ell \lambda_\ell A_\ell}{\sum_{\ell=1}^{L} J_\ell \lambda_\ell},$$

Table 3. Results for test problems from Table 2

Problem	base	$A_\ell sim$	$A_\ell appr$	% dev	$E\underline{j}_\ell sim$	$E\underline{J}_\ell appr$	% dev
1	1/2	(0.8529,0.8563)	0.8542	0.05	(9.7533,9.7615)	9.7562	0.01
2	1/2	(0.8638,0.8750)	0.8683	0.13	(4.7957,4.8161)	4.8043	0.03
3	1/2	(0.9695,0.9714)	0.9701	0.04	(4.9626,4.9655)	4.9633	0.02
4	1/2	(0.8311,0.8403)	0.8353	0.04	(4.7461,4.7640)	4.7543	0.02
5	1/2	(0.6548,0.6639)	0.6605	0.18	(4.4542,4.4737)	4.4672	0.07
6	1/2	(0.7490,0.7539)	0.7514	0.00	(4.6463,4.6545)	4.6521	0.04
7	1/2	(0.2938,0.3008)	0.2978	0.17	(3.6284,3.6497)	3.6445	0.15
8	1/2	(0.3781,0.3883)	0.3800	0.83	(3.8866,3.9096)	3.8907	0.19
9	1/2	(0.8165,0.8361)	0.8234	0.34	(4.6622,4.7032)	4.6770	0.12
10	1/2	(0.9854,0.9894)	0.9875	0.01	(4.9785,4.9851)	4.9817	0.00
11	1	(0.8631,0.8703)	0.8663	0.05	(9.7672,9.7864)	9.7782	0.01
	2	(0.0739,0.0830)	0.0797	1.54	(5.8615,5.9716)	5.9036	0.22
12	1	(0.8733,0.8785)	0.8753	0.07	(9.7915,9.8028)	9.7940	0.03
	2	(0.0078,0.0100)	0.0082	8.06	(3.9778,4.0665)	3.9965	0.64
13	1	(0.1252,0.1367)	0.1303	0.49	(7.5031,7.5736)	7.5390	0.01
	2	(0.9423,0.9455)	0.9452	0.14	(9.9154,9.9219)	9.9208	0.02
14	1	(0.8466,0.8565)	0.8512	0.04	(9.7283,9.7524)	9.7408	0.00
	2	(0.4846,0.4995)	0.4895	0.52	(9.0411,9.0802)	9.0602	0.00
15	1	(0.4413,0.4647)	0.4382	3.27	(8.6368,8.7362)	8.6387	0.55
	2	(0.7007,0.7231)	0.7012	1.50	(9.3273,9.3925)	9.3375	0.24
16	1	(0.8617,0.8694)	0.8693	0.43	(4.7934,4.8076)	4.8043	0.08
	2	(0.8625,0.8717)	0.8673	0.02	(4.7938,4.8113)	4.8028	0.00
17	1	(0.8644,0.8734)	0.8691	0.03	(4.7985,4.8139)	4.8057	0.01
	2	(0.7899,0.8005)	0.7957	0.06	(4.6831,4.7016)	4.6928	0.01
18	1	(0.8690,0.8752)	0.8707	0.16	(4.8049,4.8158)	4.8082	0.05
	2	(0.6514,0.6618)	0.6579	0.20	(4.4494,4.4689)	4.4620	0.06
19	1	(0.0742,0.0837)	0.0769	2.60	(6.2112,6.2972)	6.2354	0.30
	2	(0.0277,0.0338)	0.0301	2.15	(5.5644,5.6682)	5.5946	0.39
20	1	(0.3298,0.3430)	0.3354	0.29	(8.0845,8.1484)	8.1002	0.20
	2	(0.1417,0.1492)	0.1472	1.20	(7.2625,7.3228)	7.2967	0.06

is taken. It can be considered as the weighted average of the availabilities per base. The total availability is considered as a function of the maximal stock sizes $S_0, S_1, \cdots, S_L$; the other parameters that influence the total availability are given. The constraint for the optimization problem is an upperbound C for the total stocking costs. The stocking costs are linear in the maximum stock sizes. Let c_ℓ be the storage cost for keeping one spare at stockpoint ℓ. The (non-linear) optimization

problem can now be formulated as:

$$\begin{aligned} \max \quad & A_{tot}(S_0, \ldots, S_L), \\ \text{s.t.} \quad & \sum_{\ell=0}^{L} c_\ell S_\ell \leq C, \\ & S_\ell \geq 0, \text{ for } \ell = 0, \ldots, L. \end{aligned}$$

In the next subsection a greedy approximation scheme will be given to approximate the optimal values for $S_0, \ldots, S_L$.

4.2 Optimization algorithm

The most straightforward solution method to find optimal stock levels, is the brute force method. This method simply checks all feasible allocations and picks the one which gives the highest total availability. By assuming that A_{tot} is an increasing function, the brute force can be improved by considering only allocations on the boundary of the feasible region, that is those allocation where adding another spare part would lead to an infeasible allocation. Even this improved brute force approach turns out to be rather time consuming.

In Zijm and Avsar [10], a greedy approximation procedure is given to find the optimal allocation of stocks for an open two-indenture model. This method can also be applied on closed two-echelon repairable item systems.
At the start of the heuristic algorithm no spares are allocated. One repeatedly allocates one spare to the location that leads to the maximum increase in total availability per unit of money invested, under the constraint that the allocation is feasible. The heuristic continues as long as this maximum increase is positive; it can be presented as follows:

Algorithm 6 *Approximative optimization method (greedy approach)*

1. *(Initialization) Set* $\hat{S}_\ell = 0$*, for* $\ell = 0, 1, \ldots, L$*, and set* $\hat{C} = 0$.
2. *(Repetition) Define* Δ_ℓ *for* $\ell = 0, 1, \ldots, L$*, by*

$$\Delta_\ell = \begin{cases} \dfrac{A_{tot}(\hat{S}_0, \ldots, \hat{S}_\ell + 1 \ldots, \hat{S}_L) - A_{tot}(\hat{S}_0, \ldots, \hat{S}_\ell \ldots, \hat{S}_L)}{c_\ell} & \text{if } \hat{C} + c_\ell \leq C, \\ 0, & \text{otherwise.} \end{cases}$$

Let $\hat{\ell} = \arg\max_\ell \Delta_\ell$*. If* $\Delta_{\hat{\ell}} \leq 0$ *then stop; otherwise repeat this step after setting* $\hat{S}_{\hat{\ell}} = \hat{S}_{\hat{\ell}} + 1$ *and* $\hat{C} = \hat{C} + c_{\hat{\ell}}$.

3. *(Solution) The resulting stock allocation* $(\hat{S}_0, \hat{S}_1, \ldots, \hat{S}_L)$ *is the approximative solution to the optimization problem.*

The greedy heuristic presented above builds on the observation that $A_{tot}(S_0, \ldots, S_L)$ tends to behave as an increasing multi-dimensional concave function, in particular for not too small values of $S_i, i = 1, \ldots, L$. This observation

Table 4. Optimal stock sizes for test problems

Problem	base	J_ℓ	λ_ℓ	μ_ℓ	R_ℓ	p_ℓ	γ_ℓ	c_ℓ	C	$A_{tot,bf}$	$S_{\ell,bf}$	$A_{tot,greedy}$	$S_{\ell,greedy}$
1	depot			5	1			1	10	0.7513	2	0.7513	2
	1	5	1	5	1	0.5	10	1			4		4
	2	5	1	5	1	0.5	10	1			4		4
2	depot			5	1			1	20	0.8668	6	0.8662	7
	1	5	1	5	1	0.5	10	1			7		7
	2	5	1	5	1	0.5	10	1			7		6
3	depot			5	1			1	20	0.8328	9	0.8328	9
	1	5	1	5	1	0.5	10	2			3		3
	2	5	1	5	1	0.5	10	1			5		5
4	depot			5	1			1	20	0.7977	8	0.7977	8
	1	5	1	5	1	0.5	10	2			3		3
	2	5	1	5	1	0.5	10	2			3		3
5	depot			5	1			1	20	0.6487	4	0.6438	2
	1	5	1	5	1	0.5	1	2			4		5
	2	5	1	5	1	0.5	1	2			4		4
6	depot			5	2			1	20	0.9716	4	0.9716	4
	1	5	1	5	1	0.5	10	2			4		4
	2	7	1	5	2	0.5	10	2			4		4
7	depot			5	2			2	20	0.9987	0	0.9987	0
	1	5	1	5	1	0.5	10	1			10		10
	2	7	1	5	2	0.5	10	1			10		10
8	depot			5	3			1	20	0.9144	4	0.9144	4
	1	10	1	5	2	0.5	10	2			4		4
	2	10	1	5	2	0.5	10	2			4		4
9	depot			5	3			1	20	0.6327	6	0.6234	6
	1	10	2	5	4	0.5	10	2			5		4
	2	10	1	5	2	0.5	10	2			2		3
10	depot			3	2			1	20	0.6976	2	0.6976	2
	1	3	1	3	1	0.2	1	2			3		3
	2	7	1	3	2	0.8	1	2			6		6

of concavity is strongly supported by empirical evidence. In addition, we note that in the uncapacitated case, a formal proof of the concavity of the availability function can be given, based on convexity properties of backorder probabilities as a function of the base stock levels (see e.g. Rustenburg et al. [8]), at least when the values of $S_i, i = 1, \ldots, L$, exceed certain (low) thresholds. In other words: a law of diminishing added value is valid here, and is again very likely to hold in the capacitated case as well. If A_{tot} is an increasing function, the heuristic will stop when the boundary of the feasible region is reached. In the next section the greedy approach is numerically compared with the brute force approach.

In this section results are obtained for several test problems. The results obtained by the brute force approach are compared to the results found by the greedy approach. Even when the greedy approach gives a different allocation for spare items, the total availability only decreases slightly.
In Table 4 several test problems are presented. The parameters in this case are

J_ℓ, the maximum number of working machines at base ℓ,
λ_ℓ, the breakdown rate of individual machines at base ℓ,
μ_ℓ, the repair rate of individual machines at base ℓ (or at the depot),
R_ℓ, the number of repairmen at base ℓ (or at the depot),
p_ℓ, the probability that a machine can be repaired at base ℓ,
γ_ℓ, the transportation rate to base ℓ,
c_ℓ, the costs to store an item at base ℓ (or at the depot),
C, the available budget for storing items.

Note that the the maximal stock sizes $S_0, S_1, \ldots, S_L$ and the total availability A_{tot} are not given but computed by either the brute force approach ($A_{tot,bf}$ and $S_{\ell,bf}$) or by Heuristic 6 ($A_{tot,greedy}$ and $S_{\ell,greedy}$). The numerical results indicate that the greedy approach yields good results.

5 Summary and possible extensions

In this paper we have analyzed a closed loop two-echelon repairable item system with a fixed number of items circulating in the network. The system consists of several bases and a central repair facility (depot). Each base consists of a production cell and a base repair shop. There are transport lines leading from the depot to the bases. Transport from bases to the depot is not taken into account. The repair shops are modeled as multi-servers and the transport lines as ample servers. Repair shops at the depot as well as at the bases are able to keep a number of ready-for-use items in stock. Machines that have failed in the production cell of a certain base are immediately replaced by a ready-for-use machine from that base's stock, if available. The failed machine is sent to either the base repair facility or to the depot repair facility, in the latter case a spare machine is sent from the depot to the base, to deplete the base's stock of ready-for-use items. Once the machine at the depot is repaired, it is added to the central stock. Orders are satisfied on a first-come-first-served basis while any requirement that cannot be satisfied immediately either at a base or at the depot is backlogged. In case of a backlog at a certain base, that base's production cell performs worse. This also means that the expected total rate at which machines fail at the production cell is smaller than in the case of no backlog.

The exact analysis of a Markov chain model for this system with multiple bases and many machines or with large inventories, is difficult to handle. Therefore, we aggregated a number of states and adjusted some rates to obtain a special near-product-form solution. The new system can be observed as a Typical-server Closed Queuing Network (TCQN). The notion *typical* comes from modeling the central repair facility together with the synchronization queue, as a typical server with state dependent service rates. These state dependent service rates follow from an application of Norton's theorem for Closed Queuing Networks. An adapted Multi-Class

Marginal Distribution Analysis algorithm is developed to compute the steady state probabilities. From these steady state probabilities several performance measures can be obtained, such as the availability and the expected number of machines operating in the production cells. Numerical results show that the approximations are extremely accurate, when compared to simulation results. The approximations are used in an optimization heuristic to determine inventory levels at both the central and local facilities with a maximal total availability under a cost constraint.

A disadvantage of the adapted Multi-Class Marginal Distribution Analysis algorithm is the computational slowness. Especially for large systems with multiple bases, many machines and large inventories, the algorithm is not very fast. Here, further aggregation steps may speed up the system evaluation considerably, unfortunately at the cost of some accuracy.

Furthermore, the model considered is quite a realistic model. However, it could be more realistic by including transport from the bases to the depot and to allow for more complicated networks in the repair facilities. In the model described in this paper, each repair shop is modeled as a multi-server. An interesting extension to this, is to consider the repair facility to be a job shop and model it as a limited capacity open queuing network, as has been done in [3] for the case of an open multi-echelon repairable item system. Then, it is easy to include transport to the depot repair facility as just an additional node in the job shop. Last but not least, it is interesting to find a heuristic to optimize inventory levels at the central and local facilities in combination with optimal repair capacities. This will be the subject of future research.

References

1. Albright SC, Soni A (1988) Markovian multi-echelon repairable inventory system. Naval Research Logistics 35(1): 49–61
2. Albright SC (1989) An approximation to the stationary distribution of a multiechelon repairable-item inventory system with finite sources and repair channels. Naval Research Logistics 36(2): 179–195
3. Avsar ZM, Zijm WHM (2002) Capacitated two-echelon inventory models for repairable item systems. In: Gershwin SB et al. (eds) Analysis and modeling of manufacturing systems, pp 1–36. Kluwer, Boston
4. Buzacott JA, Shanthikumar JG (1993) Stochastic models of manufacturing systems. Prentice-Hall, Englewood Cliffs, NJ
5. Gross D, Kioussis LC, Miller DR (1987) A network decomposition approach for approximate steady state behavior of Markovian multi-echelon repairable item inventory systems. Management Science 33: 1453–1468
6. Harrison PG, Patel NM (1993) Performance modelling of communication networks and computer architectures. Addison Wesley, New York
7. Law AM, Kelton WD (2000) Simulation modeling and analysis, 3rd edn. McGraw-Hill Higher Education, Singapore
8. Rustenburg WD, van Houtum GJ, Zijm WHM (2000) Spare parts management for technical systems: resupply of spare parts under limited budgets. IIE Transactions 32: 1013–1026
9. Sherbrooke CC (1968) METRIC: a multi-echelon technique for recoverable item control. Operations Research 16: 122–141
10. Zijm WHM, Avsar ZM (2003) Capacitated two-indenture models for repairable item systems. International Journal of Production Economics 81–82: 573–588

Appendix

In this appendix, numerical results are given for various parameter settings in our model. In most cases, the availability is high as desired in practical situations. In Table 5 and Table 6 the focus is on the single base model. Multiple base models are considered in Table 7 and Table 8.

Table 5. Results for the simple single base model, $p_1 = 0.5, \lambda_1 = 1, \mu_0 = J_1, \mu_1 = J_1$

J_1	S_0	S_1	A_{exact}	A_{appr}	% dev	$E\underline{j}_1 exact$	$E\underline{j}_1 appr$	% dev
3	1	0	0.5056	0.5100	0.8575	2.3178	2.3225	0.2037
3	3	0	0.5749	0.5771	0.3784	2.4338	2.4368	0.1227
3	5	0	0.5874	0.5880	0.1066	2.4544	2.4553	0.0379
3	1	1	0.7322	0.7340	0.2516	2.6331	2.6345	0.0545
3	3	1	0.7948	0.7961	0.1590	2.7264	2.7279	0.0531
3	5	1	0.8082	0.8087	0.0578	2.7463	2.7469	0.0217
3	1	3	0.9171	0.9172	0.0114	2.8875	2.8873	0.0061
3	3	3	0.9465	0.9466	0.0106	2.9287	2.9287	0.0005
3	5	3	0.9535	0.9536	0.0055	2.9385	2.9385	0.0014
3	1	4	0.9538	0.9538	0.0008	2.9376	2.9374	0.0058
3	3	4	0.9722	0.9722	0.0001	2.9630	2.9629	0.0022
3	5	4	0.9766	0.9766	0.0006	2.9691	2.9691	0.0003
5	1	0	0.4690	0.4722	0.6947	4.1654	4.1688	0.0817
5	3	0	0.5452	0.5470	0.3263	4.3224	4.3250	0.0595
5	5	0	0.5602	0.5607	0.0987	4.3529	4.3538	0.0209
5	1	1	0.7045	0.7059	0.2070	4.5407	4.5416	0.0187
5	3	1	0.7748	0.7758	0.1318	4.6643	4.6654	0.0237
5	5	1	0.7905	0.7909	0.0486	4.6915	4.6920	0.0108
5	1	3	0.9068	0.9069	0.0094	4.8573	4.8570	0.0059
5	3	3	0.9403	0.9404	0.0078	4.9111	4.9110	0.0016
5	5	3	0.9484	0.9484	0.0040	4.9240	4.9240	0.0001
5	1	4	0.9480	0.9480	0.0007	4.9207	4.9205	0.0045
5	3	4	0.9689	0.9689	0.0006	4.9537	4.9536	0.0022
5	5	4	0.9740	0.9740	0.0002	4.9617	4.9617	0.0005
10	1	0	0.4318	0.4339	0.4703	8.9658	8.9676	0.0206
10	3	0	0.5150	0.5162	0.2363	9.1819	9.1836	0.0182
10	5	0	0.5329	0.5333	0.0756	9.2279	9.2286	0.0073
10	1	1	0.6746	0.6756	0.1407	9.4175	9.4177	0.0023
10	3	1	0.7535	0.7542	0.0907	9.5842	9.5848	0.0057
10	5	1	0.7718	0.7721	0.0336	9.6225	9.6228	0.0031
10	1	3	0.8953	0.8953	0.0058	9.8165	9.8161	0.0036
10	3	3	0.9335	0.9335	0.0043	9.8880	9.8879	0.0017
10	5	3	0.9428	0.9428	0.0021	9.9054	9.9053	0.0004
10	1	4	0.9414	0.9414	0.0009	9.8980	9.8978	0.0024
10	3	4	0.9652	0.9652	0.0011	9.9415	9.9414	0.0015
10	5	4	0.9711	0.9711	0.0002	9.9522	9.9522	0.0004

Table 6. Results for the simple single base model, $p_1 = 0.25, \lambda_1 = 1, \mu_0 = 2J_1, \mu_1 = J_1$

J_1	S_0	S_1	A_{exact}	A_{appr}	% dev	$Ej_{\underline{1}}exact$	$Ej_{\underline{1}}appr$	% dev
3	1	0	0.5348	0.5383	0.6612	2.3402	2.3436	0.1475
3	3	0	0.6743	0.6783	0.5878	2.5726	2.5777	0.1978
3	5	0	0.7282	0.7310	0.3796	2.6619	2.6658	0.1468
3	1	1	0.7201	0.7208	0.0913	2.5951	2.5956	0.0176
3	3	1	0.8384	0.8394	0.1194	2.7746	2.7757	0.0405
3	5	1	0.8906	0.8914	0.0958	2.8537	2.8548	0.0381
3	1	3	0.8705	0.8705	0.0007	2.8110	2.8109	0.0007
3	3	3	0.9311	0.9311	0.0019	2.8999	2.8999	0.0001
3	5	3	0.9613	0.9613	0.0023	2.9442	2.9443	0.0007
3	1	4	0.9075	0.9075	0.0001	2.8649	2.8649	0.0003
3	3	4	0.9505	0.9505	0.0000	2.9278	2.9278	0.0002
3	5	4	0.9726	0.9726	0.0002	2.9602	2.9602	0.0000
5	1	0	0.4900	0.4923	0.4515	4.1493	4.1514	0.0483
5	3	0	0.6429	0.6455	0.4015	4.4641	4.4675	0.0762
5	5	0	0.7066	0.7085	0.2621	4.5946	4.5975	0.0620
5	1	1	0.6814	0.6818	0.0605	4.4558	4.4560	0.0042
5	3	1	0.8147	0.8154	0.0774	4.6983	4.6990	0.0142
5	5	1	0.8761	0.8767	0.0617	4.8098	4.8105	0.0149
5	1	3	0.8477	0.8477	0.0002	4.7371	4.7370	0.0005
5	3	3	0.9182	0.9182	0.0007	4.8597	4.8596	0.0003
5	5	3	0.9540	0.9540	0.0011	4.9219	4.9219	0.0001
5	1	4	0.8904	0.8904	0.0002	4.8106	4.8106	0.0002
5	3	4	0.9409	0.9409	0.0002	4.8980	4.8980	0.0002
5	5	4	0.9672	0.9672	0.0000	4.9436	4.9436	0.0001
10	1	0	0.4390	0.4401	0.2503	8.8481	8.8489	0.0094
10	3	0	0.6051	0.6064	0.2198	9.2890	9.2906	0.0176
10	5	0	0.6807	0.6817	0.1415	9.4891	9.4906	0.0158
10	1	1	0.6338	0.6340	0.0316	9.2282	9.2282	0.0000
10	3	1	0.7843	0.7846	0.0387	9.5703	9.5706	0.0024
10	5	1	0.8574	0.8576	0.0300	9.7364	9.7367	0.0032
10	1	3	0.8177	0.8177	0.0001	9.6112	9.6112	0.0003
10	3	3	0.9007	0.9007	0.0001	9.7898	9.7898	0.0002
10	5	3	0.9440	0.9440	0.0002	9.8828	9.8828	0.0001
10	1	4	0.8675	0.8675	0.0002	9.7170	9.7170	0.0001
10	3	4	0.9277	0.9277	0.0002	9.8460	9.8460	0.0001
10	5	4	0.9597	0.9597	0.0001	9.9146	9.9146	0.0001

Table 7. Parameter settings for test problems multi-base model with transportation (2)

Problem	Depot				Base	J_ℓ	S_ℓ	λ_ℓ	μ_ℓ	R_ℓ	p_ℓ	γ_ℓ	ρ_ℓ
	S_0	μ_0	R_0	ρ_0									
21	5	10	1	0.5	1/2	5	5	1	5	1	0.5	10	0.5
22	3	10	1	0.8	1/2	5	2	1	2	1	0.2	∞	0.5
23	3	10	2	0.4	1/2	5	2	1	2	2	0.2	∞	0.25
24	3	10	2	0.4	1/2	5	2	1	2	2	0.2	5	0.25
25	2	5	1	1	1/2	5	1	1	5	1	0.5	∞	0.5
26	2	3	3	0.56	1/2	5	3	1	2	3	0.5	5	0.42
27	4	2	10	0.25	1/2	5	2	1	5	1	0.5	10	0.5
28	8	5	3	0.33	1/2	5	2	1	5	1	0.5	10	0.5
29	8	1	8	0.63	1/2	5	2	1	5	1	0.5	10	0.5
30	3	10	1	1.05	1/2	7	3	1	5	1	0.25	∞	0.35
31	3	10	1	0.75	1	5	1	1	5	1	0.5	∞	0.5
					2	10	3	1	10	1	0.5	∞	0.5
32	3	5	1	0.68	1	2	1	1	2	1	0.5	∞	0.5
					2	8	3	1	8	1	0.7	∞	0.7
33	1	10	1	0.6	1	5	2	1	5	1	0.5	∞	0.5
					2	7	2	1	5	1	0.5	∞	0.7
34	8	5	3	0.5	1/2/3	5	2	1	5	1	0.5	10	0.5
35	1	4	8	0.23	1/2/3	5	1	1	2	3	0.5	10	0.42
36	3	4	8	0.25	1	2	1	2	3	1	0.5	5	0.67
					2	5	1	1	2	3	0.5	10	0.42
					3	7	1	1	5	3	0.5	10	0.23
37	5	3	7	0.9	1	7	5	1	3	3	0.5	10	0.39
					2	7	5	2	3	3	0.2	10	0.31
					3	7	5	3	3	7	0.8	10	0.8
38	5	5	2	1.05	1	7	0	1	5	2	0.5	10	0.35
					2	7	5	1	5	2	0.5	10	0.35
					3	7	10	1	5	2	0.5	10	0.35
39	2	5	2	0.45	1	3	2	1	5	1	0.5	5	0.3
					2	3	2	1	5	2	0.5	5	0.15
					3	3	2	1	5	3	0.5	5	0.1
40	2	5	4	0.5	1/2/3/4	5	2	1	5	2	0.5	10	0.25

Table 8. Results for test problems from Table 7

Problem	Base	$A_\ell sim$	$A_\ell appr$	% dev	$E\underline{j}_\ell sim$	$E\underline{J}_\ell appr$	% dev
21	1/2	(0.9826,0.9854)	0.9840	0.00	(4.9737,4.9792)	4.9765	0.00
22	1/2	(0.8151,0.8266)	0.8192	0.20	(4.7062,4.7304)	4.7129	0.11
23	1/2	(0.9720,0.9742)	0.9731	0.01	(4.9650,4.9690)	4.9669	0.00
24	1/2	(0.8559,0.8607)	0.8563	0.23	(4.8108,4.8181)	4.8118	0.05
25	1/2	(0.5462,0.5522)	0.5526	0.61	(4.1962,4.2112)	4.2061	0.06
26	1/2	(0.8487,0.8510)	0.8493	0.06	(4.7788,4.7833)	4.7804	0.01
27	1/2	(0.8567,0.8614)	0.8594	0.04	(4.7882,4.7982)	4.7931	0.00
28	1/2	(0.8704,0.8752)	0.8714	0.16	(4.8103,4.8195)	4.8113	0.08
29	1/2	(0.8526,0.8606)	0.8555	0.13	(4.7798,4.7945)	4.7851	0.04
30	1/2	(0.6480,0.6813)	0.6608	0.58	(6.2491,6.3370)	6.2806	0.20
31	1	(0.7250,0.7325)	0.7305	0.25	(4.5786,4.5933)	4.5884	0.05
	2	(0.8776,0.8871)	0.8813	0.12	(9.7689,9.7944)	9.7783	0.03
32	1	(0.7985,0.8068)	0.8019	0.09	(1.7560,1.7676)	1.7607	0.06
	2	(0.7977,0.8020)	0.7994	0.05	(7.6077,7.6190)	7.6096	0.05
33	1	(0.8511,0.8587)	0.8561	0.14	(4.7765,4.7885)	4.7849	0.05
	2	(0.6898,0.6971)	0.6933	0.02	(6.4198,6.4366)	6.4237	0.07
34	1/2	(0.8676,0.8718)	0.8711	0.25	(4.8051,4.8128)	4.8109	0.08
35	1/2/3	(0.5041,0.5130)	0.5109	0.46	(4.2436,4.2618)	4.2558	0.07
36	1	(0.6456,0.6538)	0.6525	0.43	(1.5538,1.5658)	1.5638	0.26
	2	(0.5754,0.5821)	0.5790	0.04	(4.3828,4.3952)	4.3883	0.02
	3	(0.7056,0.7089)	0.7070	0.04	(6.5989,6.6031)	6.6016	0.01
37	1	(0.9577,0.9617)	0.9599	0.02	(6.9352,6.9433)	6.9400	0.01
	2	(0.7820,0.7903)	0.7859	0.03	(6.5683,6.5911)	6.5778	0.03
	3	(0.4492,0.4542)	0.4510	0.15	(5.8151,5.8303)	5.8196	0.05
38	1	(0.1745,0.1807)	0.1766	0.59	(5.0709,5.1143)	5.0859	0.13
	2	(0.8530,0.8624)	0.8575	0.03	(6.7166,6.7389)	6.7280	0.00
	3	(0.9845,0.9862)	0.9848	0.05	(6.9731,6.9760)	6.9738	0.01
39	1	(0.9430,0.9450)	0.9443	0.04	(2.9308,2.9337)	2.9326	0.01
	2	(0.9649,0.9671)	0.9670	0.10	(2.9595,2.9624)	2.9622	0.04
	3	(0.9674,0.9686)	0.9686	0.07	(2.9627,2.9644)	2.9644	0.03
40	1/2/3/4	(0.9250,0.9273)	0.9268	0.07	(4.9047,4.9083)	4.9074	0.02

A heuristic to control integrated multi-product multi-machine production-inventory systems with job shop routings and stochastic arrival, set-up and processing times*

P.L.M. Van Nyen[1], J.W.M. Bertrand[1], H.P.G. Van Ooijen[1], and N.J. Vandaele[2]

[1] Technische Universiteit Eindhoven, Department of Technology Management, Den Dolech 2, Pav. F-14, P.O. Box 513, 5600 MB Eindhoven, The Netherlands (e-mail: {p.v.nyen,j.w.m.bertrand,h.p.g.v.ooijen}@tm.tue.nl)
[2] University of Antwerp, Antwerpen, Belgium (e-mail: nico.vandaele@ua.ac.be)

Abstract. This paper investigates a multi-product multi-machine production-inventory system, characterized by job shop routings and stochastic demand interarrival times, set-up times and processing times. The inventory points and the production system are controlled integrally by a centralized decision maker. We present a heuristic that minimizes the relevant costs by making near-optimal production and inventory control decisions while target customer service levels are satisfied. The heuristic is tested in an extensive simulation study and the results are discussed.

Keywords: Production-inventory system – Queueing network analyser – Production control – Inventory control – Performance analysis

1 Introduction

This paper addresses the problem of determining optimal inventory and production control decisions for an integrated production-inventory (PI) system in which multiple products are made-to-stock through a functionally oriented shop. As can be seen in Figure 1, inventory is carried to service customer demand. The customers require that their orders are serviced with a target fill rate. Unsatisfied demand is backordered. Customers arrive according to a renewal process that is characterised by interarrival times with a known mean and squared coefficient of variation (scv).

* The authors would like to thank two anonymous referees and the editor for many valuable suggestions.

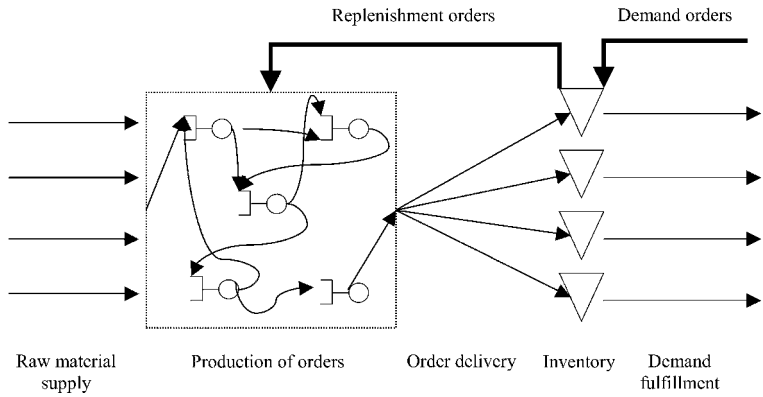


Fig. 1. Integrated production-inventory system with job shop routings

The inventory points generate replenishment orders that are, in this integrated PI system, equivalent to production orders. There is a fixed cost incurred every time a production order is generated. In this paper, all inventory points are controlled using periodic review, order-up-to (R, S) inventory policies. Other inventory policies can be embedded in the framework, if desired. The production orders are manufactured through the shop. We assume ample supply of raw material. The production system consists of multiple functionally oriented workcenters through which a considerable number of different products can be produced. Each of the products can have a specific serial sequence of production steps, which results in a job shop routing structure. The production orders for different products compete for capacity at the different workcenters, where they are processed in order of arrival (FCFS priority). Before starting the production of an order, a set-up that takes a certain time and cost is performed. The total production time of a production order depends on its size. When the production of the entire order is finished, it is moved to the inventory point where the products are temporarily stored until they are requested by a customer. We consider situations in which the average demand for end products is relatively high and stationary. Since the production system is characterized by considerable set-up times and costs, the products are produced in batches. We assume that a centralized decision maker controls both the inventory points and the production system. Then, the objective of the centralized decision maker is to minimize the sum of fixed set-up costs, work-in-process holding costs and final inventory holding costs. Moreover, we impose that customer demand has to be serviced with a target fill rate. The decision variables that can be influenced by the decision maker are the review periods and the order-up-to levels of the products.

Typically, integrated PI systems with job shop routing structure and stochastic demand interarrival times, set-up times and processing times can be found in metal or woodworking companies, e.g. the suppliers of the automotive or aircraft building industry. In particular, the advent of Vendor Managed Inventory (VMI) has forced manufacturing companies to integrate production and inventory decisions.

It appears that it is impossible to analyse this integrated PI system exactly and consequently, it is very difficult (if possible) to solve the optimization problem faced

by the centralized decision maker to optimality. Therefore, we propose a heuristic method that allows us to determine the near-optimal production and inventory control decisions. To this end, we apply and integrate aspects from production control and inventory theory. We follow the basic idea of Zipkin [34]: we use standard inventory models to represent the inventory points of every product and standard results on open queueing networks to represent the production system. Next, we link these submodels together appropriately. Our model differs from Zipkin's model in several ways. Firstly, we consider periodic review (R, S) policies, instead of continuous review (s, Q) policies. Secondly, we use the fill rate as measure of customer service, instead of a backorder cost. Thirdly, our multi-workcenter production facility is modeled as a general open queueing network, instead of an open Jackson network. The general open queueing network model allows us to account more accurately for the effect of batching on the arrival and production processes. In our heuristic, the production capacity is explicitly modeled because the inventory control and the production control are interrelated and depend both on the production capacity. The inventory policy determines the review periods, which on their turn determine the order throughput times in the production system. Based on these throughput times, the safety stock can be set at the inventory points. This reasoning shows how the production and inventory system form one integrated system. If both subsystems are controlled in isolation, a sequential control approach can be used (see e.g [24]). The review periods are set without assessing the impact on the production system (e.g. using the economic order quantity). Next, one observes the throughput times that result from the selected review periods. Based on the observed throughput times, the safety stocks can be set. The costs resulting from this sequential decision-making approach typically exceed the costs of the integrated control approach since treating the subsystems in isolation leads to suboptimality.

The integrated production-inventory control approach proposed in this paper is a simplified version of the hierarchical production control approach advocated by Lambrecht et al. [16] for the control of real-life make-to-order job shops. Their hierarchical control approach consists of three important decisions: (i) lot sizing decisions; (ii) determination of the release dates of production orders; and (iii) sequencing decisions. The control approach proposed in this paper focuses on the decisions at the highest level of the control hierarchy: the lot sizing decisions. In our control approach, the lot sizes are determined by setting the review periods for all products in the PI system. Note that the production system is characterized by considerable set-up times. Therefore, by setting the review periods the decision maker allocates the available production capacity to the different products. We propose an approximate analytical model that allows determining the review periods that minimize the total relevant costs. The approximate analytical model is an extension of the lot sizing procedure based on open queueing networks developed in [16]. Their lot sizing procedure is used to minimize the expected lead times in a make-to-order job shop. We propose an approximate analytical model that takes into account many of the characteristics of the integrated PI system. Most importantly, it explicitly deals with the interaction between the production orders for the different products in the job shop, assuming a FCFS priority rule for the sequencing of the production orders. In this way, the approximate analytical model takes into account

the influence of the review periods on the capacity utilization of the workcenters and on the order throughput times (thus dealing with the multi-product aspect of the PI system under study). Similarly to [13, 15] and [16], we observe a convex relationship between the review periods and the order throughput times. The control decisions situated at the lower level of the hierarchical framework of Lambrecht et al. [16], the determination of the release dates and the sequencing decisions, are dealt with in a straightforward way. Firstly, the production orders are released immediately. Secondly, the sequencing of all orders is done using the FCFS priority rule. The use of the FCFS rule allows for the queueing network to be analyzed using standard queueing network analysers. However, from production control theory it is known that other sequencing policies (priority rules or scheduling methods) may lead to substantial time and cost savings. An overview of priority rules and scheduling methods can be found in [21]. We believe that sequencing policies, other than the FCFS priority rule, can be embedded in the hierarchical control framework in the same way as Lambrecht et al. [16] incorporate the shifted bottleneck procedure [1] for the scheduling of production orders. Observe that by choosing reasonable – not necessarily optimal – review periods or lot sizes, the sequencing decisions situated lower in the control hierarchy become significantly easier to make.

To the best of our knowledge, this paper is the only research that studies the specific problem of minimizing the total relevant costs in integrated multi-product multi-machine PI systems with job shop routing structure and stochastic demand interarrival times, set-up times and processing times. For related problems, however, solution methods have been developed. First, for the deterministic version of this problem, Ouenniche et al. [19, 20] present heuristics that are based on cyclical production plans. The cost-optimal cyclical plans are generated using mathematical programming techniques. We believe that it is not possible to apply the results of Ouenniche et al. directly in our setting because the influence of variability in the demand and manufacturing processes makes the proposed production plans infeasible.

A second relevant contribution is the literature review on the stochastic lot scheduling problem (SLSP) by Sox et al. [25]. This paper gives an extensive overview of current research on the control of multi-product PI systems in which the production system consists of a single workcenter. The critical assumption in this body of research is that the performance of the production system is determined by a single bottleneck process. Although this assumption may be valid in some situations, it is certainly not in others. The heuristic proposed in this paper explicitly considers situations in which the production system consists of multiple workcenters.

Thirdly, Benjaafar et al. [4, 5] study managerial issues related to PI systems, such as the effect of product variety and the benefits of pooling. Benjaafar et al. [4] proposes a method to jointly optimize batch sizes and base-stock levels for a multi-product single workcenter integrated PI system controlled by continuous review (s, Q) policies.

Fourthly, Amin and Altiok [3] and Altiok and Shiue [2], study production allocation issues in multi-product PI systems. More specifically, they address such issues as “when to switch production to a new product” and “what product to switch

to". They propose to handle the first issue with a continuous review inventory control policy. The second issue is resolved by using switching policies that are based on priority structures. Amin and Altiok [3] use simulation to compare a number of manufacturing strategies and switching policies for a serial multi-stage production system with finite interstage buffer space. Altiok and Shiue [2] develop an approximate analytical model to compute the average inventory levels under different priority schemes for a single machine production system.

Fifthly, Rubio and Wein [22] study a PI system where the production system is modeled as a Jackson queueing network. Under this assumption, they derive a formula characterizing the optimal base stock level. The main difference with our approach is that they do not have batching issues in their model because of the absence of set-up times and set-up costs. Note that it is precisely the batching issue that makes the problem very hard to solve.

As a sixth contribution, we mention the work of Lambrecht et al. [16] who study the control of a stochastic make-to-order job shop. They describe a lot sizing procedure that minimizes the expected lead times and thus, the expected work-in-process costs. To this end, they model the production environment as a general open queueing network. Vandaele et al. [28] successfully implemented the method described in Lambrecht et al. [16] to solve lot sizing problems with the medium-sized metal working company Spicer Off-Highway. Their research shows that the lot sizing procedure is capable of solving realistic, large-scale problems. Our research builds on the work of Lambrecht et al. We extend their make-to-order model by including inventory points so that we obtain an integrated PI system in which products are made-to-stock instead of made-to-order.

Seventhly, Bowman and Muckstadt [7] present a production control approach for cyclically scheduled production systems that are characterized by demand and process variability. The production management can delay the release of material to the floor and increase production in a cycle to anticipate on demand and to avoid overtime in future cycles. The control approach uses estimates for the cycle time and the task criticality that are obtained from a Markov chain model. Using these estimates, the control approach seeks a trade-off between inventory holding costs and overtime costs.

Finally, Liberopoulos and Dallery [18] propose a unified modeling framework for describing and comparing several production-inventory control policies with lot sizing. The control method used in this paper is based on (R, S) inventory policies and is comparable to the Reorder Point Policies (RPPs) described in [18]. More insights on RPPs can be found in their paper. Our work differs from their work in several aspects. First, they study a N-stage serial PI system through which a single product is manufactured while we focus on a single stage PI system in which multiple products are produced. Secondly, Liberopoulos and Dallery use queueing network representations to define (not analyze or optimize) several production-inventory control policies that decide when to place and release replenishment orders at each stage. Our work focuses on a (R, S) inventory rule based control policy for which we not only define the control policy in place, but also present a heuristic to make the production and inventory control decisions that minimize the relevant costs.

The remainder of this paper is organized as follows: Section 2 presents the formal problem statement; Section 3 proposes a heuristic to determine review periods and order-up-to levels that minimize the relevant costs; in Section 4 the performance of the heuristic is tested in an extensive simulation study; in Section 5, the results of the simulation study are discussed; and finally, Section 6 summarizes the major findings of our paper.

2 Formal problem statement

First, we introduce the notation used in this paper. Then, we derive formulas for the different cost components. After this, a formal problem definition is given.

2.1 Notation

General input variables:

- P : number of products;
- M : number of workcenters in the production system;
- A_i^D : interarrival times of demand for product i (stochastic variable);
- α_i^* : target fill rate for product i;

Cost related input variables:

- c_i : fixed set-up costs incurred for one production order of product i;
- v_{ij} : echelon value of one item of product i at workcenter j;
- v_i : end value of product i;
- r : inventory holding cost per unit of time;

Tactical control variables:

- R_i : review period of product i;
- S_i : order-up-to level of product i;
- ss_i : safety stock for product i;

Performance related output variables:

- T_{ij} : throughput time of production orders for product i at workcenter j (stochastic variable);
- α_i : realized fill rate for product i;

Mathematical operators:

- E[.] : expectation of a stochastic variable;
- σ^2[.] : variance of a stochastic variable;
- c^2[.] : squared coefficient of variation of a stochastic variable.

2.2 Modeling the cost components

In a periodic review policy, a replenishment order for product i is placed every R_i time units. Consequently, the number of orders placed per time unit is given by R_i^{-1}, so that the total set-up costs per time unit for product i are given by:

$SC_i = c_i R_i^{-1}$.

We use Little's law to compute that the average number of items of product i at workcenter j equals $\frac{E[T_{ij}]}{E[A_i^D]}$. Multiplying the average work-in-process at a machine with the holding cost and summing over all machines leads to the total work-in-process cost for product i:

$$WIPC_i = \sum_{j=1}^{M} \frac{E\left[T_{ij}\right]}{E\left[A_i^D\right]} v_{ij} r\,.$$

The final inventory cost for product i is given by the formula below [24]. The term between brackets gives the average amount of final inventory at inventory point i, which consists of half the average order quantity plus the safety stock.

$$FIC_i = \left(\frac{R_i}{2E\left[A_i^D\right]} + ss_i \right) v_i r.$$

The total cost for product i is simply the sum of its components. Clearly, the total cost TC for the whole PI system is given by the sum over all products of the total costs for each product:

$$TC = \sum_{i=1}^{P} \left(SC_i + WIPC_i + FIC_i\right) \tag{1}$$

2.3 Formal problem statement

In this system, we have one centralized decision maker who wants to minimize the total costs of the PI system. As stated in the introduction, the total costs consist of set-up costs, final inventory holding costs and work-in-process holding costs. The decision maker has to ensure that the target fill rates are satisfied and that the review periods are positive. Consequently, the mathematical formulation of the optimization problem can be stated as:

$$\min_{R_i} \sum_{i=1}^{P} \left[c_i R_i^{-1} + \sum_{j=1}^{M} \frac{E\left[T_{ij}\right]}{E\left[A_i^D\right]} v_{ij} r + \left(\frac{R_i}{2E\left[A_i^D\right]} + ss_i \right) v_i r \right] \tag{2}$$

subject to:

1. $\alpha_i \geq \alpha_i^*$ for $i = 1, ..., P$
2. $R_i > 0$ for $i = 1, ..., P$

Observe that one can easily compute most of the cost components if the review periods for all the products are given. However, two variables – the throughput time

in the workcenters T_{ij} and the safety stock ss_i – cannot be computed analytically. In order to find an expression for the throughput times T_{ij} in the production system, a general open queueing network should be solved. Unfortunately, no exact results for the throughput times in such a queueing system are known. Consequently, it is also impossible to find an exact expression for the safety stock ss_i since ss_i depends on the average and the variance of the throughput times. In conclusion, it is impossible to derive exact expressions for these variables and this implies that our objective function is analytically intractable. Therefore, we have to rely on estimates to evaluate the cost of a given solution.

3 Heuristic to determine review periods and order-up-to levels

In this section, we present a three-phase heuristic that allows finding near-optimal review periods and order-up-to levels. The heuristic is based on an integrated view of the inventory and production system and takes into account all relevant costs, including work-in-process and safety stock costs. Moreover, the heuristic simultaneously considers cost and capacity aspects. This results in a solution that is near-optimal in terms of costs and feasible with respect to production capacity.

Given that the exact analytical evaluation of the objective function is mathematically intractable, we have to use estimation methods to evaluate and optimize the objective function. Two estimation methods can be used to estimate the relevant costs in the PI system under study: simulation and approximate analytical models. Simulation is an accurate estimation method and therefore, simulation-based optimization techniques can be used to accurately solve the optimization problem. For details on simulation-based optimization, see e.g. [17]. Unfortunately, these techniques are very expensive in terms of computation time. This often prohibits the use of simulation based optimization techniques, even for medium-sized problems. Alternatively, it is possible to optimize our problem using an approximate analytical model. The main advantage of an approximate analytical model is the low amount of computation time required. Obviously, the price of the gain in speed is a certain degree of inaccuracy. To have the best of both worlds, we propose a three-phase heuristic that combines an approximate analytical model with simulation techniques. The simulation techniques are used to circumvent some of the inaccuracies due to the approximate analytical model.

Our heuristic is presented graphically in Figure 2. In the optimization phase, the heuristic determines near-optimal review periods and initial order-up-to levels based on an approximate analytical model. The approximate analytical model is designed so that it captures the most essential characteristics of the PI system while it can be optimized fast using a greedy search algorithm. Unfortunately, the use of an approximate model may result in suboptimal control decisions. Also, the initial order-up-to levels computed by the approximate model may be insufficient to meet the target fill rates. Therefore, the second phase of the heuristic uses simulation techniques to fine-tune the order-up-to levels. Finally, in the third phase of the heuristic, the costs and operational characteristics (fill rates, throughput times, etc.) are estimated accurately with simulation. The remainder of this section discusses the three phases of the heuristic in more detail.

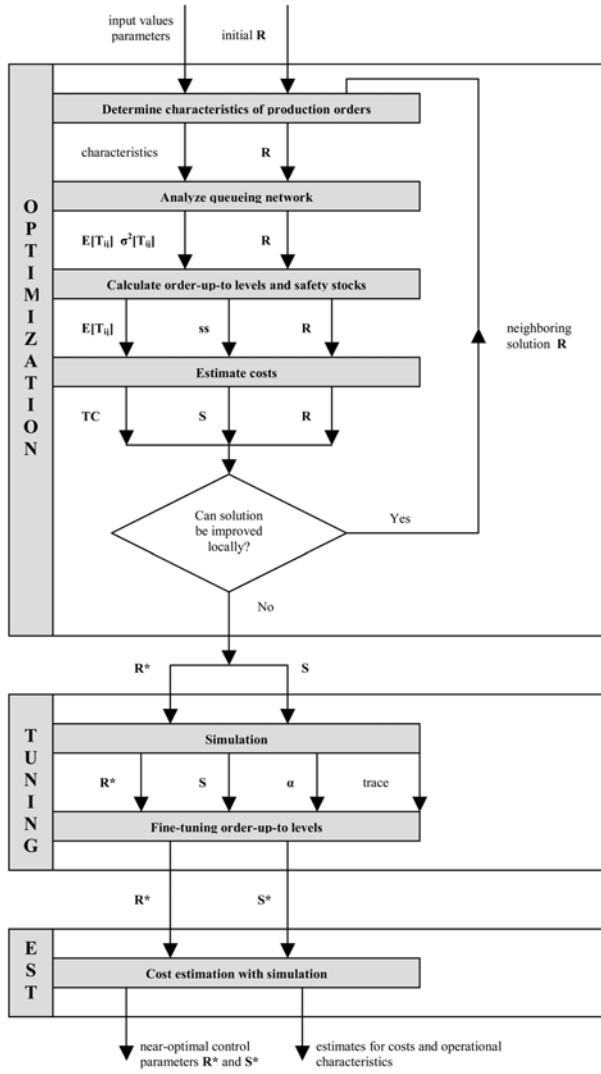


Fig. 2. Outline of three-phase heuristic

3.1 Optimization phase

In the first phase of the heuristic, near-optimal tactical inventory and production control decisions are determined. The optimization tool consists of two main elements: (i) an analytical model that approximates the relevant costs given a vector of review periods and (ii) a greedy search method that finds the vector of review periods that minimizes the relevant costs. In the subsections below, both elements are described in more detail.

The approximate analytical model follows the same basic idea as Zipkin [34]. First, we use a standard inventory model to represent the (R, S) inventory policy

of every product, see e.g. [24]. Next, we use standard results on open queueing networks to represent the production system. Our open queueing network is solved using the Queueing Network Analyser developed by Whitt [32]. Similarly to [16], the expressions for the performance measures of the queueing network are written as a function of the production lot size. Finally, we link both submodels together using concepts from renewal theory; see [8]. The resulting analytical model is an approximation to the real PI system. Similarly to the approach proposed by Zipkin, we sacrifice accuracy for the sake of simplicity and computational tractability.

3.1.1 Approximate analytical model. In this section we present an approximate analytical model to estimate the relevant costs in the PI system under study, given a set of review periods $R = (R_1, ..., R_i, ..., R_P)$. The analytical model consists of four successive steps.

Step 1: Determine characteristics of production orders

We start by analysing the generation of replenishment orders by the inventory points. Note that in the PI system under study, generating a replenishment order at an inventory point results in placing a production order to the production system. By analysing the characteristics of the replenishment orders, we therefore implicitly analyse the characteristics of the production orders that arrive to the production system. In our approximation model, we focus on two main characteristics of the production orders: the time between the arrivals of two successive production orders of product i, referred to as the interarrival time A_i^P, and the processing time of a production order for product i on machine j, denoted as P_{ij}^P. We limit ourselves to the determination of the expectation $E[.]$ and variance $\sigma^2[.]$ of the interarrival times and the processing times. In the case of an (R_i, S_i)-inventory policy, a production order of variable size is generated at every review moment, i.e. every R_i time units. Therefore, the expectation and variance of the interarrival times of production orders are given by: $E[A_i^P] = R_i$ and $\sigma^2[A_i^P] = 0$. The production orders for product i are of variable size, which we denote here by N_i. By applying limiting results from renewal theory [8] we obtain that the number of arrivals N_i in a review period R_i is approximately normally distributed with mean $E[N_i] = R_i E^{-1}[A_i^D]$ and variance $\sigma^2[N_i] \approx R_i c^2[A_i^D] E^{-1}[A_i^D]$. Note that the normal approximation for the number of arrivals in a review period is only justified if the review period is relatively long or the arrival rate is relatively high, since there must be a significant number of arrivals within a review period. Since we are concerned with situations in which the average demand for end products is relatively high, the use of the normal approximation is acceptable here.

From these expressions, we can derive the mean and variance of the production order processing times, but first we introduce some additional notation:

P_{ij} : processing time of one item of product i at machine j;

L_{ij} : set-up time of production orders of product i at machine j.

The expected processing time of an order of product i on machine j is given by the expected total processing time plus the expected set-up time, i.e. $E[P_{ij}^P] = R_i E^{-1}[A_i^D] E[P_{ij}] + E[L_{ij}]$. We assume that the processing times of single units

are independent and identically distributed (i.i.d.) and independent of the set-up time. Then, the variance of the net processing times, excluding set-up time, equals the variance of the sum of a variable number of variable processing times, which can be computed with a formula given by e.g. [24]. Consequently, the variance of the processing times of the orders of product i at machine j is given by:

$$\begin{aligned}&\sigma^2\left[P_{ij}^P\right] = E\left[N_i\right]\sigma^2\left[P_{ij}\right] + E^2\left[P_{ij}\right]\sigma^2\left[N_i\right] + \sigma^2\left[L_{ij}\right] \\ &= R_i E^{-1}\left[A_i^D\right]\sigma^2\left[P_{ij}\right] + E^2\left[P_{ij}\right] R_i c^2\left[A_i^D\right] E^{-1}\left[A_i^D\right] + \sigma^2\left[L_{ij}\right]\end{aligned} \tag{3}$$

Step 2: Analyse queueing network

The second step in the approximate analytical model uses the characterization of the production orders to compute performance measures of the production system. Based on the characterization of the production orders, we can model the production system as a general open queueing network in which the arrival and production processes of the orders have known first and second moments. From the late seventies on, extensive research has been executed on the estimation of performance measures in such queueing systems. Our procedure is based on the queueing network analyser developed by Whitt [32]. The lot sizing procedure proposed in Lambrecht et al. [16] is also based on Whitt [32]. However, our approach differs from the work of Lambrecht et al. in two ways: (i) they use a simplified expression for the scv of the aggregated arrival process, whereas we use Whitt's original approximation; (ii) we use an improved expression for the scv of the departure processes that is due to [33], whereas Lambrecht et al. use an adapted version of Shantikumar and Buzacott [23]. Van Nyen et al. [29] present simulation results on the estimation performance of the queueing network analyzer, which indicate that serious estimation errors may occur. The queueing network analyser allows us to find approximations for the expectation and variance of the throughput times of product i at the different machines j in the production system, i.e. $E[T_{ij}]$ and $\sigma^2[T_{ij}]$. In order to approximate the throughput times in the entire production system, Whitt [32] assumes that the throughput times at different machines are independent. Then, the expectation and variance of the total throughput time of a production order are given by: $E[T_i] = \sum_{j=1}^{M} E[T_{ij}]$ and $\sigma^2[T_i] = \sum_{j=1}^{M} \sigma^2[T_{ij}]$. For more details on the use of the queueing network analyser, see [26], [30] or [31].

Step 3: Calculate order-up-to levels and safety stocks

In the third step of the approximate analytical model, we determine the order-up-to levels $S = (S_1, ..., S_i, ..., S_P)$ using standard inventory theory. The reorder level is set such that the customer demand is satisfied with a target fill rate α_i^*. We need a characterisation of the customer demand $D_i^{T_i+R_i}$ during the throughput time T_i and the review period R_i to determine the appropriate order-up-to level. Note that the customer demand $D_i^{T_i+R_i}$ is related to the time between successive demand arrivals A_i^t. Using renewal theory, we obtain expressions for $E[D_i^{T_i+R_i}]$ and $\sigma[D_i^{T_i+R_i}]$. See [31] for more details.

Then, the order-up-to level S_i can be determined by:

$$S_i = E[D_i^{T_i+R_i}] + k_i\sigma[D_i^{T_i+R_i}] \tag{4}$$

where k_i is the so-called safety factor for product i that depends on the target fill rate α_i^*. Given a target fill rate, [24] presents a very accurate rational approximation for k_i in the case of normally distributed demand. Finally, the safety stock ss_i for product i can be computed as: $ss_i = k_i \sigma[D_i^{T_i+R_i}]$. This step results in a vector of order-up-to levels $S = (S_1, ..., S_i, ..., S_P)$ that correspond to the vector of review periods $R = (R_1, ..., R_i, ..., R_P)$.

Step 4: Estimate costs

In the previous steps, we presented an approach to approximate the expected throughput times $E[T_{ij}]$ and the safety stocks ss_i, given a vector $\mathbf{R} = (R_1, ..., R_i, ..., R_P)$. Using the expressions for the different cost components given in Section 2.1 we can now compute the total costs TC corresponding to a solution R.

3.1.2 Optimization of tactical control parameters. In the previous subsection, we presented an approximate analytical model to estimate the cost of a given set of review periods $\mathbf{R} = (R_1, ..., R_i, ..., R_P)$. In this subsection, we try to find the vector of review periods $\mathbf{R}^* = (R_1^*, ..., R_i^*, ..., R_P^*)$ that minimizes the total relevant costs. Unfortunately, we cannot prove the unimodality of the total cost function in terms of the review periods $\mathbf{R}$. However, extensive tests allow us to postulate that the objective function, when estimated with the approximate analytical model, is unimodal in the review periods. We ground this unimodality-postulate on the observation that (i) using a greedy search algorithm with different starting solutions always resulted in the same final solution; (ii) using simulated annealing, a general optimization technique for non-convex functions (see e.g. [9]), never resulted in a better solution than the greedy search algorithm. Moreover, our postulate is consistent with [27] where it is postulated that the expected throughput times are convex in the lot sizes.

Based on the unimodality-postulate, we propose to use a simple greedy search algorithm for the minimization of the relevant costs, called *univariant search parallel to the axes*. This approach fixes all review periods but one and performs a direct search along this variable until the minimum of the objective function in the current direction has been found. This minimum is then used as the starting point for the next iteration. Again, all review periods but one are fixed and a direct search is performed. This process is repeated until the value of the objective function cannot be further improved. The final solution $\mathbf{R}^*$ cannot be improved in any direction parallel to the axes and is the solution proposed by the greedy search heuristic. The performance of the greedy search heuristic has been tested against a simulated annealing algorithm. For all instances in the test bed, the greedy search algorithm outperformed simulated annealing.

3.2 Tuning phase

In the first phase of the heuristic, we use an approximate analytical model to determine the near-optimal review periods $\mathbf{R}^*$ and initial settings for the order-up-to levels $\mathbf{S}$. Because of the use of approximations, the realized fill rates may be lower

than the target fill rates. Given the constraints on the fill rate in the formal problem statement this may result in the infeasibility of the solution. Also, it may happen that the realized fill rates are higher than the target fill rates. Obviously, this leads to a solution that is unnecessarily expensive. For these reasons, we add a second step to the heuristic in which the order-up-levels are fine-tuned.

In this second step of the heuristic, we use a procedure proposed by Gudum and de Kok [10]. Their procedure builds on the following observation. Given that the inventory points are controlled by periodic review (R, S) policies with full backordering, the size and the timing of replenishment orders is determined by the review periods only. In an integrated PI system, this implies that the arrivals of production orders to the production system, as well as the processing times of the production orders at the different workcenters are determined by the review intervals and not by the order-up-to-levels. Therefore, the throughput times of the replenishment orders are completely determined by the review intervals. This also implies that a change in the order-up-to levels only influences the customer service levels. In conclusion, this observation states that a given selection of the review periods fully determines the stochastic behavior of the PI system and that the order-up-to levels can be adjusted to achieve a certain customer service level without affecting the behavior of the PI system. For more details on this observation, see [10].

In our heuristic, the fine-tuning phase starts by simulating the initial solution to estimate the realized fill rates corresponding to the solution **(R*, S)**. Based on a trace of the inventory levels in the first simulation run, a procedure developed in [10] is used to fine-tune the order-up-to levels. Note that we do not adjust the review periods. The procedure makes use of the observation above, stating that changes in the order-up-to levels only influence the fill rates. This procedure allows us to set the order-up-to levels to the lowest possible levels that satisfy the fill rate constraints, denoted as $\mathbf{S}^*$. This results in the solution $(\mathbf{R}^*, \mathbf{S}^*)$. From a computational point of view, the procedure developed in [10] is attractive because it uses only one simulation run instead of iterative simulation runs.

3.3 Estimation phase

In the third and final phase of the heuristic, a simulation experiment is used to estimate the costs and operational performance characteristics corresponding to the solution $(\mathbf{R}^*, \mathbf{S}^*)$. Simulation is used because of its estimation accuracy. All the costs that are listed in Section 2.2 are estimated. The operational performance characteristics that are estimated include the fill rates, the throughput times at the different workcenters and the utilization of the workcenters.

4 Testing the heuristic in a simulation study

The heuristic uses an approximate analytical model to determine the tactical production and inventory decisions, which may result in suboptimal decisions. In this section we test the performance of our heuristic in an extensive simulation study. A specific problem instance is studied in detail in order to gain understanding of

the mechanisms embedded in the optimization tool. Furthermore, we compare our heuristic to two simulation based optimization methods. Since simulation based optimization techniques allow for the accurate optimization of the objective function, the comparison enables us to assess the quality of our heuristic. This section is organized as follows. We first present the experimental design of our simulation study. Then, a specific problem instance is studied into detail. Finally, we compare the performance of our heuristic with the simulation based optimization methods.

4.1 Experimental design of the simulation study

In the simulation study, we consider an integrated PI system with 10 products and 5 workcenters. We assume that the customer demands arrive according to a Poisson process. Furthermore, the set-up times and processing times are exponentially distributed, leading to phase-type production order processing times. This assumption allows incorporating all kinds of variability that are present in real production systems: operator influences, workcenter defects, etc. The parameter values in our experimental design are based on data from two medium-sized metalworking companies.

In the simulation study, we vary four factors over several levels:

- net utilization of the workcenters $\rho^{net}(0.65, 0.75, 0.85)$;
- set-up times L_{ij} (randomly generated in the intervals $[30, 60]$ min. or $[90, 180]$ min.);
- set-up costs c_i (randomly generated in the intervals €$[0, 0]$, €$[6.67, 13.33]$, €$[20, 40]$ or €$[60, 120]$);
- target fill rates $\alpha_i^*(0.90, 0.98)$.

The number of combinations that can be made with the levels of the four factors equals $3 \times 2 \times 4 \times 2 = 48$ combinations. We use a procedure presented in Appendix 1 to generate five random instances for each combination of the levels of the four factors. Therefore, the total simulation study consists of $5 \times 48 = 240$ instances.

In order to reduce the computation time required for the optimization phase of the heuristic, we restrict the value of the review periods to multiples of 100 minutes. Since the objective function is flat around the optimum, this restriction has a negligible impact on the total cost of a solution. In the second and third step of our heuristic, we use a simulation model that is built in Simula. Simula is a general-purpose simulation language, for more details see [6]. We use the batch-means method with 10 subruns to find performance estimates. The length of the subruns is chosen so that at least 100,000 customer orders for each product arrive. The review moments are initialized by letting them start at a random moment in the interval $[0, R_i]$ for $i = 1, ..., P$. This ensures that no special patterns are built into the order generation process and into the arrival process of orders to the production system.

4.2 Mechanisms embedded in the approximate analytical model

In this section, we discuss how the approximate analytical model uses the review periods to minimize the total relevant costs. The mechanisms behind the selection of the review periods are illustrated by comparing the detailed output of the optimization tool with the output of a simple heuristic method for setting the review periods. More specifically, we use the economic order quantity (EOQ) expressed as a time supply to set the review periods, see e.g. [24]:

$$R_i = \sqrt{\frac{2c_i E\left[A_i^D\right]}{v_i r}} \tag{5}$$

The EOQ formula completely ignores the impact of the lot sizing decision on the production system. Hence, it does not take into account the costs that are related to capacity utilization and throughput times, i.e. work-in-process and safety stock costs. We use the uncapacitated EOQ approach to solve one random set of 48 problem instances of the experimental design. Below, the results for all 48 problem instances are summarized, but first we study in detail one specific problem instance. This problem instance is selected because it clearly demonstrates how our heuristic works. In this way, the reader can gain understanding of the mechanisms that are embedded in the heuristic. The selected problem instance is characterized by: $\rho^{net} = 0.85$; $L_{ij} \in [90, 180]$ min.; $c_i \in [6.67, 13.33]$ €; $\alpha_i^* = 0.98$.

In Tables 1 up to 3, we present the detailed output of the simulation of the heuristic (HEU) and the uncapacitated EOQ method for this problem instance. From the analysis of the decision variables and the corresponding performance measures, we learn how the optimization tool works and how it tries to achieve the minimal total relevant costs. Table 1 displays the decision variables (review periods and order-up-to levels) and the resulting throughput times, characterized by their expectation E[T] and standard deviation $\sigma[T]$. It can be seen from Table 1 that the heuristic proposes considerably smaller review periods than the uncapacitated EOQ method. The order-up-to levels are lowered accordingly. Remark that the review periods of the different products are decreased in a non-proportional way in order to account for the specific processing characteristics of every product. The impact of the smaller review periods on the expectation and standard deviation of the throughput times is high: the expected throughput times decrease by 36.5% on average while the standard deviations of the throughput times decrease by 42.4% on average.

Table 2 shows the impact of the changes in the decision variables on the relevant costs. Since the solution of the heuristic uses considerably smaller review periods, the set-up costs are substantially higher compared to the uncapacitated EOQ solution (+58.4%). However, since the review periods chosen by the heuristic lead to shorter and more reliable throughput times, the work-in-process costs and the final inventory holding costs decline significantly (−35.8% and −36.5%). Overall, this leads to a cost decrease realized by the heuristic versus the uncapacitated EOQ approach of 9.0%.

Finally, Table 3 gives insight into the mechanisms embedded in the optimization tool. We use elementary insights from queueing theory to illustrate the trade-offs

Table 1. Decision variables and throughput times for one problem instance: heuristic vs. uncapacitated EOQ method

Prod.	HEU				EOQ			
	R	S	$E[T]$	$\sigma[T]$	R	S	$E[T]$	$\sigma[T]$
1	3700	665	3784.2	653.1	8157	1326	6718.3	1362.6
2	4800	608	3844.4	865.7	6178	827	5385.9	1307.7
3	4600	657	5107.2	974.5	7449	1028	7812.7	1574.1
4	5400	536	2291.0	628.9	6359	679	3062.3	1012.5
5	5000	807	3639.5	857.0	5703	1092	5424.3	1481.8
6	4000	860	5028.4	923.8	7605	1490	8259.6	1539.9
7	5100	561	2781.4	782.0	7534	825	4045.0	1185.1
8	7000	574	2422.7	673.6	7289	703	3461.8	1329.2
9	4300	662	4763.3	852.6	7000	1061	7518.3	1472.9
10	3700	411	3455.2	694.8	8095	846	6780.5	1452.2
Avg.	4760	634.1	3711.7	790.6	7136.9	987.7	5846.9	1371.8

Table 2. Cost components for one problem instance: heuristic vs. uncapacitated EOQ method

Prod.	HEU			EOQ		
	OC	FIC	WIPC	OC	FIC	WIPC
1	8867.4	2526.5	2337.8	4021.7	5475.6	4183.7
2	3790.3	3230.4	2183.1	2945.1	4416.5	3079.6
3	5571.8	3242.3	2867.6	3440.8	5123.0	4410.6
4	3277.1	2849.1	1257.7	2783.0	3681.2	1681.3
5	4325.4	4521.5	2909.1	3792.2	6161.7	4257.8
6	8224.6	3579.1	3793.2	4325.6	6332.5	6192.0
7	4709.4	2842.7	1409.5	3188.0	4203.9	2050.2
8	3393.5	3581.2	1340.6	3259.2	4523.1	1960.0
9	4820.3	2670.3	2648.0	2960.6	4331.3	4224.1
10	6096.1	1877.1	1555.8	2786.5	3929.4	3055.4
Tot.	53076.0	30920.2	22302.3	33502.6	48178.1	35094.6
Overall	106298.4			116775.2		

that are made by the heuristic. From elementary queueing theoretical results for the GI/G/1 queue, e.g. Hopp and Spearman [12], we learn that there are four main elements affecting the expectation of the throughput times E[T] on the machines: (i) utilization ρ; (ii) variation of the arrivals c_a^2; (iii) average processing time t_p;

Table 3. Operational characteristics of production system for one problem instance: heuristic vs. uncapacitated EOQ method

Mach.nr.	HEU				EOQ			
	ρ	c_a^2	t_p	c_p^2	ρ	c_a^2	t_p	c_p^2
1	0.91	0.7015	582.5	0.046	0.89	0.6735	867.7	0.082
2	0.93	0.6834	547.9	0.047	0.90	0.7526	807.2	0.082
3	0.89	0.6146	1022.6	0.024	0.88	0.5964	1563.9	0.050
4	0.90	0.4963	806.3	0.045	0.88	0.6533	1256.1	0.180
5	0.92	0.691	636.2	0.053	0.89	0.7967	1070.7	0.133
Avg.	0.91	0.637	719.1	0.043	0.89	0.695	1113.12	0.105

(iv) variation of the processing times c_p^2. This insight is based on the Kingman approximation for the expectation of the throughput times in a GI/G/1 queue [14]:

$$E[T] = \left(\frac{c_a^2 + c_p^2}{2}\right)\left(\frac{\rho}{1-\rho}\right)t_p + t_p \tag{6}$$

From Table 3, it can be seen that the optimization tool adapts the review periods so that the variation of the arrivals and the expectation and the variation of the processing times are reduced. However, this happens at the expense of increased utilization levels. We may conclude that the optimization tool 'harmonizes' the review periods of the different products so as to obtain the best balance between utilization and variability.

In the job shop production system under study, the departure process of a machine is the arrival process to the next machine in the routing of a product. An elementary approximation, due to Hopp and Spearman [12], for the scv of the departure process leaving a queue is:

$$c_d^2 = \rho^2 c_p^2 + \left(1-\rho^2\right)c_a^2 \tag{7}$$

From this approximation, it can be observed that when the utilization of the machines is high, it is important to achieve low variation in the processing times in order to obtain an arrival process with low variability to the next machine. Table 3 shows that the heuristic realizes a low variation in the processing times, while the utilization levels are high (around 90%). From Table 1, it can be seen that the actions taken by the heuristic, based on the mechanisms presented above, result in shorter and less variable throughput times. Note that the mechanisms described above are embedded in the proposed approximate analytical model using advanced queueing theoretical results developed in [32, 33].

Now, we briefly present the results for the 48 instances that were solved using the uncapacitated EOQ method. In 14 out of 48 problem instances, the uncapacitated EOQ approach resulted into a solution that is infeasible with respect to production capacity. For the 34 feasible instances, the uncapacitated EOQ solution is on average 5.2% more expensive than the solution of the heuristic. The maximum cost increase reported on this set of experiments is 10.1%. The conclusion of

these experiments is that the uncapacitated EOQ method may work relatively well compared to the heuristic, but since the uncapacitated EOQ approach does not take into account capacity issues, it may result in unnecessarily expensive solutions or in solutions that are infeasible with respect to production capacity (and that require capacity expansion in the form of overwork, outsourcing, etc.).

In order to avoid that infeasible solutions are obtained, one can add capacity restrictions to the uncapacitated EOQ method. Doing so, we obtain the following mathematical programming problem, which we call the capacitated EOQ approach:

$$\begin{aligned} &\min_{R_i} \quad \sum_{i=1}^{P} \left[\frac{c_i}{R_i} + \frac{v_i r}{2E\left[A_i^D\right]} R_i \right] \\ &subject\ to: \\ &1. \quad \sum_{i=1}^{P} \left[\frac{E\left[P_{ij}\right]}{E\left[A_i^D\right]} + \frac{E\left[L_{ij}\right]}{R_i} \right] \leq \rho_j^{\max} \quad for\ j = 1, \ldots, M \\ &2. \quad R_i > 0 \qquad\qquad\qquad\qquad for\ i = 1, \ldots, P \end{aligned} \tag{8}$$

The objective function of this mathematical program is identical to the cost function of the uncapacitated EOQ method. The first set of constraints imposes that the machine utilization must be lower than a maximum allowable utilization level $\rho_j^{\max}$. The second set of constraints states that the review periods should be strictly positive. Note that the objective function and the constraints are convex in the review periods R_i. This convex programming problem can easily be solved to optimality using the commercially available CONOPT algorithm. The CONOPT algorithm attempts to find a local optimum satisfying the Karush-Kuhn-Tucker conditions. It is well known that for convex programming problems a local optimum is also the global optimum (see e.g.[11]).

The main difficulty that arises with this capacitated EOQ method is the choice of the maximum allowable utilization level $\rho_j^{\max}$. For deterministic problems $\rho_j^{\max}$ is usually chosen so that all production capacity is utilized, i.e. $\rho_j^{\max} = 1$. Clearly, in stochastic settings $\rho_j^{\max}$ should be lower than 1 for reasons of stability. It is, however, not obvious how the precise value of $\rho_j^{\max}$ should be chosen. If $\rho_j^{\max}$ is chosen too low, this leads to long review periods (in order to reduce the capacity utilization due to set-ups) and thus to high cycle stocks. On the contrary, if $\rho_j^{\max}$ is chosen too high, this results in high congestion, large amounts of work-in-process, long throughput times and high safety stocks. A priori, the EOQ model is not able to predict which value of $\rho_j^{\max}$ leads to the lowest total relevant costs. Therefore, in our experiments we vary $\rho_j^{\max}$ over a range of reasonable values and observe the resulting total relevant costs.

We use the capacitated EOQ method to solve the 48 problem instances that were also solved using the uncapacitated EOQ method. The value of $\rho_j^{\max}$ is set to 0.90, 0.95 and 0.99. Let us define the deviation in the total costs between the capacitated EOQ method and our heuristic as:

$$\Delta = \frac{TC^{eoq} - TC^{heu}}{TC^{heu}} \times 100\%.$$

Table 4. Summary statistics for Δ, the relative deviation in total costs between the capacitated EOQ method and the proposed heuristic (instances with set-up costs > 0)

		$\rho_j^{\max} = 0.90$	$\rho_j^{\max} = 0.95$	$\rho_j^{\max} = 0.99$
$\rho^{net} = 0.65$	min.	1.9	1.9	1.9
	avg.	3.7	3.7	3.7
	max.	5.4	5.4	5.4
$\rho^{net} = 0.75$	min.	1.8	1.7	1.7
	avg.	5.0	5.0	5.0
	max.	7.5	7.5	7.5
$\rho^{net} = 0.85$	min.	5.7	4.2	3.8
	avg.	25.2	7.8	6.7
	max.	80.2	10.7	10.1

Table 5. Summary statistics for Δ, the relative deviation in total costs between the capacitated EOQ method and the proposed heuristic (instances with set-up costs $= 0$)

		$\rho_j^{\max} = 0.90$	$\rho_j^{\max} = 0.95$	$\rho_j^{\max} = 0.99$
$\rho^{net} = 0.65$	min.	3.3	7.1	121.2
	avg.	3.8	9.0	157.6
	max.	4.6	11.0	224.5
$\rho^{net} = 0.75$	min.	17.5	1.8	71.5
	avg.	19.2	3.1	84.5
	max.	21.2	4.5	96.2
$\rho^{net} = 0.85$	min.	106.8	15.0	11.7
	avg.	114.6	17.9	13.8
	max.	119.1	20.4	16.0

Tables 4 and 5 give the minimum, average and maximum of Δ, respectively for the instances with set-up costs > 0 and the instances with set-up costs = 0. The results are shown for the different levels of the net utilization of the machines ρ^{net}.

The results in Tables 4 and 5 show that the proposed heuristic always outperforms the capacitated EOQ method. The capacitated EOQ approach may work reasonably well, provided that a good choice is made for $\rho_j^{\max}$: the lowest Δ observed in this set of instances is 1.7%. However, one can also observe that an inappropriate choice of $\rho_j^{\max}$ may result in a very poor performance: the maximum of Δ in this set of instances is 224.5%. As mentioned before, the EOQ approach does not provide any guideline for choosing the value of $\rho_j^{\max}$.

For the instances with set-up costs > 0 and $\rho_j^{\max} = 0.99$, the performance of the capacitated EOQ method seems reasonable: the average of Δ is 5.1% with a

maximum of 10.1%. It appears that in the majority of these instances the capacity constraints are non-binding so that the solution of the capacitated EOQ method is identical to the solution of the uncapacitated EOQ method. When $\rho_j^{\max}$ is lowered, the performance of the capacitated EOQ method degrades for the instances with high ρ^{net}. When $\rho^{net} = 0.85$ and $\rho_j^{\max} = 0.90$, the average of Δ is 25.2% with a maximum of 80.2%. For the majority of the instances with low and moderate ρ^{net}, the capacity constraints are also non-binding when $\rho_j^{\max} = 0.90$ and 0.95. Therefore, in these instances the performance of the capacitated EOQ method is similar to that of the uncapacitated EOQ method and the capacitated EOQ method with $\rho_j^{\max} = 0.99$.

For the instances with set-up costs $= 0$, it seems to be even more important to select the appropriate value of $\rho_j^{\max}$ than in the case of set-up costs > 0. For example, when $\rho^{net} = 0.65$, the capacitated EOQ method works well when $\rho^{\max} = 0.90$: the average of Δ is 3.8% with a maximum of 4.6%. If $\rho^{\max}$ is chosen too high, the performance of the capacitated EOQ method degrades strongly: for $\rho^{\max} = 0.99$, the average of Δ is 157.6% with a maximum of 224.5%. Similar results hold for the instances with $\rho^{net} = 0.75$. For the instances with $\rho^{net} = 0.85$, the capacitated EOQ approach performs rather poorly for all choices of $\rho^{\max}$: the minimum of Δ reported on this set of instances is 11.7%.

The main conclusion from these experiments is that the capacitated EOQ approach is very sensitive to the choice of $\rho^{\max}$. Since the appropriate value of $\rho^{\max}$ depends on the specific characteristics of the problem instance, it is difficult to develop a general rule of thumb for selecting $\rho^{\max}$. The heuristic proposed in this paper does not suffer from this problem. The approximate analytical model embedded in the heuristic explicitly models the impact of the review periods on capacity utilization and on congestion phenomena, taking into consideration the specific characteristics of the problem instance. Therefore, our heuristic is a more robust and reliable method to set the decision variables.

4.3 Testing the quality of the heuristic

In this section, we test the optimization quality of the heuristic. The methodology used for this test warrants some discussion. First, note that the optimal solution for the problem under study is unknown. Furthermore, no high-quality bounds on the optimal costs are available, mainly due to the difficulty to find bounds on the waiting times in the production system. Moreover, to the best of our knowledge, no other control approaches have been developed for the integrated PI system with job shop routings and stochastic arrival, processing and set-up times. Consequently, the heuristic solution cannot be compared to the true optimum, nor to a good bound, nor to another control approach reported in the literature. In short, there exists no good benchmark to test the quality of our heuristic. Therefore, we constructed our own benchmarks in order to test the performance of the heuristic.

First, we test the prediction quality of the approximate analytical model. If the prediction quality of the approximate model is satisfactory, then one may expect that the optimization quality of the tool is good. However, when the approximate analytical model wrongly estimates the absolute value of the costs, but correctly

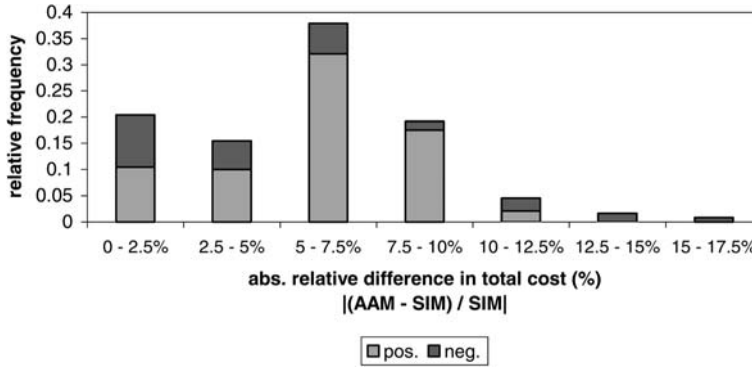


Fig. 3. Frequency diagram for relative difference between total costs approximate analytical model (AAM) and simulation (SIM)

captures the relative behavior of the costs in function of the review periods, the optimization process may still perform well. In Figure 3, we present the relative difference between the cost estimates of the approximate analytical model (AAM) and simulation (SIM) for the solutions proposed by the heuristic for the 240 instances. Since for optimization purposes mainly the absolute value of the relative deviation is relevant, we group the positive and negative intervals with the same absolute value. The frequency of negative and positive values of the relative deviation is shown in distinctive colors. From Figure 3, it can be seen that the relative approximation error is rather small, lying in the range of -17% to 12% on this set of 240 experiments. From this figure, it also can be seen that for the vast majority of the instances (more than 92%) the absolute relative approximation error is lower than 10%. Negative relative differences occur, but in more than 70% of the cases the relative error is positive. From these results, we conclude that the estimation quality of the approximate analytical model is satisfactory.

Secondly, in order to test the optimization quality of the heuristic we develop two simulation based optimization methods to solve several instances of our optimization problem. Simulation based optimization is known to be an accurate but time-consuming optimization method, see e.g. Law and Kelton [17]. The performance of the simulation based optimization methods is compared to the performance of our heuristic in terms of the optimization quality as well as the required computation time.

We use two different simulation based optimization methods:

- A modification of the greedy search algorithm presented in Section 3.1.2. We use three different step sizes to perform the search along the axes. For each of the step sizes, we use the greedy search algorithm. The solution of one phase is used as an input for the next phase. The step sizes for the review periods are 2500, 500 and 100 minutes.
- OptQuest, a commercially available software package developed by Glover et al. OptQuest combines elements of scatter search, taboo search and neural networks to find solutions for non-convex optimization problems [17]. We limit the review periods to the interval [0.5, 2] times the review periods $\mathbf{R}^*$ proposed

by our heuristic. Moreover, we use a step size of 100 minutes for the review periods.

Both optimization techniques suggest new vectors $\mathbf{R}'$ that need to be evaluated using simulation. In our research, we use the second and the third phase of our heuristic to evaluate the vector $\mathbf{R}'$. The evaluation of a vector $\mathbf{R}'$ consists of a tuning phase, see Section 3.2, in which the correct order-up-to levels are computed based on a simulation run. Next, the total cost of the solution $\mathbf{R}'$ is estimated using a second simulation run in the evaluation phase presented in Section 3.3.

In order to reduce the computation time required, we use the solution of the heuristic as initial solution. The simulation based optimization methods are then used to improve this initial solution. Furthermore, we limit the number of simulation subruns to 5. However, even with these measures, the simulation based optimization techniques take very large amounts of computation time. For this reason, it is impossible to use the simulation based optimization techniques for all 240 instances in the experimental design. Instead, we select 15 worst-case instances for which we apply simulation based optimization techniques.

The selection of the 15 worst-case instances warrants some discussion. Let us first introduce a lower bound for the total costs in the PI system under study. The lower bound neglects the impact of the variability in the system as well as the interaction between different products. More details on the lower bound can be found in Appendix 2. The computation of the lower bound is only possible for the 180 instances with set-up costs $c_i > 0$. We compute the relative deviation e_1 between the total cost of our heuristic and the lower bound. On the set of 180 instances, e_1 usually lies in the interval 10–30% with an average of 20%. In this paper, we use e_1 as an indicator for the potential improvement that can be realised when simulation based optimization techniques are used. We select 10 instances with the largest e_1 to be optimized using simulation based optimization techniques. Since these instances have the largest potential for improvement, we call them worst-case instances. We ensure that only one of the five random instances of the same combination of levels of the four factors is selected as a worst-case instance. For the case of set-up costs = 0, we cannot compute the lower bound presented above. Therefore, we use another criterion to select the instances. Since the optimization quality of the heuristic depends on the accuracy of the approximate analytical model presented in Section 3.1.1, we select the 5 instances with the largest deviation between cost estimate of the approximate model and the total costs estimated with simulation for the optimal solution $\mathbf{R}^*$ proposed by the heuristic. The selected instances are worst-case instances on the criterion of approximation performance. Again, we ensure that only one of the five random instances of the same combination of levels is chosen. In this way, 15 worst-case instances are selected.

Table 6 summarizes the findings of the simulation based optimization experiments for the 15 worst-case instances. It presents the optimal total costs of our heuristic TC^{heu}, the greedy search algorithm TC^{gs} and the OptQuest algorithm TC^{optq}. Also the 90% confidence interval on the total costs is given. Finally, Table 6 presents the relative improvement of the greedy search and OptQuest algorithm with regard to the solution of the heuristic, defined as:

Table 6. Results of simulation based optimization experiments

Exp. nr.	TC^{heu}	CI (90%)	TC^{gs}	CI (90%)	i_1 (%)	TC^{optq}	CI (90%)	i_2 (%)	max(i_1,i_2) (%)
1	172843.8	426.6	169700.5	722.7	1.8	170462.7	335.5	1.4	1.8
2	166937.9	288.5	164988.8	403.1	1.2	163182.4	153.9	2.2	2.2
3	180776	308.7	179496.9	305	0.7	175361.6	391.3	3.0	3.0
4	308863.3	508.5	305763.6	759.4	1.0	303366.3	587	1.8	1.8
5	193689.6	650.5	187546.9	838.7	3.2	191255.4	790.8	1.3	3.2
6	119521.1	592.6	118594.7	938	0.8	118768.7	657.5	0.6	0.8
7	103428.2	270.3	101259.9	240	2.1	101309.4	240.7	2.0	2.1
8	139734.6	717.5	135119.1	1177	3.3	138166.8	622.9	1.1	3.3
9	292435.2	566.0	290025.9	558.7	0.8	286411.8	424.3	2.1	2.1
10	98308.4	143.0	95414.6	100.6	2.9	95801.9	129.4	2.5	2.9
11	21316.6	95.0	21027.7	151.2	1.4	20899.5	127.9	2.0	2.0
12	26897.1	158.0	26503.7	119.7	1.5	26105.3	123.7	2.9	2.9
13	6883.8	25.0	6755.4	18.4	1.9	6752.2	22.8	1.9	1.9
14	8690.8	43.7	8510	37.7	2.1	8665.6	32.7	0.3	2.1
15	13561.9	102.2	12729	72.4	6.1	13189.4	71.3	2.7	6.1
Average					2.0			1.9	2.5

$$i_1 = \frac{TC^{heu} - TC^{gs}}{TC^{heu}} \times 100\% \text{ and } i_2 = \frac{TC^{heu} - TC^{optq}}{TC^{heu}} \times 100\%.$$

On the set of 15 experiments, the greedy search heuristic achieves a 2.0 % improvement on average. The OptQuest technique achieves a 1.9 % improvement on average. Taking the maximum improvement for each instance, we see that the solution of our heuristic can be improved by 2.5 % on average using simulation based optimization techniques. On this set of 15 worst-case instances, the maximum improvement is 6.1 %. Based on these results, we claim that the optimization quality of our heuristic is satisfactory.

Next, we compare the results of the greedy search algorithm and the OptQuest algorithm. Table 6 shows that in 60% of the instances the greedy search algorithm outperforms the OptQuest algorithm, while in 40% of the instances OptQuest outperforms the greedy search algorithm. The difference in optimization performance between both search methods does not exhibit a clear pattern and we were not able to relate it to any of the factors in the study. In our opinion, the difference in performance is due to the fact that both simulation based optimization techniques are heuristics without any performance guarantee. Both simulation based optimization techniques can get stuck into non-optimal solutions, as can be observed in Table 6.

The simulation based optimization techniques require large amounts of computation time: the OptQuest algorithm is stopped after 1000 iterations, resulting in the solutions presented in Table 6. The greedy search algorithm used a variable number of iterations to achieve the results in Table 6: on average 123, with a minimum of 80 and a maximum of 185. Depending on the problem instance, one iteration takes about 2.5 to 4 minutes on an Intel Pentium 4 – 2.00 GHz. processor. On average, the OptQuest algorithm takes about 54 hours to find the solutions presented in Table 6. The greedy search algorithm gives comparable results in a much shorter amount of time: about 6.5 hours on average. Our heuristic, however, is many times faster than the simulation based techniques: it takes about 8 minutes on average to find solutions that are only slightly worse than the solutions found by the simulation based optimization techniques. In conclusion, we can state that our heuristic performs satisfactory: it takes a fraction the time required by simulation based optimization techniques to find solutions that are only slightly worse in terms of total costs.

Table 7 presents the average of the review periods, order-up-to levels and machine utilization for every problem instance solved using simulation based optimization. The first column gives the instance number. The second and third columns present the relative difference between the average of the review periods over all products for the simulation based techniques and the heuristic. The fourth and fifth columns give the relative difference between the average order-up-to levels for the simulation based optimization methods and the heuristic. The sixth and seventh columns give the relative difference in the average of the utilization of the machines in the production system between the simulation optimization and the heuristic. Finally, the eight and ninth columns repeat the relative improvement i_1 and i_2 that is obtained by using the simulation based optimization techniques versus the heuristic.

From the analysis of Table 7, we try to gain understanding of how the three optimization techniques work. Surprisingly, no clear patterns appear in the numerical

Table 7. Relative difference in average review periods, order-up-to levels, utilization and total costs for simulation based optimization vs. heuristic

Exp. nr.	% diff. in avg. review periods		% diff. in avg. order-up-to levels		% diff. in avg. utilization level		% improvement in total costs	
	$\frac{R^x - R^{heu}}{R^{heu}} \times 100\%$		$\frac{S^x - S^{heu}}{S^{heu}} \times 100\%$		$\frac{\rho^x - \rho^{heu}}{\rho^{heu}} \times 100\%$		$\frac{TC^{heu} - TC^x}{TC^{heu}} \times 100\%$	
	gs	optq	gs	optq	gs	optq	gs	optq
1	2.4	6.5	2.2	3.3	−0.5	−0.5	1.8	1.4
2	4.9	6.8	5.6	6.3	−0.7	−1.0	1.2	2.2
3	3.3	6.3	3.9	3.9	−0.1	−0.3	0.7	3
4	4.6	10.1	5.2	7.7	−0.2	−0.2	1	1.8
5	2.6	8.7	−1.0	4.0	−0.3	−0.6	3.2	1.3
6	−3.8	−1.4	−2.3	−1.2	0.4	0.2	0.8	0.6
7	7.7	8.7	4.7	5.1	−1.2	−1.6	2.1	2
8	−8.4	−1.7	−6.0	−1.6	0.7	0.1	3.3	1.1
9	9.5	5.3	8.6	2.7	−0.6	−0.4	0.8	2.1
10	1.7	1.5	−2.6	−1.4	0.0	−0.3	2.9	2.5
11	−3.4	2.2	−1.7	−0.9	1.7	−0.7	1.4	2
12	−1.5	2.6	−1.2	−1.6	0.8	−0.5	1.5	2.9
13	4.1	4.5	0.0	0.0	−0.6	−1.1	1.9	1.9
14	−0.8	2.0	−1.9	0.1	0.1	−0.6	2.1	0.3
15	−2.4	3.0	−4.8	−1.4	1.0	−0.6	6.1	2.7

results in Table 7. The relative improvement does not seem to be directly related to the relative difference in the average review periods and order-up-to levels. Take e.g. instance 4 where the OptQuest algorithm increases the average review periods by 10.1%, while the greedy search algorithm proposes an increase of 4.6%. For this instance the relative improvement realized by the OptQuest algorithm is 1.8% while the relative improvement for the greedy search algorithm is 1.0%. One may conclude that larger differences in the review periods lead to larger cost improvements. However, this conclusion is contradicted by e.g. instances 9 and 15. In instance 9, the greedy search algorithm increases the review periods by 9.5% leading to a cost saving of 0.8%, while the OptQuest algorithm increases the lot sizes by only 5.3% leading to a larger cost saving of 2.1%. In instance 15, the greedy search algorithm obtains a cost improvement of 6.1%, the largest observed on this set of experiments. In order to achieve this cost improvement, the review periods are reduced by 2.4% on average. Furthermore, it can be observed that even if the utilization remains almost unchanged, still a substantial cost improvement can be realized by the harmonization of the review periods. This can e.g. be seen from instance nr. 10 where the utilization does not change for the greedy search algorithm, but the costs improve by 2.9%.

From this analysis, we conclude that there is no direct relation between the average review periods, order-up-to levels, machine utilization and the relative cost improvement that can be obtained from the use of simulation based optimization techniques. This conclusion leads to the obvious question how the relative cost improvement is then realized by the simulation based optimization techniques. We believe that the answer lies in the mechanisms that were described in Section 4.2. Similarly to the approximate analytical model, the simulation based optimization techniques 'harmonize' the review periods of the different products so as to obtain the best balance between utilization and variability. Both simulation based optimization techniques seek the best possible trade-off between the utilization of the machines, variation of the arrivals, average processing times and variation of the processing times. In this way, it is possible that relatively small differences in the average review periods can lead to relatively large cost savings.

5 Discussion of the simulation results

In this section we analyse and interpret the results of the heuristic for the 240 instances in the simulation study. This allows us to verify the soundness of the solutions proposed by the heuristic.

5.1 Main effects of factors

Figures 4–6 show the impact of the four factors on the average costs, review periods and order-up-to levels. The impact of every factor is discussed successively.

The impact of increases in the net utilization on the total costs is about 17 % and 19 % for changes from 0.65 to 0.75 and from 0.75 to 0.85 respectively. The increase of the net utilization from 0.65 to 0.75 causes the review periods to decrease, while

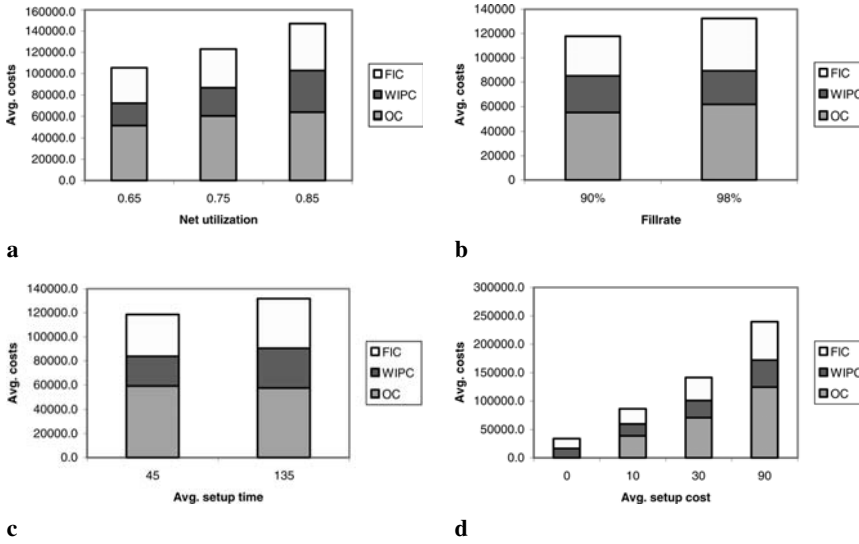


Fig. 4a–d. Impact of factor on average costs: **a** net utilization – **b** fill rate – **c** avg. set-up times – **d** avg. set-up costs

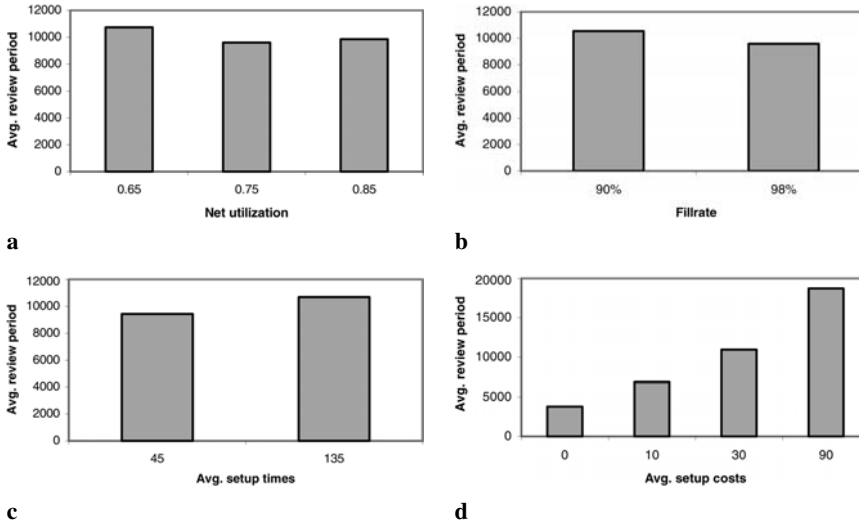


Fig. 5a–d. Impact of factor on average review periods: **a** net utilization – **b** fill rate – **c** avg. set-up times – **d** avg. set-up costs

they increase when the net utilization is increased from 0.75 to 0.85. The reason for this pattern is to be found in the mechanisms embedded in the heuristic, as described in Section 4.2. When the net utilization goes from 0.65 to 0.75, the review periods are decreased in order to reduce throughput times, work-in-process costs and final inventory holding costs. However, a further increase in the net utilization requires that the review periods be slightly increased in order to reduce the impact of the

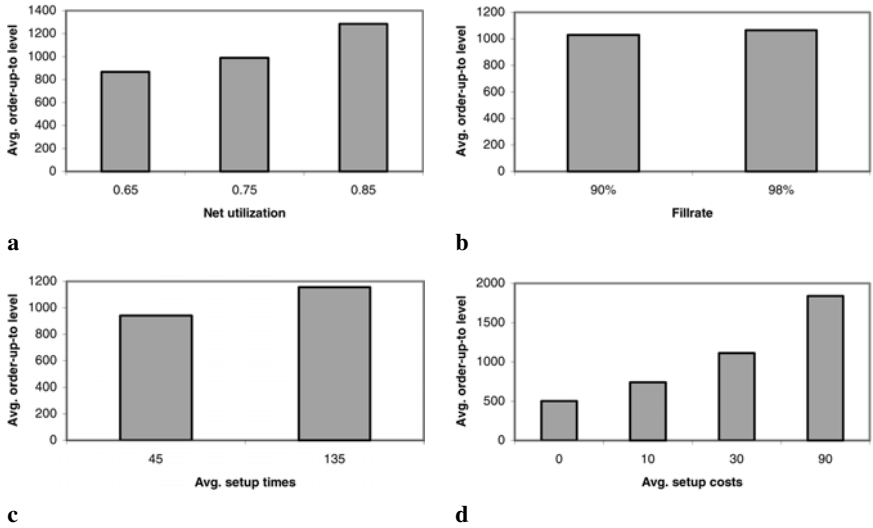


Fig. 6a–d. Impact of factor on average order-up-to levels: **a** net utilization – **b** fill rate – **c** avg. set-up times – **d** avg. set-up costs

set-up times on the capacity utilization. The order-up-to levels on the contrary, increase steadily when the utilization is increased. This can be explained by two effects. First, in the experiments, the rise in the net utilization is caused by increases in the demand rate for the products. The increased demand rate results in higher demand during a review period and increased cycle stock. Secondly, the rise in the net utilization increases the congestion in the system, leading to longer order throughput times and higher safety stocks.

When the fill rates increase from 90 % to 98 %, the average total costs increase with 11%. In order to account for the increase in the fill rate, the order-up-to levels are increased. However, the increase in the order-up-to levels is fairly small. This is due to the fact that the heuristic proposes smaller review periods, which lead to lower cycle stock. Moreover, the order throughput times are reduced so that less safety stock is required.

The increase of the average set-up times from 45 to 135 leads to an increase in the total costs of 12%. The review periods are raised to limit the impact on the capacity utilization. The increased review periods lead to lower set-up costs, but also to longer throughput times and higher work-in-process costs. Because of the increases in the review periods and the throughput times, the order-up-to levels and the final inventory costs increase.

The set-up costs appear to be the dominant factor in our experimental design: when the average set-up costs increase from 0 to 10, the average total cost rises with 156%. Further increases in the average set-up costs result in cost increases of 64 % and 70 %. The review periods and the order-up-to levels increase in a similar fashion. From these results, it appears that efforts to cut set-up costs (as advocated by e.g. the Just-In-Time philosophy) may effectively result in large savings in the overall costs. Figure 4-d presents the division of the total costs (TC) over the different

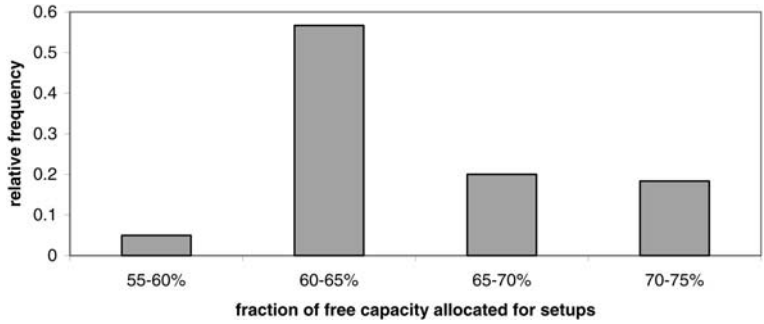


Fig. 7. Frequency diagram of allocation of free capacity for set-ups (instances with set-up costs $= 0$)

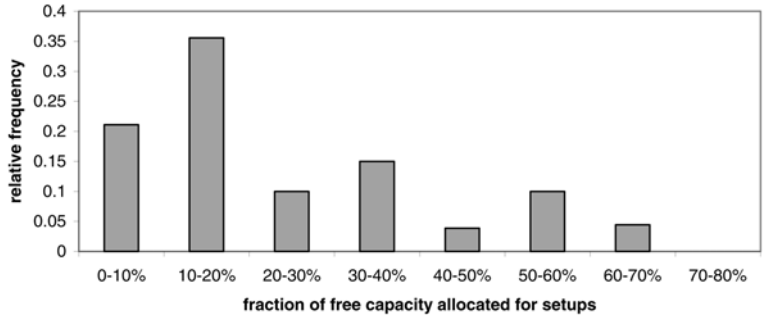


Fig. 8. Frequency diagram of allocation of free capacity for set-ups (instances with set-up costs > 0)

cost components. For the instances with set-up costs $= 0$, the heuristic proposes solutions in which the final inventory costs (FIC) and the work-in-process costs (WIPC) are almost balanced, the former being slightly dominant. For the instances with set-up costs > 0, the heuristic seeks a balance between the fixed set-up costs (SC) and the final inventory and work-in-process holding costs. Remarkably, this result is similar to the well-known Economic Order Quantity model for which the holding costs and fixed ordering costs are the same if the economic order quantity is ordered. Similarly to the instances with set-up cost $= 0$, the final inventory costs dominate the work-in-process costs.

5.2 Allocation of capacity for set-ups

Now we take a look at the fraction of the free capacity, computed as $1 - \rho^{net}$, that is allocated for performing set-ups. From Figure 7 we see that for the instances with set-up costs $= 0$, the allocated fraction of free capacity lies in the interval 60–74%, the average being 65%. As a rule of thumb, it appears that in the case of set-up costs $= 0$ about 2/3 of the free capacity should be allocated for set-up times. However, from the analysis of results of simulation based optimization techniques in Table 7 it appears that it is not only important to select the right level of capacity utilization.

The numerical results in Table 7 indicate that substantial savings can be realised by adjusting the review periods, but keeping the capacity utilization more or less at the same level. Indeed, the harmonization of the review periods is of high importance in the multi-product system under study. Figure 8 shows the case of non-zero set-up costs for which a wide range of allocation of free capacity is observed (3–69%). In general, the allocated fraction of free capacity for set-ups is lower than in the case of zero set-up costs. Moreover, our experiments indicate that the fraction of allocated capacity decreases when the set-up costs increase. This sound behavior is due to increases in the review periods because of rising set-up costs. Obviously, the increases in the review periods lead to lower capacity allocation for set-ups.

5.3 Behavior of review periods when set-up times are changed

Next, we turn our attention to the behavior of the average review periods, order-up-to levels and capacity utilization when the set-up times are tripled from [30, 60] to [90, 180]. The results are displayed in Figures 9 and 10. When set-up costs are zero and the set-up times are tripled, our heuristic increases the review periods and order-up-to levels with almost the same factor as the set-up times ([2.7, 3.2] in our set of experiments). In this way, the fraction of free capacity allocated for set-ups remains almost unchanged. For the instances with set-up costs > 0, a substantial change in set-up times has virtually no impact on the average review periods and order-up-to levels. In the main part of the instances, the proportion lies in the interval [0.9, 1.3]. Therefore, the fraction of free capacity used for set-up times increases with the same factor as the increase in set-up times. From this observation, we conclude that the cost aspect dominates the capacity aspect in the optimization process when the set-up costs > 0. Finally, we observe that these results fade when set-up costs are relatively small, i.e. $L_{ij} \in [6.67, 13.33]$. Especially when the net utilization is high (0.85), an increase in the set-up times leads to rises in the review periods and the

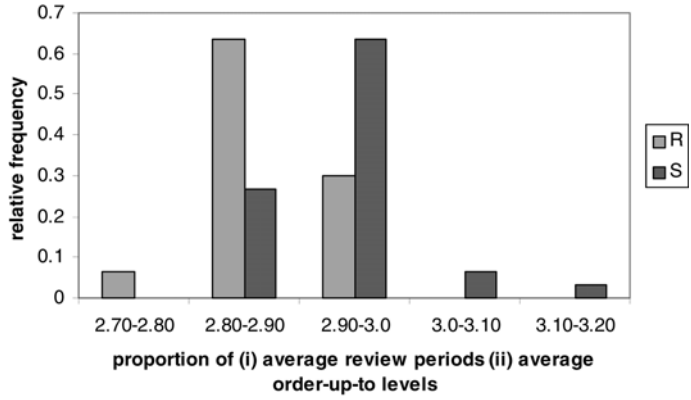


Fig. 9. Frequency diagram of proportion of average review periods for set-up times [90, 180] over [30, 60] (instances with set-up costs $= 0$)

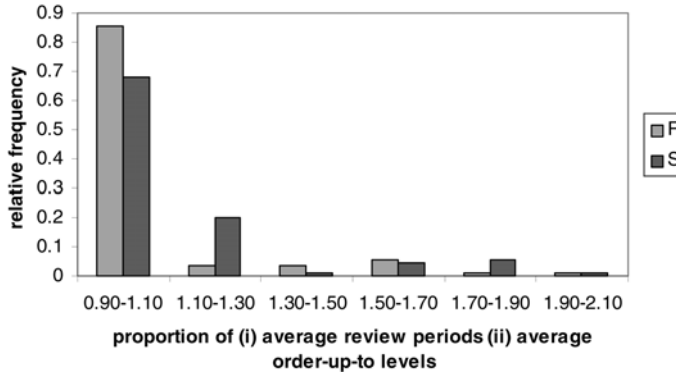


Fig. 10. Frequency diagram of proportion of review periods for set-up times [90, 180] over [30, 60] (instances with set-up costs > 0)

corresponding order-up-to levels. Again, this indicates that our optimization tool works soundly.

From the results in this subsection and the previous subsection, it appears that behaviour of the optimized control parameters is rather different in the case where set-up costs are equal to zero and the case where the set-up costs are larger than zero. In the case that set-up costs are zero, the optimization tool focuses more on the capacity utilization aspect whereas when the set-up costs are larger than zero, the tool is mainly concerned with the cost aspect. The lesson that can be learned from these observations is that it is very important to take into account both cost and capacity issues when making production and inventory control decisions. Decision support systems that focus solely on one of these issues are cursed to make errors that can result in substantial cost increases. Unfortunately it is the case that most decision support systems do focus either on the capacity aspect or on the cost aspect. Finally, these observations illustrate the importance of a good knowledge of the cost structure of the PI system, so that the application of sound management accounting techniques should be given high priority.

6 Conclusions

We propose a three-step heuristic to coordinate production and inventory control decisions in an integrated multi-product multi-machine production-inventory system characterized by job shop routings and stochastic demand, set-up and processing times. Our heuristic minimizes the sum of set-up costs, work-in-process holding costs and final inventory holding costs while stochastic customer demand is satisfied with a target fill rate. The first step uses an approximate analytical model and a greedy search algorithm to find near-optimal control parameters. Several approximations are used in this step. Since this may result in customer service levels that are too low or too high, the order-up-to levels are fine-tuned in the second step. This step ensures that all customer service level requirements are satisfied. In the third step, the performance characteristics of the system are accurately estimated using simulation.

We tested our heuristic in an extensive simulation study, consisting of 240 instances. We selected a subset of 15 worst-case instances that were optimized using our heuristic and two simulation based optimization techniques. The comparison of the performance of our heuristic to the simulation based optimization techniques allowed us to conclude that our heuristic performs satisfactory, both in terms of optimization quality and required computation time.

The detailed analysis of one problem instance helped to gain understanding of the mechanisms that are embedded in the heuristic. It appeared that our optimization tool harmonizes the review periods of the different products so that the variability in the arrival and production processes, the average processing times and the utilization of the workcenters are balanced.

Based on the results of our simulation study, we conclude that the set-up cost is the dominant factor in the study. The impact of the set-up costs on the total costs is many times higher than that of the other factors in the study: utilization, fill rate and set-up times. The results support the insight that substantial cost savings can be realized by set-up cost reduction programs, as advocated by the Just-In-Time philosophy. When compared to the other relevant costs components, i.e. work-in process and final inventory holding costs, the set-up costs are dominant. For the instances with set-up costs > 0, about 50% of the total costs are due to the fixed set-up costs. The work-in-process costs are slightly dominated by the final inventory holding costs, both in the instances with and without set-up costs. When set-up costs are absent, review periods are chosen so that about 2/3 of available capacity is allocated to set-ups. When set-up costs are considerably high, it seems that the review periods are chosen based on cost considerations only. Instances that are in between the both extremes show a trade-off between cost and capacity considerations. These results indicate that a good knowledge of the cost structure of the production-inventory system is of the highest importance in order to make control decisions that minimize the total relevant costs. Unlike other approaches, our heuristic integrates both capacity and cost aspects. Therefore, our heuristic is able to make robust decisions for every instance, regardless of the values of the different cost parameters. Moreover, the heuristic captures the interaction between different products and different workcenters and their impact on the congestion phenomena in the production system. Finally, our heuristic combines the speed of a queueing network model with the accuracy of a simulation experiment. Therefore, it works fast and accurately.

Some interesting directions for further research may be to improve the approximation techniques for open queueing networks, to extend the heuristic to other inventory policies and to test the heuristic in real-life case studies. Furthermore, it would be interesting to compare the performance of the reorder point policy based control method presented in this paper to other control methods. Other control methods can e.g. be based on cyclical production plans or pull-type control. Finally, it can be worthwhile to investigate the influence of different priority rules on the performance of the production-inventory system.

Appendix 1: generation of problem instances

We use the following procedure to generate the problem instances.

1. Randomly generate a set of routings. The routing structures are chosen so that the average number of operations per product equals 3 and the number of operations per product lies in the interval [2, 4]. Furthermore, the number of products per workcenter lies in the interval [4, 8];
2. Allocate to every product i a relative share of capacity utilization of workcenter j, denoted as $rscu_{ij}$. We use the capacity utilization profiles that are presented in Table A.1. These profiles depend on the number of products that are produced at a workcenter, denoted as N_j;
3. Randomly draw the demand for product i $\lambda_i = E^{-1}\left[A_i^D\right]$ from the interval $\left[\lambda^{LB},\ \lambda^{UB}\right]$. This interval is chosen so that the expected item production time $E\left[P_{ij}\right]$ varies between $P^{LB} = 1$ min. and $P^{UB} = 5$ min. Then λ^{LB} and λ^{UB} are given by: $\lambda^{LB} = \frac{\rho^{net} \max_{i,j}(rscu_{ij})}{P^{UB}} = 0.06\,\rho^{net}$ and $\lambda^{UB} = \frac{\rho^{net} \min_{i,j}(rscu_{ij})}{P^{LB}} = 0.1\,\rho^{net}$. If $\rho^{net} = 0.85$, then the yearly demand lies in the interval [26805; 44676].
4. Calculate the average item processing time for every product i and every workcenter j : $E\left[P_{ij}\right] = \frac{rscu_{ij}\rho^{net}}{\lambda_i}$;
5. Generate randomly: $r \in [0.15, 0.25]$ €/(€* year);
6. Generate randomly for every i:
   - $c_i \in$ € [0, 0] or € [6.67, 13.3] or € [20, 40] or € [60, 120];
   - $v_i \in$ € [10, 15];
   - Cost of raw material as a fraction [0.35, 0.50] of v_i;
   - The difference between v_i and the cost of raw material is the added value. To find the echelon value v_{ij} of a product, we distribute the added value equally over the different production steps.
7. Generate randomly for every i and every j:
   - $L_{ij} \in$ [30, 60] min. or [90, 180] min.
8. The length of a simulation subrun is chosen as 100,000* $\max_i E\left[A_i^D\right]$.

Appendix 2: lower bound on total costs

In this appendix we propose a lower bound for the total costs in a production-inventory system with job shop routings and stochastic arrival and processing times. The lower bound neglects the impact of the variability in the system as well as the interaction between different products. The lower bound consists of three terms:

1. final inventory costs. The lower bound is based on an inventory model characterized by deterministic demand and zero replenishment lead times;
2. work-in-process costs due to processing times and set-up times (waiting times are excluded);
3. fixed set-up costs (which are known exactly).

Table A1. Capacity utilization profiles

$\downarrow$ profile nr. $\backslash N_j \rightarrow$	4	5	6	7	8
1	0.30	0.30	0.25	0.20	0.17
2	0.28	0.25	0.20	0.17	0.15
3	0.25	0.20	0.17	0.16	0.15
4	0.17	0.15	0.15	0.14	0.12
5		0.10	0.13	0.12	0.11
6			0.10	0.11	0.10
7				0.10	0.10
8					0.10
Total	1	1	1	1	1

We formulate a lower bound for the total costs for product i for a given review period R_i:

$$LBTC_i\left(R_i\right) = c_i R_i^{-1} + \sum_{j=1}^{M} \left\{ \frac{v_{ij} r}{E\left[A_i^D\right]} \left(\frac{R_i E\left[P_{ij}\right]}{E\left[A_i^D\right]} + E\left[L_{ij}\right] \right) \right\} + \frac{\left(\alpha_i^*\right)^2 R_i v_i r}{2E\left[A_i^D\right]} \tag{9}$$

Based on the formula for $LBTC_i\left(R_i\right)$, the review period R_i^* that minimizes $LBTC_i$ equals:

$$R_i^* = \sqrt{\frac{\left(2E^2\left[A_i^D\right] c_i\right)}{2\sum_{j=1}^{M} E\left[P_{ij}\right] v_{ij} r + \left(\alpha_i^*\right)^2 E\left[A_i^D\right] v_i r}} \tag{10}$$

The computation of the lower bound becomes infeasible if $c_i = 0$. Furthermore, the lower bound for the total costs of the whole system is given by the sum of the lower bounds of the different products since the interaction between the different products is ignored.

References

1. Adams J, Balas E, Zawack D (1988) The shifting bottleneck procedure for job-shop scheduling. Management Science 34: 391–401
2. Altiok T, Shiue GA (2000) Pull-type manufacturing systems with multiple product types. IIE Transactions 32: 115–124
3. Amin M, Altiok T (1997) Control policies for multi-product multi-stage manufacturing systems: an experimental approach. International Journal of Production Research 35: 201–223
4. Benjaafar S, Kim JS, Vishwanadham N (2004) On the effect of product variety in production-inventory systems. Annals of Operations Research 126: 71–101

5. Benjaafar S, Cooper WL, Kim JS (2003) On the benefits of pooling in production-inventory systems. Working paper, University of Minnesota, Minneapolis. Management Science (to appear)
6. Birtwistle GM, Dahl OJ, Nijgaard K (1984) Simula begin. Studentenlitteratur, Lund
7. Bowman RA, Muckstadt JA (1995) Production control of cyclic schedules with demand and process variability. Production and Operations Management 4: 145–162
8. Cox DR (1962) Renewal theory. Methuen, London
9. Eglese RW (1990) Simulated annealing: a tool for operational research. European Journal of Operational Research 46: 271–281
10. Gudum CK, de Kok AG (2002) A safety stock adjustment procedure to enable target service levels in simulation of generic inventory systems. Working paper, BETA Research School, The Netherlands
11. Hillier FS, Lieberman GJ (2005) Introduction to Operations Research, 8th edn. McGraw-Hill, Boston
12. Hopp WJ, Spearman ML (1996) Factory physics. Irwin, Chicago
13. Karmarkar US (1987) Lot sizes, lead times and in-process inventories. Management Science 33: 409–419
14. Kingman JFC (1961) The single server queue in heavy traffic. Proceedings of the Cambridge Philosophical Society 57: 902–904
15. Lambrecht MR, Vandaele NJ (1996) A general approximation for the single product lot sizing model with queueing delays. European Journal of Operational Research 95: 73–88
16. Lambrecht MR, Ivens PL, Vandaele NJ (1998) ACLIPS: A capacity and lead time integrated procedure for scheduling. Management Science 44: 1548–1561
17. Law AM, Kelton WD (2000) Simulation modeling and analysis, 3rd edn. McGraw-Hill, Boston
18. Liberopoulos G, Dallery Y (2003) Comparative modelling of multi-stage production-inventory control policies with lot sizing. International Journal of Production Research 41: 1273–1298
19. Ouenniche J, Boctor F (1998) Sequencing, lot sizing and scheduling of several products in job shops: the common cycle approach. International Journal of Production Research 36: 1125–1140
20. Ouenniche J, Bertrand JWM (2001) The finite horizon economic lot sizing problem in job shops: the multiple cycle approach. International Journal of Production Economics 74: 49–61
21. Pinedo M, Chao X (1999) Operations scheduling with applications in manufacturing and services. McGraw-Hill, London
22. Rubio R, Wein LM (1996) Setting base stock levels using product-form queueing networks. Management Science 42: 259–268
23. Shantikumar JG, Buzacott JA (1981) Open queueing network models of dynamic job shops. International Journal of Production Research 19: 255–266
24. Silver EA, Pyke DF, Peterson R (1998) Inventory management and production planning and scheduling. Wiley, New York
25. Sox CR, Jackson PL, Bowman A, Muckstadt JA (1999) A review of the stochastic lot scheduling problem. International Journal of Production Economics 62: 181–200
26. Suri R, Sanders JL, Kamath M (1993) Performance evaluation of production networks. In: Graves SC et al (eds) Handbooks in Operations Research and Management Science, vol 4, pp 199–286. Elsevier, Amsterdam
27. Vandaele NJ (1996) The impact of lot sizing on queueing delays: multi product, multi machine models. Ph.D. thesis, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium

28. Vandaele NJ, Lambrecht MR, De Schuyter N, Cremmery R (2000) Spicer Off-Highway Products Division-Brugge improves its lead-time and scheduling performance. Interfaces 30: 83–95
29. Van Nyen PLM, Van Ooijen HPG, Bertrand JWM (2004) Simulation results on the performance of Albin and Whitt's estimation method for waiting times in integrated production-inventory systems. International Journal of Production Economics 90: 237–249
30. Van Nyen PLM, Bertrand JWM, Van Ooijen HPG, Vandaele NJ (2004) The control of an integrated multi-product multi-machine production-inventory system. Working paper, BETA Research School, The Netherlands
31. Van Nyen PLM (2005) The integrated control of production inventory systems. Ph.D. thesis, Department of Technology Management, Technische Universiteit Eindhoven, The Netherlands (to appear)
32. Whitt W (1983) The queueing network analyzer. Bell System Technical Journal 62: 2779–2815
33. Whitt W (1994) Towards better multi-class parametric-decomposition approximations for open queueing networks. Annals of Operations Research 48: 221–224
34. Zipkin P (1986) Models for design and control of stochastic multi-item batch production systems. Operations Research 34: 91–104

Performance analysis of parallel identical machines with a generalized shortest queue arrival mechanism

G.J. van Houtum[1], I.J.B.F. Adan[2], J. Wessels[2], and W.H.M. Zijm[1,3]

[1] Faculty of Technology Management, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands (e-mail: G.J.v.Houtum@tm.tue.nl)
[2] Faculty of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands
[3] Faculty of Applied Mathematics, University of Twente, Twente, The Netherlands

Received: February 8, 2000 / Accepted: November 28, 2000

Abstract. In this paper we study a production system consisting of a group of parallel machines producing multiple job types. Each machine has its own queue and it can process a restricted set of job types only. On arrival a job joins the shortest queue among all queues capable of serving that job. Under the assumption of Poisson arrivals and identical exponential processing times we derive upper and lower bounds for the mean waiting time. These bounds are obtained from so-called flexible bound models, and they provide a powerful tool to efficiently determine the mean waiting time. The bounds are used to study how the mean waiting time depends on the amount of overlap (i.e. common job types) between the machines.

Key words: Queueing system – Shortest queue routing – Performance analysis – Flexibility – Truncation model – Bounds

1 Introduction

In this paper we consider a queueing system consisting of a group of parallel identical servers serving multiple job types. Each server has its own queue and is capable of serving a restricted set of job types only. Jobs arrive according to a Poisson process and on arrival they join the shortest feasible queue. The service times are exponentially distributed. We will refer to this queueing model as the *Generalized Shortest Queue System* (GSQS). This model is motivated by a situation encountered in the assembly of Printed Circuit Boards (PCBs). This is explained in more detail below.

Figure 1 shows a typical layout of an assembly system for PCBs. It consists of three parallel *insertion machines*, each with its own local buffer. An insertion

Correspondence to: G.J. van Houtum

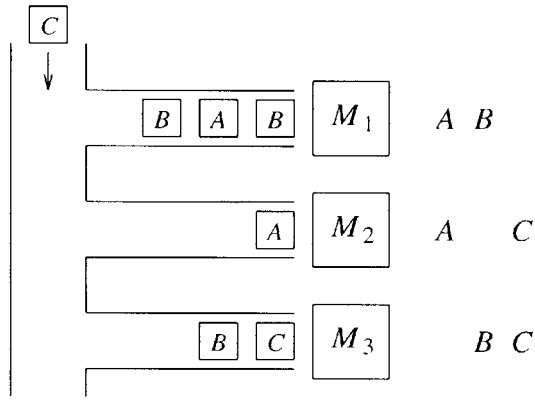


Fig. 1. A flexible assembly system consisting of three parallel insertion machines, on which three types of PCBs are made

machine mounts vertical components, such as resistors and capacitators, on a PCB by the *insertion head*. The components are mounted in a certain sequence, which is prescribed by a Numerical Control program. The insertion head is fed by the *sequencer*, which picks components from tapes and transports them in the right order to the insertion head. Each tape contains only *one* type of components. The tapes are stored in the *component magazine*, which can contain at most 80 tapes, say. Each PCB needs on average 60 different types of components. To assemble a PCB all required components have to be available in the component magazine. Hence, the set of components available in the magazine determines the set of PCB types that can be processed on that machine. The system in Figure 1 has to assemble three PCB types, labeled A, B and C. The machines are basically similar, but due to the fact that they are loaded with different types of components, the sets of PCB types that can be handled by the machines are different. Machine M_1 can handle the A and B types, machine M_2 the A and C, and machine M_3 the B and C. When the mounting times for all PCB types are approximately the same, it is reasonable to send arriving PCBs to the shortest feasible queue.

Since the assembly of PCBs is often characterized by relatively few job types, large production batches and small mounting times (see Zijm [16]), the use of a queueing model seems appropriate to predict performance characteristics such as the mean waiting time. An important issue is the assignment of the required components to the machines. Ideally, each machine should get all components needed to process all PCB types. However, since the component magazines have a finite capacity, they can contain the components needed for a (small) subset of PCB types only. In this paper we will investigate how much overlap (i.e. common components) between the machines is required such that the system nearly performs as in the ideal situation where the machines are equipped with all components.

The GSQS is also relevant for many other practical situations; e.g., for parallel machines loaded with different sets of tools, computer disks loaded with different

information files, or operators in a call center handling requests from different customers. Nevertheless, the literature on the GSQS is limited. Schwartz [12] (see also Roque [11]) considered a system related to the GSQS, but with a specific server hierarchy. He derived some expressions for the mean waiting times. Adan, Wessels and Zijm [2] derived rough approximations for the mean waiting times in a GSQS. Green [7] constructed a truncation model for a related system with two types of jobs and two types of servers: servers which can serve both job types and servers which can only serve jobs of the second type.

For the present model with general (i.e. nonexponential) arrivals, Sparaggis, Cassandras and Towsley [13] showed that the generalized shortest queue routing is optimal with respect to the overall mean waiting time for symmetric cases (see Theorem 3.1 in [13]; see also Subsection 2.3). For more general systems, Foss and Chernova [6] used a fluid approximation approach to establish ergodicity conditions (see also the remarks at the end of Section 2.2). The issue of ergodicity has also been considered in a recent report by Foley and McDonald [5]. Their main contribution, however, consists of results on the asymptotic behavior of a GSQS with two exponential servers with different service rates. Finally, Hassin and Haviv [8] have studied a symmetric GSQS with two servers and an additional property called threshold jockeying. They focus on the difference in waiting time between jobs which can choose between both servers and jobs which can not choose.

The GSQS can be described by a continuous-time Markov process with multi-dimensional states where each component denotes the queue length at one of the servers. Only in very special cases exact analytical solutions can be found (see e.g. [3]). Therefore, to determine the mean waiting times, we will construct truncation models which: (i) are flexible (i.e. the size of their state space can be controlled by one or more truncation parameters); (ii) can be solved efficiently; (iii) provide upper and lower bounds for the mean waiting times. Such models are called solvable flexible bound models. They are derived by using the so-called *precedence relation method*. This is a systematic approach for the construction of bound models, which has been developed in [14, 15]. In this paper we will construct a lower and upper bound model for the mean waiting times. These two models constitute the core of a powerful numerical approach: the two bound models are solved for increasing sizes of the truncated state space until the mean waiting times are determined *within a given, desired accuracy*.

This paper is organized as follows. In Section 2, we describe the GSQS and we discuss conditions under which the GSQS is ergodic and balanced. Next, in Section 3, we construct the flexible bound models and we formulate a numerical approach to determine the mean waiting times. Finally, in Section 4, we investigate how the mean waiting times for the GSQS depend on the amount of overlap (i.e. common job types) between the servers. This is done by numerically evaluating several scenarios.

2 Model

This section consists of three subsections. In the first subsection, we describe the GSQS. In Subsection 2.2 we present a simple condition that is necessary and suf-

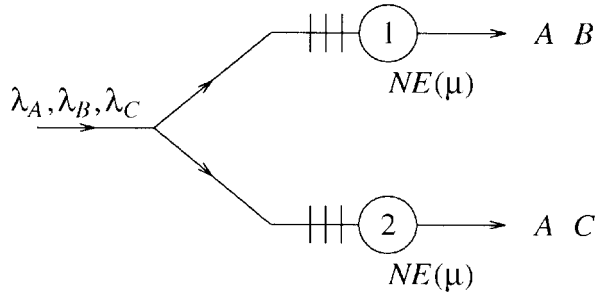


Fig. 2. A GSQS with $c = 2$ servers and three job types

ficient for ergodicity. In the last subsection, we present a related condition under which the GSQS is said to be *balanced* and we briefly discuss *symmetric* systems.

2.1 Model description

The GSQS consists of $c \geq 2$ parallel servers serving multiple job types. Each server has its own queue and is capable of serving a restricted set of job types only. All service times are exponentially distributed with the same parameter $\mu > 0$. The arrival stream of each job type is Poisson and an arriving job joins the shortest queue among all queues capable of serving that job (ties are broken with equal probabilities). Figure 2 shows a GSQS with $c = 2$ servers and three job types: type A, B and C jobs arrive with intensity λ_A, λ_B and λ_C, respectively. The A jobs can be served by both servers, the B jobs can only be served by server 1, and the C jobs must be served by server 2.

We introduce the following notations. The servers are numbered from $1, \ldots, c$ and the set I is defined by $I = \{1, \ldots, c\}$. The set of all job types is denoted by J. The arrival intensity of type $j \in J$ jobs is given by $\lambda_j \geq 0$, and $\lambda = \sum_{j \in J} \lambda_j$ is the total arrival intensity. For each $j \in J$, $I(j)$ denotes the set of servers that can serve the jobs of type j. We assume that each job type can be served by at least one server and each server can handle at least one job type; so, $I(j) \neq \emptyset$ for all $j \in J$, and $\cup_{j \in J} I(j) = I$. Without loss of generality, we set $\mu = 1$. Then the average workload per server is given by $\rho = \lambda/c$. Obviously, the requirement $\rho < 1$ is necessary for ergodicity.

The behavior of the GSQS is described by a continuous-time Markov process with states $(m_1, \ldots, m_c)$, where m_i denotes the length of the queue at server i, $i \in I$ (jobs in service are included). So, the state space is equal to

$$M = \{m \mid m = (m_1, \ldots, m_c) \text{ with } m_i \in \boldsymbol{N}_0 \text{ for all } i \in I\} . \tag{1}$$

We assume that $\sum_{j \in J} \lambda_j 1_{\{i \in I(j)\}} > 0$ for all servers $i \in I$ (here, $1_{\{G\}}$ is the indicator function, which is 1 if G is true and 0 otherwise), i.e., that all servers have a positive *potential* arrival rate. This guarantees that the Markov process is

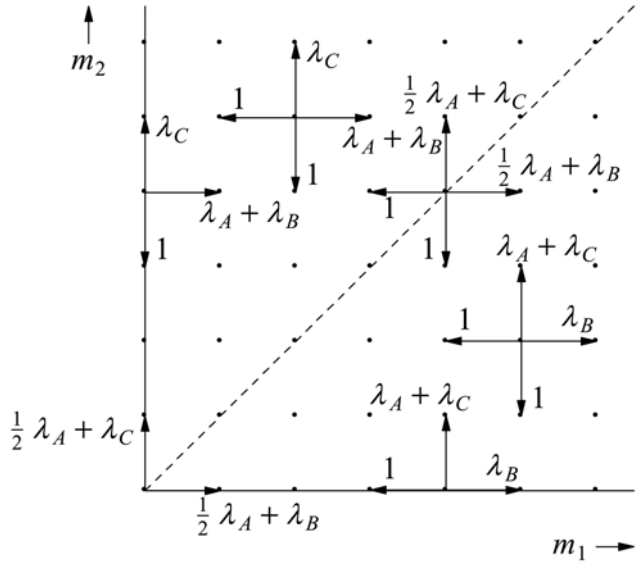


Fig. 3. The transition rate diagram for the GSQS in Figure 2

irreducible. The transition rates are denoted by $q_{m,n}$. Figure 3 shows the transition rates for the GSQS in Figure 2.

The relevant performance measures are the mean waiting times $W^{(j)}$ for each of job type $j \in J$, and the overall mean waiting time W, which is equal to

$$W \;=\; \sum_{j \in J} \frac{\lambda_j}{\lambda}\, W^{(j)}\,. \tag{2}$$

It is obvious that for an ergodic system,

$$W^{(j)} \;=\; \sum_{(m_1,\ldots,m_c)\in M} \left(\min_{i \in I(j)} m_i \right) \pi(m_1, \ldots, m_c)\,, \qquad j \in J, \tag{3}$$

where $\pi(m_1, \ldots, m_c)$ denotes the steady-state probability for state $(m_1, \ldots, m_c)$.

2.2 Ergodicity

By studying the job routing, we obtain a simple, *necessary* condition for the ergodicity of the GSQS. For each subset $J' \subset J$, $J' \neq \emptyset$, jobs of type $j \in J'$ arrive with an intensity equal to $\sum_{j \in J'} \lambda_j$ and they must be served by the servers $\cup_{j \in J'} I(j)$. This immediately leads to the following lemma.

Lemma 1 *The GSQS can only be ergodic if*

$$\sum_{j \in J'} \lambda_j \;<\; |\cup_{j \in J'} I(j)| \qquad \textit{for all } J' \subset J,\; J' \neq \emptyset. \tag{4}$$

Note that for $J' = J$, this inequality is equivalent to $\rho < 1$. For the GSQS in Figure 2, condition (4) states that for ergodicity it is necessary that the inequalities $\lambda_B < 1$, $\lambda_C < 1$ and $\lambda < 2$ (or, equivalently, $\rho < 1$) are satisfied. It appears that condition (4) is also sufficient for ergodicity. To show this, we consider so-called corresponding static systems.

A *corresponding static system* is a system that is identical to the GSQS, but with *static (random) routing* instead of dynamic shortest queue routing. The static routing is described by discrete distributions $\{x_i^{(j)}\}_{i \in I(j)}$, $j \in J$, where for each $j \in J$ and $i \in I(j)$, the variable $x_i^{(j)}$ denotes the probability that an arriving job of type j is sent to server i. Under static routing, it holds for each $j \in J$ that the Poisson stream of arriving type j jobs is split up into Poisson streams with intensities $x_{j,i} = \lambda_j x_i^{(j)}$, $i \in I(j)$, for type j arrivals joining server i. Hence the queues $i \in I$ constitute independent $M/M/1$ queues with identical mean service times equal to $\mu = 1$ and arrival intensities $\sum_{j \in J(i)} x_{j,i}$, where $J(i) = \{j \in J | i \in I(j)\}$. As a result, we obtain a simple necessary and sufficient condition for the ergodicity of a corresponding static system, viz.

$$\sum_{j \in J(i)} x_{j,i} \; < \; 1 \qquad \text{for all } i \in I.$$

Lemma 2 *For a GSQS, there exists a corresponding static system that is ergodic, if and only if condition (4) is satisfied.*

Proof. There exists a corresponding static system that is ergodic if and only if there exists a nonnegative solution $\{x_{j,i}\}_{(j,i) \in A}$, with $A = \{(j,i) \mid j \in J,\ i \in I \text{ and } i \in I(j)\}$, of the following equations and inequalities:

$$\sum_{i \in I(j)} x_{j,i} \; = \; \lambda_j \;\; \text{for all } j \in J, \qquad \sum_{j \in J(i)} x_{j,i} \; < \; 1 \;\; \text{for all } i \in I; \tag{5}$$

the equalities in (5) guarantee that the solution $\{x_{j,i}\}_{(j,i) \in A}$ corresponds to discrete distributions $\{x_i^{(j)}\}_{i \in I(j)}$ which describe a static routing, and the inequalities in (5) must be satisfied for ergodicity. It is easily seen that (5) has no solution if condition (4) is not satisfied.

Now, assume that condition (4) is satisfied. To prove that there exists a nonnegative solution $\{x_{j,i}\}_{(j,i) \in A}$ of (5), we consider a *transportation problem* with supply nodes $\hat{V}_1 = J \cup \{0\}$, demand nodes $\hat{V}_2 = I$, and arcs $\hat{A} = A \cup \{(0,i) | i \in I\}$ (supply node 0 denotes an extra type of jobs, which can be served by all servers). Define the supplies $\hat{a}_j$ by $\hat{a}_j = \lambda_j$ for all $j \in \hat{V}_1 \setminus \{0\}$ and $\hat{a}_0 = c - \lambda - c\epsilon$, where

$$\epsilon \; := \; \min_{\substack{J' \subset J \\ J' \neq \emptyset}} \frac{|\cup_{j \in J'} I(j)| \; - \; \sum_{j \in J'} \lambda_j}{|\cup_{j \in J'} I(j)|}$$

(from (4), it follows that $\epsilon > 0$, and $\hat{a}_0 \geq 0$ since by taking $J' = J$ we obtain the inequality $\epsilon \leq (c - \lambda)/c$). Further, we define the demands $\hat{b}_i$ by $\hat{b}_i = 1 - \epsilon$ for all $i \in \hat{V}_2$; note that $\sum_{j \in \hat{V}_1} \hat{a}_j = \sum_{i \in \hat{V}_2} \hat{b}_i$. It may be verified that this transportation problem satisfies a necessary and sufficient condition for the existence of a feasible flow; see Lemma 5.4 of [14] and its proof is based on a transformation

to a maximum-flow problem followed by the application of the max-flow min-cut theorem (see e.g. [4]). So, there exists a feasible flow for the transportation problem, i.e., there exists a nonnegative solution $\{\hat{x}_{j,i}\}_{(j,i)\in\hat{A}}$ of the equations

$$\sum_{\substack{i\in\hat{V}_2\\(j,i)\in\hat{A}}} \hat{x}_{j,i} \;=\; \hat{a}_j \;\text{ for all } j\in\hat{V}_1, \qquad \sum_{\substack{j\in\hat{V}_1\\(j,i)\in\hat{A}}} \hat{x}_{j,i} \;=\; \hat{b}_i \;\text{ for all } i\in\hat{V}_2.$$

It is easily seen that then the solution $\{x_{j,i}\}_{(j,i)\in A}$ defined by $x_{j,i} = \hat{x}_{j,i}$ for all $(j,i)\in A$, is a nonnegative solution of (5), which completes the proof. □

In situations with many job types shortest queue routing will balance the queue lengths more than any static routing. So if there is a corresponding static system that is ergodic, then the GSQS will also be ergodic. Together with Lemma 2, this informally shows that the following theorem holds.

Theorem 1 *The GSQS is ergodic if and only if condition (4) is satisfied.*

For a formal proof of this theorem, the reader is referred to Foss and Chernova [6] or Foley and McDonald [5]. In the latter paper, a generalization of condition (4) is proved to be necessary and sufficient for the (more general) model with different service rates. Their proof also exploits the connection with a corresponding static system. Foss and Chernova [6] use a fluid approximation approach to derive necessary conditions for a model with general arrivals and general service times.

2.3 Balanced and symmetric systems

It is desirable that the shortest queue routing, as reflected by the sets $I(j)$, balances the workload among the servers. Formally, we say that a GSQS is *balanced* if there exists a corresponding static system for which all queues have the same workload. This means that there must exist discrete distributions $\{x_i^{(j)}\}_{i\in I(j)}$ such that for each server $i\in I$, the arrival intensity $\sum_{j\in J,\ (j,i)\in A} x_{j,i}$ is equal to $\lambda/c = \rho$, where the $x_{j,i}$ and the set A are defined as before. Such discrete distributions exist if and only if there exists a nonnegative solution $\{x_{j,i}\}_{(j,i)\in A}$ of the equations

$$\sum_{\substack{i\in I\\(j,i)\in A}} x_{j,i} \;=\; \lambda_j \;\text{ for all } j\in J, \qquad \sum_{\substack{j\in J\\(j,i)\in A}} x_{j,i} \;=\; \frac{\lambda}{c} \;\text{ for all } i\in I. \tag{6}$$

These equations are precisely the equations which must be satisfied by a feasible flow for the transportation problem with supply nodes $V_1 = J$, demand nodes $V_2 = I$, arcs A, supplies $a_j = \lambda_j$ for all $j\in V_1$ and demands $b_i = \lambda/c$ for all $i\in V_2$. Applying the necessary and sufficient condition for the existence of such a feasible flow (see [14]) leads to the following lemma.

Lemma 3 *A GSQS is balanced if and only if*

$$\sum_{j\in J'} \lambda_j \;\le\; |\cup_{j\in J'} I(j)|\,\frac{\lambda}{c} \qquad \textit{for all } J'\subset J. \tag{7}$$

Note that for $J' = \emptyset$ and $J' = J$, condition (7) holds by definition. Further, it follows that a balanced GSQS satisfies condition (4) if and only if $\rho < 1$. So, for a balanced GSQS, the simple condition $\rho < 1$ is necessary and sufficient for ergodicity.

For a balanced GSQS the workloads under the shortest queue routing are not necessarily balanced. This can be seen by considering the GSQS in Figure 2. According to condition (7), this GSQS is balanced if and only if $\lambda_B \leq \lambda/2$ and $\lambda_C \leq \lambda/2$, i.e. if and only if $\lambda_B \leq \lambda_A + \lambda_C$ and $\lambda_C \leq \lambda_A + \lambda_B$. This condition is obviously satisfied if we take $\lambda_C = \lambda_A + \lambda_B$. In this case, equal workloads for both servers can only be obtained if all jobs of type A are sent to server 1. But, under the shortest queue routing, it will still occur that jobs of type A are sent to server 2, and therefore server 2 will have a higher workload than server 1. Nevertheless, one may expect that for a balanced GSQS, the shortest queue routing at least ensures that the workloads will not differ too much.

A subclass of balanced systems are the symmetric systems. A GSQS is said to be *symmetric*, if

$$\lambda(I_1) = \lambda(I_2) \qquad \text{for all } I_1, I_2 \subset I \text{ with } |I_1| = |I_2|, \tag{8}$$

where

$$\lambda(I') := \sum_{\substack{j \in J \\ I(j) = I'}} \lambda_j \,, \qquad I' \subset I.$$

So, a GSQS is symmetric, if for all subsets $I' \subset I$ with the same number of servers $|I'|$, the arrival intensity $\lambda(I')$ for the jobs which can be served by precisely the servers of I', is the same. The GSQS in Figure 2 is symmetric if $\lambda_B = \lambda_C$.

For a symmetric GSQS, all queue lengths have the same distribution, which implies that all servers have equal workloads. For such a system, it follows from Sparaggis et al. [13], that the shortest queue routing minimizes the total number of jobs in the system and hence the overall mean waiting time W. In particular, this implies that the overall mean waiting time in a symmetric GSQS is less than in the corresponding system consisting of N independent $M/M/1$ queues with workload ρ.

3 Flexible bound models

In this section we construct two truncation models which are much easier to solve than the original model. One truncation model produces lower bounds for the mean waiting times, and the other one upper bounds. At the end of this section we describe a numerical method for the computation of the mean waiting times within a given, desired accuracy.

The truncation models exploit the property that the shortest queue routing causes a drift towards states with equal queue lengths. The state space M' of the two models is obtained by truncating the original state space M around the diagonal, i.e.,

$$M' = \{m \in M \mid m = (m_1, \ldots, m_c) \text{ and } m_i \leq \min(m) + T_i \text{ for all } i \in I\} \,, \tag{9}$$

where $\min(m) := \min_{i \in I} m_i$ and $T_1, \ldots, T_c \in \boldsymbol{N}$ are so-called threshold parameters; the corresponding vector $\hat{T} := (T_1, \ldots, T_c)$ is called the threshold vector. So state $m \in M$ also lies in M' if and only if for each $i \in I$ the length of queue i is at most T_i greater than the length of any other queue. Later on in this section we discuss how appropriate values for $\hat{T}$ can be selected. There are two types of transitions pointing from states inside M' to states outside M':

(i) in state $m = (m_1, \ldots, m_c) \in M'$ with $\min(m) > 0$ and $I' = \{i \in I | m_i = \min(m) + T_i\} \neq \emptyset$, at a server $k \in I$ with $m_k = \min(m)$ a service completion occurs with rate 1 and leads to a transition from m to state $n = m - e_k \notin M'$;

(ii) in state $m = (m_1, \ldots, m_c) \in M'$ with $I' = \{i \in I | m_i = \min(m) + T_i\} \neq \emptyset$, at a server $i \in I'$ an arrival of a new job leads to a transition from m to the state $n = m + e_i \notin M'$; this transition occurs with rate $\sum_{j \in J} |I(j;m)|^{-1} \lambda_j 1_{\{i \in I(j;m)\}}$, where the set $I(j;m)$ is defined by $I(j;m) = \{i \in I(j) \mid m_i = \min_{k \in I(j)} m_k\}$ (note that this rate may be equal to 0).

In the lower (upper) bound model, the transitions to states n outside M' are redirected to states n' with less (more) jobs inside M'.

In the lower bound model, the transition in (i) is redirected to $n' = m - e_k - \sum_{i \in I'} e_i \in M'$. This means that the departure of a job at a non-empty shortest queue is accompanied by killing one job at each of the queues $i \in I'$, which are already T_i greater than the shortest queue. The transition in (ii) is redirected to m itself, i.e., a new job arriving at one of the servers $i \in I'$ is rejected. The lower bound model is therefore called the *Threshold Killing and Rejection (TKR) model*.

In the upper bound model, the transition in (i) is redirected to m itself. This means that if at least one queue is already T_i greater than the shortest queue, the finished job in the shortest queue is not allowed to depart, but is served once more; this is equivalent to saying that the servers at the shortest queues are blocked. Transition (ii) is redirected to $n' = m + e_i + \sum_{k \in I_{sq}} e_k \in M'$, with $I_{sq} = \{k \in I | m_k = \min(m)\}$. This means that an arrival of a new job at one of the queues which is already T_i greater than the shortest queue, is accompanied by the addition of one extra job at each of the shortest queues. The upper bound model is therefore called the *Threshold Blocking and Addition (TBA) model*. Note that this model may be non-ergodic while the original model is ergodic. However, the larger the values of the thresholds T_i the more unlikely this situation. In Figure 4, we show the redirected transitions in the lower and upper bound model for the GSQS of Figure 3.

It is intuitively clear that the queues in the TKR model are stochastically smaller than the queues in the original model. Hence, for each $j \in J$, the TKR model yields a lower bound for the mean length of the shortest queue among the queues $i \in I(j)$, and thus also for the mean waiting time of type j jobs (cf. (3)). Denote the steady-state probabilities in the TKR model by $\pi_{TKR}(m_1, \ldots, m_c)$ and let

$$W^{(j)}_{TKR}(\hat{T}) = \sum_{(m_1,\ldots,m_c) \in M'} \left(\min_{i \in I(j)} m_i \right) \pi_{TKR}(m_1, \ldots, m_c) \,, \qquad j \in J.$$

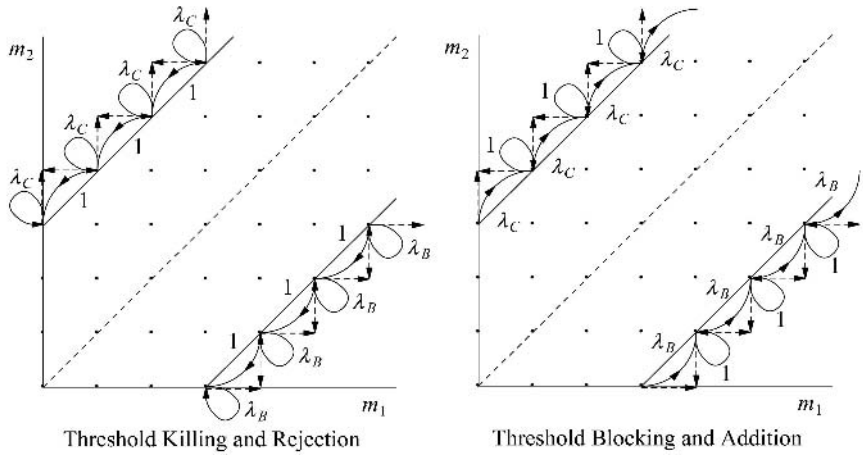


Fig. 4. The redirected transitions in the TKR and TBA model for the GSQS depicted in Figure 2. For both models, $\hat{T} = (T_1, T_2) = (3, 3)$

Then we have for each $j \in J$ that $W^{(j)}_{TKR}(\hat{T}) \leq W^{(j)}$, and thus (cf. (2))

$$W_{TKR}(\hat{T}) = \sum_{j \in J} \frac{\lambda_j}{\lambda} W^{(j)}_{TKR}(\hat{T})$$

yields a lower bound for the overall mean waiting time W. The lower bounds $W^{(j)}_{TKR}(\hat{T})$ monotonically increase as the thresholds $T_1, \ldots, T_c$ increase. Similarly the TBA model produces monotonically decreasing upper bounds $W^{(j)}_{TBA}(\hat{T})$, $j \in J$, and $W_{TBA}(\hat{T})$. The bounds and the monotonicity properties can be rigorously proved by using the *precedence relation method*, see [14]. This method is based on Markov reward theory and it has been developed in [14, 15].

The truncation models can be solved efficiently by using the *matrix-geometric approach* described in [10]. Since the truncation models exploit the property that shortest queue routing tries to balance the queues, one may expect that the bounds are tight for already moderate values of the thresholds $T_1, \ldots, T_c$.

We will now formulate a numerical method to determine the mean waiting times with an absolute accuracy ϵ_{abs}. The method repeatedly solves the TKR and TBA model for increasing threshold vectors $\hat{T} = (T_1, \ldots, T_c)$. For each vector $\hat{T}$ we use $(W^{(j)}_{TKR}(\hat{T}) + W^{(j)}_{TBA}(\hat{T}))/2$ as an approximation for $W^{(j)}$ and $\Delta^{(j)}(\hat{T}) = (W^{(j)}_{TBA}(\hat{T}) - W^{(j)}_{TKR}(\hat{T}))/2$ as an upper bound for the error; we similarly approximate W by $(W_{TKR}(\hat{T}) + W_{TBA}(\hat{T}))/2$ where the error is at most $\Delta(\hat{T}) = (W_{TBA}(\hat{T}) - W_{TKR}(\hat{T}))/2$. The approximations and error bounds are set equal to ∞ if the TBA model is not ergodic (which may be the case for small thresholds). The computation procedure stops when all error bounds are less than or equal to ϵ_{abs}; otherwise at least one of the thresholds is increased by 1 and new approximations are computed. The decision to increase a threshold T_i is based on the rate of redirections $r_{rd}(i)$. This is explained in the next paragraph.

The variable $r_{rd}(i)$, $i \in I$, denotes the rate at which redirections occur in the boundary states $m = (m_1, \ldots, m_c)$ with $m_i = \min(m) + T_i$ of the truncated state space. If for given $\hat{T}$ only the TKR model is ergodic, then $r_{rd}(i)$ denotes the rate for the TKR model, otherwise $r_{rd}(i)$ denotes the sum of the rate for the TKR and TBA model. The rates $r_{rd}(i)$ can be computed directly from the steady-state distributions of the bound models. The higher the rate $r_{rd}(i)$, the higher the expected impact of increasing T_i. The computation procedure increases all thresholds T_i for which $r_{rd}(i) = \max_{k \in I} r_{rd}(k)$. The numerical method is summarized below.

Algorithm *(to determine the mean waiting times for the GSQS)*

Input:	The data of an ergodic instance of the GSQS, i.e., c, J, $I(j)$ for all $j \in J$, and λ_j for all $j \in J$; the absolute accuracy ϵ_{abs}; the initial threshold vector $\hat{T} = (T_1, \ldots, T_c)$.
Step 1.	Determine $W^{(j)}_{TKR}(\hat{T})$, $W^{(j)}_{TBA}(\hat{T})$ and $\Delta^{(j)}(\hat{T})$ for all $j \in J$, and $W_{TKR}(\hat{T})$, $W_{TBA}(\hat{T})$ and $\Delta(\hat{T})$, and $r_{rd}(i)$ for all $i \in I$.
Step 2.	If $\Delta^{(j)}(\hat{T}) > \epsilon_{abs}$ for some $j \in J$ or $\Delta(\hat{T}) > \epsilon_{abs}$, then $T_i := T_i + 1$ for all $i \in I$ with $r_{rd}(i) = \max_{k \in I} r_{rd}(k)$, and return to Step 1.
Step 3.	$W^{(j)} = (W^{(j)}_{TKR}(\hat{T}) + W^{(j)}_{TBA}(\hat{T}))/2$ for all $j \in J$, and $W = (W_{TKR}(\hat{T}) + W_{TBA}(\hat{T}))/2$.

Note that for a symmetric GSQS it is natural to start with a threshold vector $\hat{T}$ with equal components. Then in each iteration all rates $r_{rd}(i)$ will be equal, and hence each T_i will be increased by 1. So the components of $\hat{T}$ will remain equal.

4 Numerical study of the GSQS

In this section we consider three scenarios. In Subsection 4.1 we distinguish two types of jobs: *common jobs* and *specialist jobs*. The common jobs can be served by all servers and the other ones can be served by only one specific server. We focus on the behavior of the overall mean waiting time W as a function of the fraction of work due to common jobs. The higher this fraction, the more balanced the queues and the better the performance. So W will be decreasing as the number of common jobs increases. In one extreme case, viz. when all jobs are specialist jobs, the GSQS reduces to independent $M/M/1$ queues, and W is maximal. In the other extreme case, viz. when all jobs are common jobs, the GSQS is identical to a pure Symmetric Shortest Queue System (SSQS), and W is minimal. In Subsection 4.1 we investigate how W behaves in between these two extremes.

In Subsection 4.2 we consider a symmetric GSQS with $c = 3$ servers, and, besides common and specialist jobs, we also have *semi-common jobs*. These jobs can be served by two servers. We compare two situations: (i) a GSQS with a given fraction of common jobs (and no semi-common jobs); (ii) a GSQS with twice this

fraction of semi-common jobs (and no common jobs). In both cases the average number of servers capable of serving an arbitrary job is the same. In Subsection 4.3 we evaluate a series of balanced, asymmetric systems. We investigate how the mean waiting times deteriorate due to the asymmetry. Finally, in Subsection 4.4, the main conclusions are summarized.

4.1 The impact of common jobs

We distinguish $c+1$ job types, numbered $1,\ldots,c,c+1$. Type j jobs are specialist jobs, which can only be served by server j, $j=1,\ldots,c$. The type $c+1$ jobs are common jobs, which can be served by all servers. The total arrival intensity is equal to $\lambda = c\rho$, with $\rho \in (0,1)$. The common jobs constitute a fraction p, $p \in [0,1]$, of the total arrival stream, while each of the streams of specialist jobs constitutes an equal part of the remaining stream. So $\lambda_{c+1} = p\lambda$ and $\lambda_j = (1-p)\lambda/c$ for $j=1,\ldots,c$.

Table 1 lists the mean waiting times for specialist jobs ($= W^{(1)} = \ldots = W^{(c)}$), common jobs ($= W^{(c+1)}$), and an arbitrary job ($= W$) as a function of p for a system with $c=2$ and $c=3$ servers, respectively, and a workload $p=0.9$. For $p=0$ there are no common jobs; then $W^{(c+1)}$ is defined as the limiting value of the waiting time of common jobs as $p \downarrow 0$. For $p=1$ a similar remark holds for the mean waiting times $W^{(1)} = \cdots = W^{(c)}$. Table 1 also lists the *realized reduction* $rr(p)$. This is defined as

$$rr(p) \;=\; \frac{W_{M/M/1} - W}{W_{M/M/1} - W_{SSQS}}\,, \tag{10}$$

where $W_{M/M/1}$ and W_{SSQS} denote the mean waiting time in an $M/M/1$ system and SSQS, respectively, both with the same workload $\rho = 0.9$ and mean service time $\mu = 1$ as for the GSQS. The mean waiting time $W_{M/M/1}$ is realized when $p=0$, and W_{SSQS} is realized when $p=1$. Clearly, $rr(0)=0$ and $rr(1)=1$ by definition. For all cases in Table 1, $W_{M/M/1}=9$ and $W_{SSQS}=4.475$ for $c=2$ and $W_{SSQS}=2.982$ for $c=3$. The mean waiting times in the SSQS have been determined with an absolute accuracy of 0.0001 by using the bound models in [1]. The mean waiting times in Table 1 have been determined by using the algorithm described in Section 3 with an absolute accuracy $\epsilon_{abs}=0.005$.

In Table 1 we see that the overall mean waiting time $W = pW^{(c+1)} + (1-p)W^{(1)}$ sharply decreases for small values of p; see also Figure 5. Already 73% of the maximal reduction is realized when 20% of the jobs is common and 91% of the maximal reduction is realized when 50% of the jobs is common. A surprising result is that the realized reduction $rr(p)$ is almost the same for $c=2$ and $c=3$ servers. Further note that for large p the mean waiting time $W^{(1)}$ for specialist jobs is only a little bit larger than the mean waiting time $W^{(c+1)}$ for common jobs. This is due to the balancing effect of the common jobs.

The behavior of the overall mean waiting time W is further investigated in Table 2 for different values of p, ρ and c. The mean waiting times are again determined with an absolute accuracy $\epsilon_{abs} = 0.005$ (and 0.0001 for W_{SSQS}). Only for low workloads (i.e., $\rho \leq 0.4$), the mean waiting time has been determined even more

Table 1. Mean waiting times as a function of p and c

	$c=2$				$c=3$			
p	$W^{(1)}$	$W^{(c+1)}$	W	$rr(p)$	$W^{(1)}$	$W^{(c+1)}$	W	$rr(p)$
0.0	9.00	4.26	9.00	0.0 %	9.00	2.69	9.00	0.0 %
0.1	6.80	4.36	6.56	54.0 %	6.07	2.82	5.75	54.1 %
0.2	6.04	4.40	5.72	72.6 %	5.06	2.88	4.63	72.7 %
0.3	5.66	4.43	5.29	82.0 %	4.56	2.91	4.06	82.0 %
0.4	5.43	4.44	5.04	87.6 %	4.25	2.93	3.72	87.7 %
0.5	5.28	4.45	4.86	91.4 %	4.05	2.95	3.50	91.4 %
0.6	5.17	4.46	4.74	94.1 %	3.90	2.96	3.34	94.1 %
0.7	5.09	4.46	4.65	96.1 %	3.79	2.97	3.21	96.1 %
0.8	5.02	4.47	4.58	97.7 %	3.71	2.97	3.12	97.7 %
0.9	4.97	4.47	4.52	99.0 %	3.64	2.98	3.04	99.0 %
1.0	4.93	4.48	4.48	100.0 %	3.58	2.98	2.98	100.0 %

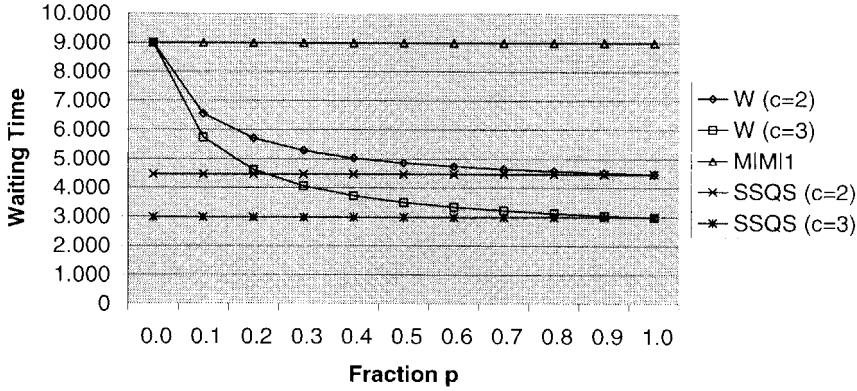


Fig. 5. Graphical representation of the mean waiting times W listed in Table 1

accurately in order to obtain sufficiently accurate estimates for $rr(p)$. The results in Table 2 show that for each combination of p and c, the values for W for varying workloads ρ are not that far away from the values for W_{SSQS}; in particular, the absolute differences are small for small workloads ρ and the relative differences are small for high workloads ρ. The results also suggest that the $rr(p)$ is insensitive to the number of servers c. However, $rr(p)$ strongly depends on ρ; it is rather small for low workloads and large for high workloads (it seems that $rr(p) \uparrow 1$ as $\rho \uparrow 1$).

4.2 Common versus semi-common jobs

In Subsection 4.1 we distinguished two job types only, specialist and common jobs. For GSQSs with more than two servers, one may also have jobs in between,

Table 2. Mean waiting times as a function of p, ρ and c

			$c=2$			$c=3$		
p	ρ	$W_{M/M/1}$	W	W_{SSQS}	$rr(p)$	W	W_{SSQS}	$rr(p)$
0.25	0.2	0.25	0.19	0.07	32.1 %	0.18	0.02	30.8 %
	0.4	0.67	0.51	0.26	39.6 %	0.46	0.13	38.6 %
	0.6	1.50	1.10	0.68	49.2 %	0.97	0.42	48.9 %
	0.8	4.00	2.67	1.96	64.8 %	2.24	1.29	64.8 %
	0.9	9.00	5.47	4.47	77.9 %	4.31	2.98	78.0 %
	0.95	19.00	10.69	9.49	87.3 %	7.94	6.33	87.4 %
	0.98	49.00	25.86	24.49	94.4 %	18.17	16.35	94.4 %
0.50	0.2	0.25	0.14	0.07	58.7 %	0.12	0.02	57.2 %
	0.4	0.67	0.40	0.26	66.4 %	0.32	0.13	65.5 %
	0.6	1.50	0.89	0.68	74.5 %	0.70	0.42	74.3 %
	0.8	4.00	2.27	1.96	84.7 %	1.70	1.29	84.8 %
	0.9	9.00	4.86	4.47	91.4 %	3.50	2.98	91.4 %
	0.95	19.00	9.93	9.49	95.4 %	6.92	6.33	95.4 %
	0.98	49.00	24.97	24.49	98.1 %	16.98	16.35	98.1 %

i.e., jobs that can be served by two or more, but not all servers. In this subsection we investigate which job types lead to the largest reduction of W: common or semi-common jobs?

We consider a GSQS with $c = 3$ servers and a total arrival rate $\lambda = 3\rho$ with $\rho \in (0, 1)$. The following two cases are distinguished for the detailed arrival streams. For *case I*, we copy the situation in Subsection 4.1. In this case there are 4 job types. The type 4 jobs are common jobs; they arrive with intensity $\lambda_4 = p\lambda$ with $p \in [0, 0.5]$ (the reason why p may not exceed 0.5 follows below). Type j jobs, $j = 1, 2, 3$ are specialist jobs which only can be served by server j; they arrive with intensity $\lambda_j = (1 - p)\lambda/3$. So the mean number of servers capable of serving an arbitrary job is equal to $1 + 2p$. In *case II* we have 6 job types. The type j jobs, $j = 1, 2, 3$, are again specialist jobs which can only be served by server j. The type 4, 5 and 6 jobs are semi-common jobs; the type 4 jobs can be served by the servers 1 and 2, the type 5 jobs by 1 and 3, and the type 6 jobs by 2 and 3. To guarantee that the mean number of servers capable of serving an arbitrary job remains the same (i.e., equal to $1 + 2p$), the arrival intensity λ_j is set equal to $\lambda_j = 2p\lambda/3$ for $j = 4, 5, 6$ and $\lambda_j = (1 - 2p)\lambda/3$ for $j = 1, 2, 3$ (to avoid negative intensities, p must be less than or equal to 0.5).

Table 3 lists the overall mean waiting time W for different values of p and ρ. The results for case I are copied from Table 2. We can conclude that the absolute difference between the mean waiting time W in case I and II is rather small in each situation. This suggests that W is mainly determined by the mean number of servers capable of serving an arbitrary job; it does not matter whether this mean number is realized by common or by (twice as many) semi-common jobs. Nevertheless, the results in Table 3 also show that in each situation case II yields a smaller W than

Table 3. Mean waiting times as a function of p and ρ

		W		Diff. (I−II)	
p	ρ	Case I	Case II	Abs.	Rel.
0.25	0.2	0.18	0.14	0.04	23.4 %
	0.4	0.46	0.38	0.08	18.1 %
	0.6	0.97	0.83	0.15	14.9 %
	0.8	2.24	1.97	0.27	11.9 %
	0.9	4.31	3.92	0.38	8.9 %
	0.95	7.94	7.46	0.48	6.0 %
	0.98	18.17	17.62	0.55	3.0 %
0.50	0.2	0.12	0.05	0.07	55.1 %
	0.4	0.32	0.22	0.10	32.5 %
	0.6	0.70	0.56	0.14	20.1 %
	0.8	1.70	1.51	0.19	11.2 %
	0.9	3.50	3.27	0.22	6.4 %
	0.95	6.92	6.67	0.25	3.6 %
	0.98	16.98	16.72	0.26	1.5 %

case I. This may be explained as follows. Let us consider the situation with $p = 0.5$. In case I, $\lambda_1 = \lambda_2 = \lambda_3 = \lambda/6$ and $\lambda_4 = \lambda/2$. Hence, for each group of 6 arriving jobs, on average 4 jobs join the shortest queue, 1 job joins the shortest but one queue, and 1 job joins the longest queue. In case II, however, $\lambda_1 = \lambda_2 = \lambda_3 = 0$ and $\lambda_4 = \lambda_5 = \lambda_6 = \lambda/3$. Thus for each group of 6 arriving jobs, on average 4 jobs join the shortest queue and 2 jobs joins the shortest but one queue. So in case II the balancing of queues will be slightly stronger, and thus W will be slightly smaller.

4.3 Balanced asymmetric systems

In this subsection we study the GSQS with $c = 2$ servers and three job types as depicted in Figure 2. The parameters are chosen as follows: $\rho = 0.9$, $\lambda = 2\rho = 1.8$, $\lambda_A = \lambda/2 = 0.9$, $\lambda_B = \hat{p}\lambda/2 = 0.9\hat{p}$, $\lambda_C = (1-\hat{p})\lambda/2 = 0.9(1-\hat{p})$ where $\hat{p} \in [0, 0.5]$. So one half of the jobs are common (type A) jobs and the other half are specialist (type B and C) jobs. But the specialist jobs are not equally divided over the servers. The fraction $\hat{p}$ of specialist jobs which must be served by server 1 (i.e., the type B jobs) is less than or equal to the fraction $1-\hat{p}$ of specialist jobs which must be served by server 2 (i.e. the type C jobs). Only for $\hat{p} = 0.5$ we have a symmetric system. For all $\hat{p} \in [0, 0.5)$ we have an asymmetric, but balanced system; a static system with equal workloads for both servers is obtained when a fraction $1-\hat{p}$ of the type A jobs is sent to server 1 and a fraction $\hat{p}$ to server 2.

Table 4 shows the mean waiting times $W^{(A)}, W^{(B)}, W^{(C)}$ for each job type and the overall mean waiting time W for $\hat{p} = 0, 0.1, \ldots, 0.5$. These waiting times have again been computed with an absolute accuracy $\epsilon_{abs} = 0.005$. In the last column of

Table 4. Mean waiting times as a function of $\hat{p}$

$\hat{p}$	$W^{(A)}$	$W^{(B)}$	$W^{(C)}$	W	$rr(\hat{p})$
0.0	4.28	4.34	13.05	8.66	7.5 %
0.1	4.37	4.52	8.52	6.25	60.8 %
0.2	4.42	4.68	6.93	5.45	78.5 %
0.3	4.44	4.84	6.12	5.09	86.5 %
0.4	4.45	5.03	5.62	4.92	90.3 %
0.5	4.45	5.28	5.28	4.86	91.4 %

Table 4 we list the realized reduction $rr(\hat{p})$ defined by (10), where $W_{M/M/1} = 9$ and $W_{SSQS} = 4.475$ for $\rho = 0.9$. The results in Table 4 show that $W^{(A)}$ is fairly constant for all values of $\hat{p}$. As expected, $W^{(B)}$ decreases and $W^{(C)}$ increases as $\hat{p}$ decreases. A striking observation is that $W^{(C)}$ sharply increases for $\hat{p}$ close to 0; and thus also $W = (W^{(A)} + \hat{p}W^{(B)} + (1 - \hat{p})W^{(C)})/2$. For $\hat{p} = 0$ we have $\lambda_A = \lambda_C = 0.9$ and $\lambda_B = 0$, and the overall mean waiting time W is equal to 8.66. This is close to $W_{M/M/1} = 9$, which is realized when all type A jobs would be sent to server 1.

4.4 Conclusion

The main conclusion from the numerical experiments is that the overall mean waiting time may already be reduced significantly by creating a little bit of (semi-) common work. Furthermore, this reduction is mainly determined by the amount of overlap, i.e., the mean number of servers capable of handling an arbitrary job. Finally, the beneficial effect of (semi-)common jobs may vanish for highly asymmetric situations.

References

1. Adan IJBF, Van Houtum GJ, Van der Wal J (1994) Upper and lower bounds for the waiting time in the symmetric shortest queue system. Annals of Operations Research 48: 197–217
2. Adan IJBF, Wessels J, Zijm WHM (1989) Queueing analysis in a flexible assembly system with a job-dependent parallel structure. In: Operations Research Proceedings, pp 551–558. Springer, Berlin Heidelberg New York
3. Adan IJBF, Wessels J, Zijm, WHM (1990) Analysis of the symmetric shortest queue problem. Stochastic Models 6: 691–713
4. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms, and applications. Prentice-Hall, Englewood Cliffs, NJ
5. Foley RD, McDonald DR (2000) Join the shortest queue: Stability and exact asymptotics. The Annals of Applied Probability (to appear)
6. Foss S, Chernova N (1998) On the stability of a partially accessible multi-station queue with state-dependent routing. Queueing Systems 29: 55–73

7. Green L (1985) A queueing system with general-use and limited-use servers. Operations Research 33: 168–182
8. Hassin R, Haviv M (1994) Equilibrium strategies and the value of information in a two line queueing system with threshold jockeying. Stochastic Models 10: 415–435
9. Latouche G, Ramaswami V (1993) A logarithmic reduction algorithm for quasi-birth-death processes. Journal of Applied Probability 30: 650–674
10. Neuts MF (1981) Matrix-geometric solutions in stochastic models. Johns Hopkins University Press, Baltimore
11. Roque DR (1980) A note on "Queueing models with lane selection". Operations Research 28: 419–420
12. Schwartz BL (1974) Queueing models with lane selection: a new class of problems. Operations Research 22: 331–339
13. Sparaggis PD, Cassandras CG, Towsley D (1993) Optimal control of multiclass parallel service systems with and without state information. In Proceedings of the 32nd Conference on Decision and Control, San Antonio, pp 1686–1691
14. Van Houtum GJ (1995) New approaches for multi-dimensional queueing systems. Ph.D. Thesis, Eindhoven University of Technology, Eindhoven
15. Van Houtum GJ, Zijm WHM, Adan IJBF, Wessels J (1998) Bounds for performance characteristics: A systematic approach via cost structures. Stochastic Models 14: 205–224 (Special issue in honor of M.F. Neuts)
16. Zijm WHM (1991) Operational control of automated PCB assembly lines. In: Fandel G, Zaepfel G (eds) Modern production concepts: theory and applications, pp 146–164. Springer, Berlin Heidelberg New York

A review and comparison of hybrid and pull-type production control strategies

John Geraghty[1] **and Cathal Heavey**[2]

[1] School of Mechanical and Manufacturing Engineering, Dublin City University, Glasnevin, Dublin 9, Ireland (e-mail: john.geraghty@dcu.ie)
[2] Department of Manufacturing and Operations Engineering, University of Limerick, Limerick, Ireland (e-mail: cathal.heavey@ul.ie)

Abstract. In order to overcome the disadvantages of Kanban Control Strategy (KCS) in non-repetitive manufacturing environments, two research approaches have been followed in the literature in past two decades. The first approach has been concerned with developing new, or combining existing, pull-type production control strategies in order to maximise the benefits of pull control while increasing the ability of a production system to satisfy demand. The second approach has focused on how best to combine Just-In-Time (JIT) and Material-Requirements-Planning (MRP) philosophies in order to maximise the benefits of pull control in non-repetitive manufacturing environments. This paper provides a review of the research activities in these two approaches, presents a comparison between a Production Control Strategy (PCS) from each approach, and presents a comparison of the performance of several pull-type production control strategies in addressing the Service Level vs. WIP trade-off in an environment with low variability and a light-to-medium demand load.

Keywords: Hybrid Push/Pull – CONWIP/Pull – EKCS – BSCS – Kanban – Markov decision process – Discrete event simulation – Simulated annealing optimization algorithm

1 Introduction

The selection, implementation and management of an appropriate Production Control Strategy is an important tool to any organisation aiming to adopt a Lean Manufacturing Philosophy. Production control strategies that push products through the system based on forecasted customer demands are classified as Push-type production control strategies. Such strategies aim to maximise the throughput of the system

Correspondence to: J. Geraghty

so as to minimise shortage in supply and tend to result in excess work-in-progress inventory, WIP, that masks flaws in the system. Production control strategies that pull products through the system based on actual customer demands at the end of the line are classified as Pull-type production control strategies. Such strategies tend to minimise WIP and unveil flaws in the system at the risk of failure to satisfy demand. The advantages and disadvantages of push systems such as MRP and pull systems such as kanban controlled Just-In-Time have been well documented in the literature [11, 23, 24, 31]. In order to overcome the disadvantages of Kanban Control Strategy (KCS), two research approaches have been followed in the last two decades. The first approach has been concerned with developing new, or combining existing, pull-type production control strategies in order to maximise the benefits of pull control while increasing the ability of a production system to satisfy demand. The second approach has focused on how best to combine JIT and MRP philosophies in order to maximise the benefits of pull control in non-repetitive manufacturing environments. A hybrid production system could be characterised as a production system that combines elements of the two philosophies in order to minimise inventory and unmask flaws in the system while maintaining the ability of the system to satisfy demand. These research approaches are not mutually exclusive as there are intersections between these approaches. For instance, we classify CONWIP as a pull-type production control strategy, however CONWIP could also be considered as a hybrid Push/Pull production control strategy that utilises a pull-type control strategy to limit the amount of inventory in the line and a push-type control control strategy within the line to speed the progress of inventory toward the finished-goods buffer. In addition, Geraghty and Heavey [15] showed that under certain conditions the Horizontally Integrated Hybrid Production Control Strategy, HIHPS, favoured by Hodgson and Wang [20, 21] is equivalent to the pull-type production control strategy hybrid Kanban-CONWIP introduced by Bonvik and Gershwin [3], Bonvik et al. [2].

In this paper we firstly present a brief review of the research efforts in the design and development of both Pull-type PCS and Horizontally Integrated Hybrid Systems, HIHS (see Sects. 2 and 3 respectively). Section 4 compares the performance of one popular model of HIHS with the Pull-type PCS known as Extended Kanban Control Strategy, EKCS. Section 5 presents an experiment to explore the comparative performance of several Pull-type PCS in addressing the Service Level vs. WIP trade-off. Finally Section 6 presents a discussion of the main results of the experiments and research presented in this paper.

2 Pull-type production control strategies

The Kanban Control Strategy, KCS, developed by Toyota allows part flow in a Just-In-Time, JIT, line to be controlled by basing production authorisations on end-item demands. KCS is often referred to as a 'Pull' production control strategy since part demands travel upstream and pull products down the line by authorising production based on the presence of Kanban cards which are limited in number and circulated between production stages. KCS has been the focus of considerable research effort since the early 1980's. In particular, optimising the number and distribution of

Kanbans has received a lot of attention. However, in practice, Kanban distributions tend to be determined by implementing rules of thumb or simple formulae [2]. Berkley [1] provides a review of Kanban control literature, while Muckstadt and Tayur [26] provide a review of KCS mechanisms that have been developed.

The Basestock Control Strategy, BSCS, is the oldest pull-type production control strategy. The definitive paper on BSCS [7] was published in 1960. In a BSCS line the inventory points of each stage are initialised to predefined levels. When a demand event occurs, demand cards are transmitted to each production stage. These demand cards are matched with a part in the stage's input buffer to authorise production and are destroyed once production begins. Liberopoulos and Dallery [25] demonstrated that BSCS is equivalent to the Hedging Point Control System which has its origins in the work of Kimemia and Gershwin [22]. The primary advantage of BSCS is that it responds quickly to demand events. Every stage is informed instantly of demand events, unlike KCS where demand information must pass slowly upstream. However, BSCS has been criticised for the loose coordination provided between stages and the fact that it does not provide any guarantee to limit the number of parts that may enter the system [25]. Every demand event authorises the release of new parts into the system.

Generalised Kanban Control Strategy, GKCS, and Extended Kanban Control Strategy, EKCS, are both based on the integration of KCS and BSCS. GKCS was first proposed by [4, 36] and EKCS was proposed by [9, 10]. In both systems the inventory points are initialised to a predefined level as in BSCS and demand information is communicated to each stage in the line. The movement of parts between stages is coordinated by Kanbans as in KCS. The difference between the control structures employed in both systems is very subtle and has to do with how demand information is communicated to the individual production stages. In GKCS when a demand event occurs information about the demand is communicated to the final stage in the form of demand cards. Each demand card must be matched with a free Kanban. When this match occurs, a demand card is sent to the stage's immediate predecessor and production at the stage is authorised if the demand-Kanban match can be matched with a part. Therefore, demand information is not necessarily transferred instantly to all production stages. The arrival of demand information at a stage can be delayed if downstream stages fail to match the demand cards with Kanbans instantly. In an EKCS governed line demand information is communicated instantly to all production stages. Production is authorised when a demand card, a Kanban and a part are available. The advantages of EKCS over GKCS are, firstly, its comparative simplicity and secondly, the separation of the role of the basestock and Kanban parameters is clearly distinguishable, whereas in a GKCS system it is not [25].

Constant Work In Process or CONWIP has received a lot of research attention since it was first proposed by [29, 30]. Initially CONWIP was proposed as a Pull alternative to KCS and often referred to as an Order Release Mechanism as opposed to a Production Control Strategy. CONWIP was purported to bring the advantages of pull-control to non-repetitive manufacturing environments [29, 30]. The mechanism utilised by CONWIP is very simple. A limit known as the WIP Cap is placed on the amount of inventory that may be in the system at any given period of time.

Once this level of inventory has been achieved, inventory may not enter the system until a demand event removes a corresponding amount of inventory from the line. With only one parameter to optimise, i.e. the WIP Cap, CONWIP is very simple to implement and maintain. The main reason why CONWIP lines outperform KCS lines is that demand information is instantly communicated to the initial stage and the release rate is adjusted to match the demand rate. In a KCS line, as has been stated earlier, demand information has to travel upstream from the end-item inventory point to the initial stage. The longer the line and the more delays encountered at individual production stages (e.g. processing time, breakdown/repair time, set-up time etc.) the longer the information delay encountered. A disadvantage of CONWIP is that inventory levels are not controlled at the individual stages, which can result in high inventory levels building up in front of bottleneck stages.

Chang and Yih [6] introduced a Pull PCS named Generic Kanban System (GKS) applicable to dynamic, multi-product, non-repetitive manufacturing environments. KCS requires inventories of semi-finished products of each product type to be maintained at each production stage. In multi-product environments the amount of semi-finished inventory maintained in the line could be prohibitively large [6]. GKS operates by providing a fixed number of Kanbans at each workstation that can be acquired by any part. A part/job can only enter the system if it acquires a Kanban from each of the workstations in the system. GKS reduces to CONWIP if an equal number of Kanbans are distributed to all workstations. Comparison with a push-type production control strategy was favourable with GKS shown to be less susceptible to the position of the bottleneck. GKS outperformed KCS in terms of WIP required to achieve a desired Cycle Time. Comparison with CONWIP was favourable and GKS was shown to be more flexible in that by manipulating the number of Kanbans at each workstation the performance of GKS could be improved beyond that achieved by CONWIP. Chang and Yih [5] presented a simulated annealing algorithm for determining the optimal Kanban distribution for a GKS line.

In order to overcome the disadvantages of loose coordination between production stages in a CONWIP line Bonvik and Gershwin [3] and Bonvik et al. [2] proposed an alternative strategy, hybrid Kanban-CONWIP. In hybrid Kanban-CONWIP, as in CONWIP, an overall cap is placed on the amount of inventory allowed in the production system. In addition, inventory is controlled using Kanbans in all stages except the last stage. CONWIP can be considered as special case of hybrid Kanban-CONWIP in which there is an infinite number of Kanbans distributed to each production stage [2]. A comparison of KCS, minimal blocking KCS, BSCS, CONWIP and hybrid Kanban-CONWIP was presented in [2]. The different PCS were compared in a four-stage tandem production line using simulation. Each of the PCS were compared using constant demand and demand that had a stepped increase/decrease. It was found that the hybrid Kanban-CONWIP strategy decreased inventories by 10% to 20% over KCS while main taining the same service levels (percentage of demands instantaneously matched with a finished product). The performance of basestock and CONWIP strategies fell between those of KCS and hybrid Kanban-CONWIP.

Two papers that generalize hybrid Kanban-CONWIP are [14] and [13]. These papers propose a generic pull model that, as well as encapsulating the three basic

pull control strategies, KCS, CONWIP and BSCS, also allows customized pull control strategies to be developed. Simulation and an evolutionary algorithm were used to study the generic model. Details of the evolutionary algorithm are given in [14] while results on extensive experimentation on the effect of factors (i.e., line imbalance, machine reliability) on the proposed generic pull model are given in [13]. Gaury and Kleijnen [12] noted that Operations Research has traditionally concentrated on optimisation whereas practitioners find the robustness of a proposed solution more important. A methodology was presented in [12] that was a stagewise combination of four techniques: (i) simulation, (ii) optimization, (iii) risk or uncertainty analysis, and (iv) bootstrapping. Gaury and Kleijnen [12] illustrated their methodology through a production-control study for the four-stage, single product production line utilised by [2]. Robustness was defined in [12] as the capability to maintain short-term service, in a variety of environments; i.e. the probability of the short-term fill-rate (service level) remaining within a pre-specified range. Besides satisfying this probabilistic constraint, the system minimised expected long-term WIP. Four systems were compared in [12], namely Kanban, CONWIP, hybrid Kanban-CONWIP, and Generic. The optimal parameters found in [2] were used for KCS, CONWIP and hybrid Kanban-CONWIP. Gaury and Kleijnen [12] used a Genetic Algorithm to determine the optimal parameters for the Generic pull system. For the risk analysis step, seventeen inputs were considered; the mean and variance of the processing time for each of the four production stages, mean time between failures and mean time to repair per production stage, and the demand rate. The inputs were varied over a range of $\pm 5\%$ around their base values. Gaury and Kleijnen [12] concluded that in this particular example, hybrid Kanban-CONWIP was best when risk was not ignored; otherwise Generic was best and therefore, risk considerations can influence the selection of a PCS.

Each of the pull-type production control strategies discussed above, with the exception of GKCS, have one important advantage over KCS that ensures that they are more readily applicable to non-repetitive manufacturing environments. That advantage stems from the manner in which demand information is communicated in comparison to KCS. In KCS, demand information is not communicated directly to production stages that release parts/jobs into the system. Rather it is communicated sequentially up the line from the finished goods buffer as withdrawals are made by customer demands. This communication delay means that the pace of the production line is not adjusted automatically to account for changes in the demand rate. The arrival of demand information to the initial stages in a GKCS line might be delayed if the demand cards at a production stage in the line are not instantaneously matched with Kanban cards. BSCS, EKCS, CONWIP, GKS and hybrid Kanban-CONWIP all, however, communicate the demand information instantaneously to the initial stages allowing the release rate to be paced to the actual demand rate. For instance, Bonvik et al. [2] showed that if the demand rate decreases unexpectedly the impact on a CONWIP strategy and hybrid Kanban-CONWIP strategy would be for the finished-goods buffer to increase toward the WIP Cap with all intermediate buffers tending toward empty. The impact, however, on a KCS line would be that all the intermediate buffers would increase toward their maximum permissible limits.

Therefore, the KCS line would have semi-finished inventory distributed throughout the line.

3 Hybrid production control strategies

Hybrid control strategies can be classified into two categories: vertically integrated hybrid systems (VIHS) or horizontally integrated hybrid systems (HIHS) [8]. VIHS consist of two levels, usually an upper level push-type PCS and a lower level pull-type PCS. For example, Synchro MRP utilises MRP for long range planning and KCS for shop floor execution [17]. The main disadvantage of VIHS is that MRP calculations must be performed for each stage in the production system. This makes VIHS complex to implement and maintain and accounts for their relative lack of use in industry [20]. HIHS consist of one level where some production stages are controlled by push-type PCS and other stages by pull-type PCS. Only HIHS are considered in the discussion that follows.

Hodgson and Wang [20, 21] developed a Markov Decision Process (MDP) model for HIHS. The model was solved using both dynamic programming and simulation for several production strategies, including pure push and pure pull production strategies and strategies based on the integration of push and pull control. In this push/pull integration strategy each individual stage may push or pull. This type of control strategy is denoted as Hybrid Push/Pull in [20, 21]. Initially in [20], the research was applied to a four-stage semi-continuous production iron and steel works (see Fig. 1), with the first two stages in parallel and the remaining stages as serial production stages. In order to simplify the analysis the model assumes that the production process is a discrete time process and that demand per period and the amount of inventory are both integer multiples of a unit size. The research was later extended to a five-stage production system [21]. For both the four and five stage production systems, a strategy where production stages 1 and 2 (P_1 and P_2 in Fig. 1) push and all other stages pull was demonstrated to result in the lowest average gain (average system cost). Hodgson and Wang [21] stated that they had observed similar results for an eight-stage system and concluded that this strategy would be the optimal hybrid integration strategy for a J-stage system. Subsequent papers that use the model in [20, 21] or extensions of it are [11], [28] and [35].

Deleersnyder et al. [11] considered that the complexity of the control structure required for the successful implementation of Synchro MRP resulted in it being largely ignored by industry. Synchro MRP requires MRP control to be linked into every stage in the production line while utilising local kanban control to authorise production at each stage. Deleersnyder et al. [11] developed a hybrid production strategy that limited the number of stages into which MRP type information is added in order to reduce the complexity of the hybrid strategy in comparison to Synchro MRP, while realizing the benefits of integrating push and pull type control strategies. The model developed in [11] is similar to that presented in [20, 21] and comparable results were obtained for a serial production line.

Pandey and Khokhajaikiat [28] extended the model in [20, 21] to allow for the inclusion of raw material constraints at each stage. The modified model also allowed for a stage to require more than one item of inventory and/or more than one

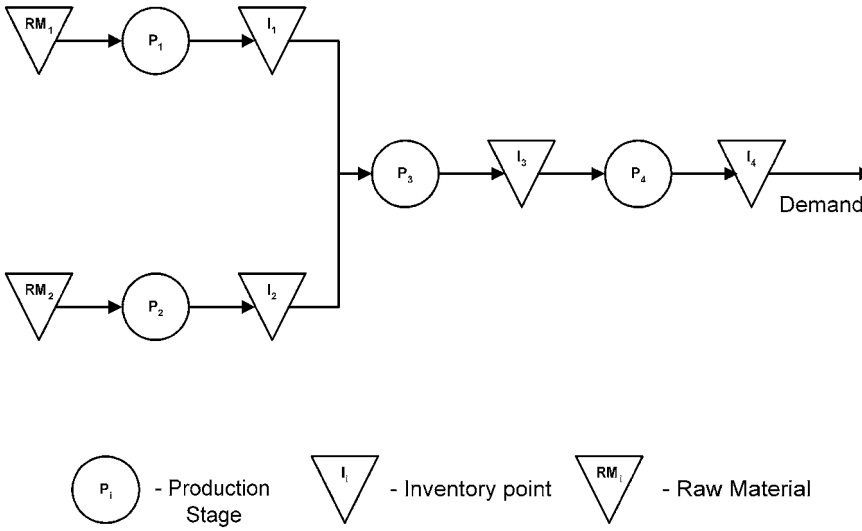


Fig. 1. Parallel/Serial four stage production system modelled by Hodgson and Wang [20]

item of raw material to produce a part. Pandey and Khokhajaikiat [28] presented results from two sets of experiments. In the first set they modelled a four-stage parallel/serial production line similar to the system shown in Figure 1. The initial production stages (P_1 and P_2 in Fig. 1) operated under raw material availability constraints, had different order purchasing and delivery distributions but had identical production unreliability. Sixteen integration strategies were considered. In the second experimental set the authors applied the raw material availability constraint to all stages of the production line. The authors concluded that the hybrid strategy in which the initial stages (P_1 and P_2) operate under push control and the remaining stages operate under pull control is the best strategy when raw material constraints apply only to the initial stages. When the raw material availability constraint is applied to all stages the push strategy becomes the optimal control strategy. For systems with large variability in demand none of the strategies dominated.

Wang and Xu [35] presented an approach that facilitated the evaluation of a wide range of topologies that utilize hybrid push/pull. They used a structure model to describe a manufacturing system's topology. Their methodology was used to investigate four 45-stage manufacturing systems: (i) A single-material serial processing system; (ii) A multi-material serial processing system; (iii) A multi-part processing and assembly system, and (iv) A multi-part multi-component processing and assembly system. Wang and Xu [35] compared pure pull and push strategies against the optimal hybrid strategy found in [20, 21], where the initial stages push and all other stages pull. Their results suggest that the optimal hybrid strategy out-performs pure push or pull strategies.

Other models that implement hybrid push/pull control strategies similar to [20, 21] have been developed. Takahashi et al. [32] defined push/pull integration as a system in which there is a single junction point between push stages and pull stages. In Takahashi et al. [32] a model was presented to evaluate this control strategy. Two

subsequent papers, [33] and [34] further developed and experimented with this model. Hirakawa et al. [19] and Hirakawa [18] developed a mathematical model for a hybrid push/pull control strategy that allows each production stage to switch between push and pull control depending on whether demand can be forecasted reliably or not. Cochran and Kim [8] presents a HIHS with a movable junction point between a push sub-system and a pull sub-system. The control strategy presented had three decision variables: (i) the junction point, i.e., the last push stage in the HIHS; (ii) the safety stock level at the junction point; (iii) the number of kanbans for each stage in the pull sub-system. Simulation combined with simulated annealing was used to find the optimal decision variables for the control strategy.

4 Comparison of EKCS and the push-type PCS modelled by Hodgson and Wang

Several comparisons of Pull-PCS have been reported in the literature, for example [2, 12, 13, 25]. There has also been several comparisons between HIHS and KCS, for example [8, 27, 32, 33, 34, 20, 21, 11, 28, 35]. Comparisons between HIHS and other Pull-Type PCS are rare in the literature. Orth and Coskunoglu [27] included CONWIP, in addition to KCS, in the comparison analysis. In a previous paper [15] we demonstrated that the optimal HIHS selected by Hodgson and Wang [20, 21] where initial stages employ push control and all other stages employ pull control, is equivalent to a Pull-Type PCS, namely hybrid Kanban-CONWIP [3, 2]. As well as considering several alternative integration strategies, Hodgson and Wang [20, 21] also included a Push-Type and a Pull-Type PCS in their analysis, which they referred to as 'Pure Push' and 'Pure Pull' PCS. After examining the equations used in [20, 21] to model the 'Pure Push' PCS we felt that there were similarities to the control structure implemented by the Pull-Type PCS know as EKCS [9, 10]. Therefore in this section we explore these comparisons. The notation used in the remainder of this paper is shown in Table 1.

In Hodgson and Wang's 'Pure-Push' PCS, production is authorised when (i) sufficient space exists in the output buffer of the stage, (ii) sufficient inventory exists in the input buffer of the stage, (iii) sufficient production capacity exists at the stage, and (iv) downstream inventory levels have decreased below forecasted requirements necessary to meet expected demand. The MDP model presented in [20, 21] required the evaluation of two equations (i.e. the production trigger, $A_j(n)$, and the production objective, $PO_j(n)$ in order to determine production authorisations for a stage in period n,). For the purposes of the discussion presented here we have combined these equations to form a single equation for the number of production authorisations, $PA_j(n)$, available to a stage in period n. This was achieved without making any simplifying assumptions. $PA_j(n)$ for a system controlled by Hodgson and Wang's 'Pure-Push' PCS can be modelled by Eq. (1) where $1 \leq j \leq J-1$ and by Eq. (2) for the final production stage.

Table 1. Notation used in models presented

Notation	Description
$A_j(n)$:	Production trigger for stage j in period n.
$PO_j(n)$:	Production objective for stage j in period n.
$I_j^{\max}$:	Maximum capacity of inventory point j.
SS:	Desired safety stock level of finished product
NS_j:	The number of stages that succeed stage j (i.e., number of stages that components produced at stage j traverse after stage j before reaching the customer.
$D(n)$:	Forecasted demand in period n.
j, J:	Unique number identifying a production stage where $1 \leq j \leq J$.
n:	Production period.
$d(n)$:	The actual demand quantity in period n.
$PA_j(n)$:	The Production Authorisation for stage j in period n.
$P_j^{\min}$:	The minimum production capacity of stage j
$P_j^{\max}$:	The maximum production capacity of stage j
$P_j(n)$:	Production quantity for stage j in period n.
q:	The production reliability of a stage, which is modelled by a Probability Mass Function
I_j:	The output buffer of stage j.
$I_j(n)$:	The amount of inventory held in the output buffer of production stage j in period n.
$\{B_j(n)\}$:	The set of inventories held in the output buffers of the immediate predecessors of stage j in period n
$c_j(n)$:	The sum of inventories held in the output buffers of stages parallel to, but with stage number greater than, production stage j
K_j:	The number of Kanbans allocated to production stage j.
CC:	The cap on total inventory allowed in CONWIP and hybrid Kanban-CONWIP lines.
$DC_j(n)$:	Number of demand cards held at stage j in period n in BSCS and EKCS lines.
S_j:	The initialisation stock level for stage j in BSCS and EKCS lines
$S_j^{\min}$:	The minimum initialisation stock level for stage j in BSCS and EKCS lines
P_j:	Production center at stage j

$$PA_j(n) = \min\left\{I_j^{\max} - I_j(n-1), \max\left[P_j^{\min}, SS + (NS_j+1)\times D(n) - \left(\sum_{i=j}^{J} I_i(n-1) - c_j(n-1)\right)\right], \{B_j(n-1)\}, P_j^{\max}\right\}, \quad \forall j \leq J-1 \tag{1}$$

$$PA_J(n) = \min\left\{I_J^{\max} - I_J(n-1) + D(n), \max\left[P_J^{\min}, SS + D(n) - I_J(n-1)\right], \{B_J(n-1)\}, P_J^{\max}\right\} \tag{2}$$

It is possible for the term $I_J(n-1)$ to become negative. This occurs in the event of a shortage in period $n-1$, i.e. a failure to satisfy demand. Therefore the term

$I_J(n-1)$ is not only used to record the inventory in the finished goods buffer in period $n-1$ but also the backlog in period $n-1$. If a backlog occurs, i.e. $I_J(n-1)$ is negative, Eq. (2) would effectively result in a temporary increase of the maximum capacity of the finished goods inventory buffer. Equation (3) models the number of production authorisations available to the final stage where shortages are not permitted to temporarily increase the maximum capacity of the finished goods inventory buffer.

$$PA_J(n) = \min\left\{I_J^{\max} - \max\left[0, I_J(n-1)\right] + D\left(n\right), \max\left[P_J^{\min}, SS + D\left(n\right) - I_J(n-1)\right], \left\{B_J\left(n-1\right)\right\}, P_J^{\max}\right\} \tag{3}$$

Notice that Hodgson and Wang's model of Push control includes a limit on inventory in the output buffer of a stage, $I_j^{\max}$. Since a production stage does not become aware of a change in state of the buffer until the subsequent production period, $n+1$, and with the exception of the final stage does not attempt to predict the removal of inventory from the output buffer by immediate successors, this limit behaves similarly to Kanban control. However, the Push strategy modelled by [20, 21] is not equivalent to KCS since each stage has information about the status of line downstream from its output buffer, through the term $\sum_{i=j}^{J} I_i(n-1) - c_j(n-1)$ in Eq. (1) above. Since only a demand event can change the state of the downstream section of the line, in terms of total WIP, this is equivalent to demand cards being passed to each stage in the line. Therefore, it would appear that there is some similarity between the 'Pure-Push'PCS modelled by [20, 21] and EKCS.

In an EKCS system, a production stage has authorisation to produce a part when: (i) inventory is available in its buffer; (ii) a Kanban card is available and (iii) a demand card is available. The number of demand cards available to a production stage in period n is given by Eq. (4). In period n the number of production authorisations at stage j, $PA_j(n)$, is given by Eq. (5). For an EKCS system where the introduction of temporary Kanbans in the event of shortages is not permitted $PA_j(n)$ is modelled by Eq. (6) for the final production stage.

$$DC_j\left(n\right) = DC_j\left(n-1\right) - P_j\left(n-1\right) + d\left(n\right), \quad \forall j \leq J \tag{4}$$

$$PA_j\left(n\right) = \min\left\{K_j - I_j\left(n-1\right), \max\left[P_j^{\min}, DC_j\left(n\right)\right], \left\{B_j\left(n-1\right)\right\}, P_j^{\max}\right\}, \quad \forall j \leq J-1 \tag{5}$$

$$PA_J\left(n\right) = \min\left\{K_J - \max\left[0, I_J\left(n-1\right)\right], \max\left[P_J^{\min}, DC_J\left(n\right)\right], \left\{B_J\left(n-1\right)\right\}, P_J^{\max}\right\} \tag{6}$$

Let us assume a serial production line with J stages. The state transition equations for the inventory levels of the buffers, for either model, can be determined from Eqs. (7) and (8):

$$I_j(n) = I_j(n-1) + P_j(n) - P_{j+1}(n) \qquad \forall j \leq J-1 \tag{7}$$

$$I_J(n) = I_J(n-1) + P_J(n) - d(n) \tag{8}$$

From examining the equations for the two models, for both dynamic and static Kanban distributions, it is clear that in order for the PCS to be equivalent three conditions must be satisfied: (i) the inventory level of a buffer in the 'Pure-Push'

model must equate to the inventory level of the same buffer in the EKCS model in a given production period n. (ii) the Kanban distribution in EKCS must be equal to the buffer capacity limits in the 'Pure-Push' model.

$$K_j = I_j^{\max} \qquad \forall\, j \le J-1 \tag{9}$$

$$K_J = I_J^{\max} + D\,(n) \tag{10}$$

and (iii) the following two equalities must hold:

$$DC_J(n) = SS + D\,(n) - I_J\,(n-1) \tag{11}$$

$$DC_j\,(n) = \left[SS + (NS_j + 1) \times D\,(n) - \left(\sum_{i=j}^{J} I_i\,(n-1) - c_j\,(n-1) \right) \right] \qquad \forall j \le J-1 \tag{12}$$

Substituting Eq. (11) into Eq. (4) yields:

$$SS + D\,(n) - I_J\,(n-1) = SS + D\,(n) - I_J\,(n-2) - P_J\,(n-1) + d\,(n) \tag{13}$$

Re-writing Eq. (13) yields:

$$I_J\,(n-1) = I_J\,(n-2) + P_J\,(n-1) - d\,(n) \tag{14}$$

Equation (14) is not equivalent to Eq. (8) and therefore the two PCS are not equivalent. The primary difference between the two PCS stems from the method in which demand information is communicated to each stage. In EKCS demand information is instantly communicated to all stages. In the 'Pure-Push' PCS the communication of demand information is delayed by one period. In fact, the 'Pure-Push' PCS described by [20, 21] could more accurately be described as a Vertically Integrated Hybrid System, in which each production stage develops a forecast of production requirements for the production period through the term $SS + (NS_j + 1) \times D(n) - (\sum_{i=j}^{J} I_i(n-1) - c_j(n-1))$ and utilises kanbans to implement the production plan on the shop-floor plan. It would, therefore, be more accurate to refer to this PCS as Synchro MRP than 'Pure-Push'.

However, it is worth noting that the two PCS can be made equivalent if the demand information is communicated instantly in both PCS or delayed for one period in both PCS. For instance, communication of demand information in the 'Pure-Push' PCS can be made instantaneous by adjusting Eqs. (1), (2) and (3) by including a component to adjust the downstream inventory levels by the demand quantity $d(n)$ as shown by Eqs. (15), (16) and (17) below.

$$PA_j(n) = \min\left\{ I_j^{\max} - I_j(n-1), \max\left[P_j^{\min}, SS + (NS_j + 1) \times D\,(n) - \left(\sum_{i=j}^{J} I_i(n-1) - c_j(n-1) - d\,(n) \right) \right], \{B_j\,(n-1)\}, P_j^{\max} \right\}, \qquad \forall j \le J-1 \tag{15}$$

$$PA_J(n) = \min\left\{ I_J^{\max} - I_J\,(n-1) + D\,(n), \max\left[P_J^{\min}, SS + D\,(n) \right. \right.$$

$$-\left(I_J(n-1)-d\left(n\right)\right)],\left\{B_J\left(n-1\right)\right\},P_J^{\max}\} \tag{16}$$

$$PA_J(n)=\min\left\{I_J^{\max}-\max\left[0,I_J(n-1)\right]+D\left(n\right),\max\left[P_J^{\min},SS\right.\right.$$
$$\left.\left.+D\left(n\right)-\left(I_J(n-1)-d\left(n\right)\right)\right],\left\{B_J\left(n-1\right)\right\},P_J^{\max}\right\} \tag{17}$$

The PCS will be equivalent if the first two conditions stated earlier are met and the following two equalities hold:

$$DC_J(n)=SS+D\left(n\right)-\left(I_J\left(n-1\right)-d\left(n\right)\right) \tag{18}$$

$$DC_j\left(n\right)=\left[SS+\left(NS_j+1\right)\times D\left(n\right)-\left(\sum_{i=j}^{J}I_i\left(n-1\right)-c_j\left(n-1\right)-d\left(n\right)\right)\right]$$
$$\forall j\leq J-1 \tag{19}$$

Substituting Eq. (18) into Eq. (4) yields:

$$SS+D\left(n\right)-\left(I_J\left(n-1\right)-d\left(n\right)\right)=SS+D\left(n\right)-\left(I_J\left(n-1\right)-d\left(n-1\right)\right)$$
$$-P_J\left(n-1\right)+d\left(n\right) \tag{20}$$

Re-writing Eq. (20) yields:

$$I_J\left(n-1\right)=I_J\left(n-2\right)+P_J\left(n-1\right)-d\left(n-1\right) \tag{21}$$

Equation (21) is equivalent to Eq. (8) and therefore requires no further proof.

Substituting Eq. (19) into Eq. (4) yields:

$$\left[SS+\left(NS_j+1\right)\times D\left(n\right)-\left(\sum_{i=j}^{J}I_i(n-1)-c_j(n-1)-d\left(n\right)\right)\right]=$$
$$\left[SS+\left(NS_j+1\right)\times D\left(n\right)-\left(\sum_{i=j}^{J}I_i(n-2)-c_j(n-2)-d\left(n-1\right)\right)\right]$$
$$-P_j\left(n-1\right)+d\left(n\right) \tag{22}$$

Re-writing Eq. (22) yields:

$$\left(\sum_{i=j}^{J}I_i(n-1)-c_j(n-1)\right)=\left(\sum_{i=j}^{J}I_i(n-2)-c_j(n-2)\right)$$
$$+P_j\left(n-1\right)-d\left(n-1\right) \tag{23}$$

We can use the state transition equations (i.e. Eqs. (7) and (8)) to prove Eq. (23) as shown below. Note that in a serial line the terms $c_j(n-1)$ and $c_j(n-2)$ are both zero.

$$
\begin{array}{llll}
I_j\,(n-1) & = I_j\,(n-2) & + P_j\,(n-1) & - P_{j+1}\,(n-1) \\
I_{j+1}\,(n-1) & = I_{j+1}\,(n-2) & + P_{j+1}\,(n-1) & - P_{j+2}\,(n-1) \\
I_{j+2}\,(n-1) & = I_{j+2}\,(n-2) & + P_{j+2}\,(n-1) & - P_{j+3}\,(n-1) \\
\vdots & = \vdots & + P_{j+3}\,(n-1) & - \vdots \\
\vdots & \vdots\ \vdots & \vdots\ \vdots & \vdots\ \vdots \\
\vdots & = \vdots & + \vdots & - P_{J-1}\,(n-1) \\
I_{J-1}\,(n-1) & = I_{J-1}\,(n-2) & + P_{J-1}\,(n-1) & - P_J\,(n-1) \\
I_J\,(n-1) & = I_J\,(n-2) & + P_J\,(n-1) & - d\,(n-1) \\
\hline
\multicolumn{4}{l}{\sum_{i=j}^{J} I_i\,(n-1) = \sum_{i=j}^{J} I_i\,(n-2) + P_j\,(n-1) - d\,(n-1)}
\end{array}
$$

The initialisation stock levels for the buffers for both models can be determined from Eqs. (4) and (19). For instance, assume that the initial number of demand cards, $DC_j(0)$, the production in the previous period, $P_j(0)$, and the demand in the previous period $d(0)$ are all zero. Therefore, from Eq. (4) the number of demand cards available to each production stage in the first period, $DC_j(1)$, will be zero. The initialisation stocks for both models can be calculated from Eq. (19) as follows:

$$0 = \left[SS + (NS_j + 1) \times D\,(n) - \left(\sum_{i=j}^{J} I_i(0) - c_j(0) \right) \right] \tag{24}$$

$$\sum_{i=j}^{J} I_i(0) - c_j(0) = SS + (NS_J + 1) \times D\,(n) \tag{25}$$

Therefore, for the final production stage the initialisation stock level should be:

$$I_J(0) - c_J(0) = SS + (NS_j + 1) \times D\,(n) \tag{26}$$

Since it is the final production stage, the term $c_J(0)$ on the LHS will equal zero and on the RHS the term NS_J will equal zero. Therefore:

$$I_J(0) = SS + D\,(n) \tag{27}$$

For stage $J-1$ the initialisation stock level will be:

$$I_J(0) + I_{J-1}(0) - c_{J-1}(0) = SS + (NS_{J-1} + 1) \times D\,(n) \tag{28}$$

Given that $c_{J-1}(0) = 0$, $NS_{J-1} = 1$ and substituting in Eq. 27, then:

$$SS + D\,(n) + I_{J-1}(0) = SS + 2 \times D\,(n) \tag{29}$$

$$I_{J-1}(0) = SS + 2 \times D\,(n) - SS - D\,(n) \tag{30}$$

$$I_{J-1}(0) = D\,(n) \tag{31}$$

In fact it can be shown that for $1 \leq j \leq J-1$ the appropriate choice for initialisation stock is $I_j(0) = D(n)$. Of course, if the initial number of demand cards available to production stages is not zero then appropriate initialisation stock levels for both models can be determined in a similar manner from Eq. (19). Therefore, EKCS and Hodgson and Wang's 'Pure-Push' PCS are equivalent if:

(1) The demand event is communicated instantaneously in the 'Pure-Push' PCS or delayed by one production period in the EKCS PCS,
(2) The Kanban distribution for EKCS and buffer capacity limits for the 'Pure-Push' PCS are equivalent,
(3) The initialisation stocks of both models are equivalent and calculated from Eq. (19) for all stages, i.e. $j = 1, ..., J$, and
(4) The forecasted demand quantity, $D(n)$, in the 'Pure-Push' PCS is constant for all values of n.

5 Comparison of pull-type PCS

We now turn our attention toward examining the comparative performance of several Pull-Type PCS. The PCS examined are KCS, CONWIP, hybrid Kanban-CONWIP, BSCS and EKCS. The study presented here differs from Bonvik et al. [2] and Gaury and Kleijnen [12] as EKCS is included in the analysis, variable demand is used and unsatisfied demand is backlogged rather than being treated as a lost opportunity.

The system modelled for the purposes of these experiments was the five-stage parallel/serial line described by Hodgson and Wang [21]. The line produces a single product type produced from two components. In the line, stages 1 and 2 operate in parallel to input the two components to the system that are assembled on a one-to-one ratio at stage 3. Stages 3, 4 and 5 are in series. The output buffer of stage 5 is the finished goods buffer, from which all demands must be satisfied. Demand in a given period, n, is either 3 or 4 units with equal probability. For the purposes of the experimental work presented here it is assumed that minimum production level ($P_j^{\min}$) of stage j in period n is zero. The reliability of stage j in period n was modelled by the Probability Mass Function given in Table 2. Noting that q and $PAj(n)$ are independent, the probability that stage j produces q units in period n given that the production authorisation is $PA_j(n)$, i.e. $\Pr[P_j(n) = q|PA_j(n)]$, is given by:

$$\Pr\left[P_j(n) = q|PA_j(n)\right] = \Pr\left[P_j(n) = q\right], \qquad q = 0, 1, \ldots, PA_j(n) - 1 \tag{32}$$

$$\begin{aligned}\Pr\left[P_j(n) = PA_j(n)|PA_j(n)\right] &= \Pr\left[P_j(n) = PA_j(n)\right] \\ &+ \Pr\left[P_j(n) = PA_j(n) + 1\right] \\ &+ \ldots + \Pr\left[P_j(n) = P_j^{\max}\right] \quad q \geq PA_j(n)\end{aligned} \tag{33}$$

Table 2. Probability mass function for reliability in production of individual stages

q	3	4	5
$\Pr[q]$	0.2	0.6	0.2

The remainder of this section details the models that were developed for each PCS examined, the experiment design and the results from the experiment. The

models have been developed by the authors with reference to the notation and methodologies employed by [20, 21], and [28].

5.1 Kanban control strategy

In a KCS system, production at stage j is authorised by the presence of Kanban cards and parts. When stage j begins production on a part, a Kanban card is attached to the part and travels downstream with the part. When the succeeding stage begins production on the part the Kanban card is removed and passed back to stage j to be available to authorise production of a new part. The Production Authorisation for period n for KCS stage j, where $1 \leq j \leq J-1$, is obtained from Eq. (34). The Production Authorisation for the final stage is obtained from Eq. (35) and is different from the model in [20, 21] in that the number of Kanbans available to the final stage cannot be increased temporarily in response to a shortage.

$$PA_j\,(n) = \min\left[K_j - I_j\,(n-1)\,, \{B_j\,(n-1)\}\,, P_j^{\max}\right], \quad \forall j \leq J-1 \tag{34}$$

$$PA_J\,(n) = \min\left[K_J - \max\left[0, I_J\,(n-1) - d(n)\right], \{B_J\,(n-1)\}\,, P_J^{\max}\right] \tag{35}$$

5.2 CONWIP control strategy

For CONWIP systems, $PA_j(n)$ for an input stage ($j = 1, 2$) was modelled by Eq. (36). $PA_j(n)$ for an input stage is constrained by a cap (CC) on the total inventory in the system, the number of components available in the raw material buffers and the maximum production capacity of the stage. For the purposes of the experiments conducted raw material was assumed to be always available. For this situation the term $\{B_j(n-1)\}$ would be infinitely large. $PA_j(n)$ for all other stages is only constrained by the maximum amount of units that the stage can produce in a production period and the availability of components in the stage's input buffer. Therefore, Eq. (37) was used to model $PA_j(n)$ for all stages that are not input stages, i.e $3 \leq j \leq J$,.

$$PA_j\,(n) = \min\left[CC - \left(\left(\sum_{i=j}^{J} I_i\,(n-1)\right) - c_j\,(n-1) - d\,(n)\right),\right.$$
$$\left.\{B_j\,(n-1)\}\,, P_j^{\max}\right], \quad 1 \leq j \leq 2 \tag{36}$$

$$PA_j\,(n) = \min\left[\{B_j\,(n-1)\}\,, P_j^{\max}\right], \quad 3 \leq j \leq J \tag{37}$$

5.3 Hybrid Kanban-CONWIP control strategy

Production Authorisations for production stages in a hybrid Kanban-CONWIP system were determined by combining the equations used to model $PA_j(n)$ for KCS and CONWIP. For an input stage of a hybrid Kanban-CONWIP system ($j = 1, 2$), $PA_j(n)$ was modelled by Eq. (38). This equation was developed by further constraining Eq. (35) such that sufficient Kanbans must also be available at the

stage to authorise production. For stage j, where $3 \leq j \leq J-1$, $PA_j(n)$ was modelled by Eq. (34). For the final stage $PA_J(n)$ was modelled by Eq. (37) where $j = J$.

$$PA_j(n) = \min \left[CC - \left(\left(\sum_{i=j}^{J} I_i(n-1) \right) - c_j(n-1) - d(n) \right), \right. \\ \left. K_j - I_j(n-1), \{B_j(n-1)\}, P_j^{\max} \right], \quad 1 \leq j \leq 2 \tag{38}$$

5.4 Basestock control strategy and extended Kanban control strategy

In a system employing BSCS, production at stage j in period n is authorised by the presence of demand cards at the production stage. When a demand occurs the equivalent number of demand cards are dispatched to each production stage to authorise the production of new parts. When the stage begins production of a new part the demand card is destroyed. The number of demand cards available to production stage j in period n, $DC_j(n)$, was determined from Eq. (39). $PA_j(n)$ for a BSCS system was determined by employing Eq. (40).

$$DC_j(n) = DC_j(n-1) - P_j(n-1) + d(n), \forall j \leq J \tag{39}$$

$$PA_j(n) = \min \left[DC_j(n), \{B_j(n-1)\}, P_j^{\max} \right], \quad \forall j \leq J \tag{40}$$

The production in period n of stage j in an EKCS system is constrained by the availability of Kanban and Demand cards. When a demand occurs, as with BSCS, the equivalent number of demand cards are dispatched to each production stage to authorise the production of new parts. However, before production can be authorised by the presence of a demand card, the demand card must be matched with a Kanban card and an available part. A demand card is destroyed when stage j begins production on the part while the associated Kanban card is attached to the part and travels downstream with the part. When the succeeding stage begins production on the part the Kanban card is removed and passed back to stage j to be available to authorise production of a new part. For an EKCS system the number of demand cards available to production stage j in period n, $DC_j(n)$, was also modelled by Eq. (39) while $PA_j(n)$ for an EKCS system was determined by employing Eq. (41) for $1 \leq j \leq J-1$ and Eq. (42) for the final production stage, i.e. $j = J$.

$$PA_j(n) = \min \left[DC_j(n), K_j - I_j(n-1), \{B_j(n-1)\}, P_j^{\max} \right], \\ \forall j \leq J-1 \tag{41}$$

$$PA_J(n) = \min \left[DC_J(n), K_J - \max\left[0, I_J(n-1) - d(n)\right], \right. \\ \left. \{B_J(n-1)\}, P_J^{\max} \right] \tag{42}$$

5.5 Experimental conditions

The models just described for each PCS were translated into discrete event simulation models in eM-Plant, an object-oriented simulation software tool developed

by Tecnomatix Technologies Ltd. The powerful debugging environment within eM-Plant was utilised to conduct a step-by-step walk through of each simulation model in order to verify the timing and accuracy of the calculations of the conceptual models had been correctly encapsulated in the models. In order to validate the simulation models, we firstly developed simulation models of the various PCS explored by [20, 21]. These models were validated against the results published in [20, 21] and the results were presented in Geraghty and Heavey [15]. In order to validate the individual simulation models developed for the PCS examined in this section we conducted the following:

(1) The output of our simulation model of KCS was compared to the output of our validated simulation model of Hodgson and Wang's conceptual model of KCS. The results were identical when the assumption that a backlog could not temporarily increase the number of kanbans available to the final stage was incorporated into our simulation model of Hodgson and Wang's conceptual model of KCS.
(2) In Geraghty and Heavey [15] we showed mathematically that the optimal HIHS identified by [20, 21] is equivalent to hybrid Kanban-CONWIP, under certain conditions. We also demonstrated this equivalence by comparing the outputs of our simulation model of hybrid Kanban-CONWIP with our validated simulation model of Hodgson and Wang's optimal HIHS.
(3) In this paper we have demonstrated mathematically that EKCS and Hodgson and Wang's 'Pure-Push' PCS will give equivalent results if the occurrence of demand events are communicated at the same time in both PCS and other conditions detailed earlier are met. Results are presented in [16] that demonstrate that our simulation model of EKCS achieves the same results as our validated simulation model of Hodgson and Wang's 'Pure-Push' PCS when all conditions for equivalence are met.
(4) It was not possible to validate our models of CONWIP and BSCS. However, these PCS are simplifications of hybrid Kanban-CONWIP and EKCS, respectively, in which kanbans are not distributed. Therefore, since we have been able to validate our simulation models of hybrid Kanban-CONWIP and EKCS, we assume that our simulation models of CONWIP and BSCS are valid.

For the purposes of the experimental process the simulation run-time over which statistics were collected was $10,000$ periods with a warm-up period of $1,000$ periods. Ten replications of each simulation were conducted. The PCS were compared by conducting a partial enumeration of the solution space for their control parameters. A detailed description of the solution spaces evaluated for each PCS for each demand distribution is described below.

The comparison of the strategies was achieved by conducting a partial enumeration of the control parameters of the five PCS examined. The minimum values for the Kanban allocations for the KCS, EKCS and hybrid Kanban-CONWIP models were eight for each stage. This was selected since preliminary work indicated that values below this level significantly degraded the solution. For instance setting the Kanban levels of the input stages ($j = 1, 2$) equal to 7 always resulted in a Service

Level of 0 regardless of the number of Kanbans allocated to the remaining stages for both KCS and hybrid Kanban-CONWIP.

CONWIP Cap, CC, values below 16 resulted in service levels of less than 10% for both CONWIP and hybrid Kanban-CONWIP. A minimum of four parts was selected for the initialisation stocks (S_j) for both BSCS and EKCS. This value was selected because (i) the nature of the control strategies implies that the initialisation stocks must be greater than zero and (ii) mean demand was 3.5 and it was desirable to initialise the buffers such that they could satisfy the mean demand. For KCS and hybrid Kanban-CONWIP the maximum value for the number of Kanbans considered for distribution to workstations 1 to 3 was 16 each with a maximum of 20 to workstation 4. For KCS workstation 5 had an upper bound of 20 Kanbans. For CONWIP and hybrid Kanban-CONWIP the maximum value considered for CC was 50. For BSCS the upper bounds for the initialisation stocks of workstations 1 to 5 were 12, 12, 12, 16 and 50 respectively. For each simulation run the models of the individual PCS were initialised with inventory as described by Table 3.

For the EKCS model it would have been impossible to conduct a partial enumeration of the solution space for all parameters (i.e. all possible combinations of Kanban and initialisation stock levels). The amount of computer time required would not have been feasible. For instance, suppose a partial enumeration of the solution space for the EKCS model were conducted with minimum values as described above and maximum values for K_j and S_j equal to the maximum values for K_i for the KCS model. Over 90,000,000 hours of CPU time would have been required to conduct this experiment (based on 5.3 seconds per replication and 10 replications per iteration on a 1.8GHz Intel Pentium 4 Dell PC with 256Mb of RAM). Therefore, in order to minimise the time requirements a method had to be found to predetermine the Kanban distribution or the initialisation stock levels. Dallery and Liberopoulos [10] noted that the production capacity of the EKCS only depends on K_j and not on S_j; $i = 1, \ldots, J$. They suggest ed that a reasonable design procedure for the EKCS could be to first design parameters K_j to obtain a desirable production capacity level, and subsequently design parameters S_j to obtain a desirable customer satisfaction level.

It seemed that a reasonable design for the Kanban allocation for the EKCS model might be the allocation that achieved 100% Service Level for the hybrid Kanban-CONWIP model. Therefore, it is not claimed that EKCS was compared for optimality with the other PCS. Just that a reasonable design for EKCS was compared. Under hybrid Kanban-CONWIP Kanbans are not allocated to the final stage since the maximum amount of inventory that can be in the output buffer of the final stage in any period is CC. Therefore, if it is desired to design the Kanban allocation for the EKCS such that it has at most the equivalent amount of inventory as a hybrid Kanban-CONWIP line then the number of Kanbans to allocate to the final stage for the EKCS model would be the maximum inventory from hybrid Kanban-CONWIP minus the minimum inventory to be allocated to the internal buffers in

the EKCS design, i.e. $CC - 12$[1]. The Kanban allocation for EKCS therefore was 10, 10, 15, 9 and 13 for workstations 1 to 5 respectively. These values were also the set as the maximum initialisation stock levels, $S_j^{\max}$, for each workstation.

Table 3. Initialisation levels for each Buffer under each PCS

Strategy	I_1	I_2	I_3	I_4	I_5
KCS	K_1	K_2	K_3	K_4	K_5
CONWIP	0	0	0	0	CC
Hybrid Kanban-CONWIP	0	0	0	0	CC
BSCS	S_1	S_2	S_3	S_4	S_5
EKCS	S_1	S_2	S_3	S_4	S_5

5.6 Experimental results

Of the five PCS examined, KCS was consistently the worst performer in terms of addressing the Service Level vs. WIP trade-off. Table 4 illustrates this by giving the percentage reduction in minimum WIP required by each PCS to achieve a targeted Service Level when compared to KCS. hybrid Kanban-CONWIP was consistently the best performer, requiring 9% to 15.5% less WIP than KCS to achieve a targeted Service Level. A paired-t test demonstrated that the performance of hybrid Kanban-CONWIP was statistically significantly better than CONWIP at both 95% and 99% significance levels. BSCS and EKCS required on average 8% to 13.5% less WIP than KCS to achieve a targeted Service Level. A paired-t test demonstrated that the performance of EKCS was statistically significantly better than BSCS at both 95% and 99% significance levels for all targeted service levels with the exception of a targeted Service Level of 96%. Tables 5 and 6 illustrate the inventory placement patterns achieved by each PCS for targeted service levels of 100% and 99.9% respectively. KCS required more semi-finished inventory than the other four PCS and a similar amount of end-item inventory as CONWIP and hybrid Kanban-CONWIP to achieve a targeted Service Level. While the differences between the other four PCS in terms of total WIP was small, the inventory placement patterns of the PCS were different. CONWIP and hybrid Kanban-CONWIP tended to maintain less WIP in semi-finished inventory and more in the end-item buffer than BSCS and EKCS.

6 Discussion

In the last two decades researchers have followed two approaches to developing production control strategies to overcome the disadvantages of KCS in non-repetitive

[1] CC is a component based inventory cap, therefore the internal inventory for a component in this parallel/serial model in period n is $I_1(n) + I_3(n) + I_4$(n) or $I_2(n) + I_3(n) + I_4(n)$ and the value 12 is arrived at as $S_1^{\min} + S_3^{\min} + S_4^{\min}$ or $S_2^{\min} + S_3^{\min} + S_4^{\min}$

Table 4. Percentage reduction over KCS in minimum inventory required by each PCS to achieve a targeted service level

SL $\geq$	100%	99.9%	99%	98%	97%	96%
CONWIP	8.9%	14.0%	14.3%	15.1%	14.2%	10.2%
Hybrid Kanban-CONWIP	9.0%	14.5%	14.8%	15.5%	14.7%	13.0%
BSCS	7.8%	12.8%	12.8%	13.3%	12.5%	12.8%
EKCS	7.9%	12.9%	12.9%	13.5%	12.6%	8.5%

Table 5. Inventory placements under optimal parameters for each PCS for targeted service level of 100%

	KCS	CONWIP	Hybrid Kanban-CONWIP	BSCS	EKCS
I_1	4.3118	3.9492	3.9365	4.2644	4.2500
I_2	4.3113	3.9473	3.9346	4.2625	4.2480
I_3	4.8917	3.7785	3.8283	4.0694	4.1127
I_4	5.0808	3.7794	3.7499	6.6772	4.0277
I_5	9.0833	9.7595	9.7341	6.2555	8.8573
Internal	18.5956	15.4544	15.4493	19.2736	16.6384
Total	27.6790	25.2139	25.1833	25.5291	25.4958

Table 6. Inventory placements under optimal parameters for each PCS for targeted service level of 99.9%

	KCS	CONWIP	Hybrid Kanban-CONWIP	BSCS	EKCS
I_1	4.3118	3.9486	3.9075	4.2644	4.2500
I_2	4.3113	3.9469	3.9053	4.2625	4.2480
I_3	4.8917	3.7779	3.7708	4.0694	4.1127
I_4	5.0808	3.7794	3.7746	4.0558	4.0277
I_5	6.0843	5.7608	5.7313	4.8784	4.8593
Internal	18.5956	15.4528	15.3581	16.6522	16.6384
Total	24.6799	21.2136	21.0895	21.5305	21.4977

manufacturing environments. The first approach has been to develop new or combine existing Pull-type PCS while the second approach has been to develop hybrid PCS based on combining elements of Push and Pull PCS. In a previous paper [15] it was demonstrated that the optimal HIHS selected by Hodgson and Wang [20, 21] where initial stages employ push control and all other stages employ pull control, is equivalent to a Pull-type PCS, namely hybrid Kanban-CONWIP [3, 2]. Here it was shown that the 'Pure-Push' PCS modelled by Hodgson and Wang [20, 21] would

be more accurately described as a vertical integration production control strategy, since each production stage forecasts its production requirements and utilises kanbans to control shop-floor production for each production period. However, it was also shown that by ensuring that demand information in the 'Pure-Push' PCS is communicated the instant it occurs rather than been delayed for one period the 'Pure-Push' PCS is equivalent to EKCS.

Using the model presented in Hodgson and Wang [21] a comparative study of KCS, CONWIP, hybrid Kanban-CONWIP, BSCS and EKCS was carried out. The criterion used in the study was the Service Level vs. WIP trade-off. KCS performed worst in terms of addressing this trade-off in that KCS consistently required more inventory than the other four PCS to achieve a targeted Service Level. The reason for the poor performance of KCS is due to the information delay that occurs in a KCS line. When a demand event occurs this information is only communicated to the final production stage to authorise production of replacement parts. The longer the line and more delays that occur in the system such as downtime due to machine unreliability, the longer the delay in communicating the demand information to initial stages. Therefore, the release rate is not easily adjusted to match changes in the demand rate. CONWIP and hybrid Kanban-CONWIP employ limits on inventory in the system and once this limit has been reached only the occurrence of a demand event can authorise the release of a part into the system. The release rate is therefore paced to match the demand rate. BSCS and EKCS use demand cards that are instantly communicated to each production stage to pace the production rate of the line to the demand rate.

For the system modelled, the demand rate was 3.5 parts per production period, which was 87.5% of the isolated production rate of a stage (4 parts per period). The coefficient of variation of the demand distribution was approximately 14% and the coefficient of variation of the production rate of a stage in isolation was approximately 16%. This therefore is a system with low variability and a light-to-medium demand load. For this system there was minimal difference between the performances of the various PCS examined, with the exception of KCS. A statistical analysis of the data however revealed these differences to be statistically significant. For this system, hybrid Kanban-CONWIP performed the best in addressing the Service Level vs. WIP trade-off.

EKCS tended to maintain similar overall inventory levels as CONWIP, hybrid Kanban-CONWIP and BSCS. However, EKCS tended to maintain more of this inventory internally in the line, i.e. in a semi-finished state, than CONWIP and hybrid Kanban-CONWIP. This may be either an advantage or disadvantage and will depend on the manufacturing objectives of the organisation. The strategy of the organisation might be to maintain as much as possible of the WIP in a finished state and thereby provide the organisation with greater flexibility to respond to unexpected demands. If this is the strategy of the organisation then hybrid Kanban-CONWIP is the preferable PCS for the manufacturing system. On the other hand the strategy of the organisation might be to maintain WIP in semi-finished states close to the completion state allowing the organisation to respond to changes in customer demands by reassigning WIP to other customers or altering the WIP to

meet new customer specifications. If this is the strategy of the organisation then EKCS is the preferable PCS for the manufacturing system.

Finally, as has been stated, the experiment presented here to examine the comparative performance of various Pull-type PCS was for a manufacturing system with moderate variability and a light-to-medium demand load. Future planned work is to examine the comparative performance of the five PCS further by examining how the PCS respond as the coefficient of variation of the demand distribution increases and as the mean of the demand distribution approaches the maximum capacity of the manufacturing system.

References

1. Berkley BJ (1992) A review of the kanban production control research literature. Production and Operations Management 1(4): 393–411
2. Bonvik AM, Couch CE, Gershwin SB (1997) A comparison of production-line control mechanisms. International Journal of Production Research 35(3): 789–804
3. Bonvik AM, Gershwin SB (1996) Beyond kanban – creating and analyzing lean shop floor control policies. In: Proceedings of Manufacturing and Service Operations Management Conference, Dartmouth College, The Amos Tuck School, Hanover, NH, USA
4. Buzacott JA (1989) Queueing models of kanban and MRP controlled production systems. Engineering Cost and Production Economics 17: 3–20
5. Chang T-M, Yih Y (1994a) Determining the number of kanbans and lotsizes in a generic kanban system: A simulated annealing approach. International Journal of Production Research 32(8): 1991–2004
6. Chang T-M, Yih Y (1994b) Generic kanban systems for dynamic environments. International Journal of Production Research 32(4): 889–902
7. Clark AJ, Scarf H (1960) Optimal policies for the multi-echelon inventory problem. Management Science 6(4): 475–490
8. Cochran JK, Kim S-S (1998) Optimum junction point location and inventory levels in serial hybrid push/pull production systems. International Journal of Production Research 36(4): 1141–1155
9. Dallery Y, Liberopoulos G (1995) A new kanban-type pull control mechanism for multistage manufacturing systems. In: Proceedings of the 3rd European Control Conference vol 4(2), pp 3543–3548
10. Dallery Y, Liberopoulos G (2000) Extended kanban control system: combining kanban and base stock. IIE Transactions 32(4): 369–386
11. Deleersnyder JL, Hodgson TJ, King RE, O'Grady PJ, Savva A (1992) Integrating kanban type pull systems and MRP type push systems: Insights from a Markovian model. IEE Transactions 24(3): 43–56
12. Gaury EGA, Kleijnen JPC (2003) Short-term robustness of production management systems: A case study. European Journal of Operational Research 148: 452–465
13. Gaury EGA, Kleijnen JPC, Pierreval H (2001) A methodology to customize pull control systems. Journal of the Operational Research Society 52(7): 789–799
14. Gaury EGA, Pierreval H, Kleijnen JPC (2000) An evolutionary approach to select a pull system among kanban, CONWIP and hybrid. Journal of Intelligent Manufacturing 11(2): 157–167
15. Geraghty J, Heavey C (2004) A comparison of hybrid Push/Pull and CONWIP/Pull production inventory control policies. International Journal of Production Economics 91(1): 75–90

16. Geraghty JE (2003) An investigation of pull-type production control mechanisms for lean manufacturing environments in the presence of variability in the demand process. PhD, University Of Limerick, Ireland
17. Hall RW (1983) Zero inventories. Dow Jones-Irwin, Homewood, IL
18. Hirakawa Y (1996) Performance of a multistage hybrid push/pull production control system. International Journal of Production Research 44: 129–135
19. Hirakawa Y, Hoshino K, Katayama H (1992) A hybrid push/pull production control system for multistage manufacturing processes. International Journal of Operations and Production Management 12(4): 69–81
20. Hodgson TJ, Wang D (1991a) Optimal hybrid push/pull control strategies for a parallel multi-stage system: Part I. International Journal of Production Research 29(6): 1279–1287
21. Hodgson TJ, Wang D (1991b) Optimal hybrid push/pull control strategies for a parallel multi-stage system: Part II. International Journal of Production Research 29(7): 1453–1460
22. Kimemia J, Gershwin SB (1983) An algorithm for the computer control of a flexible manufacturing system. IIE Transactions 15(4): 353–362
23. Krajewski LJ, King BE, Ritzman LP, Wong DS (1987) Kanban, MRP, and shaping the manufacturing environment. Management Science 33(1): 39–57
24. Lee LC (1989) A comparative study of the push and pull productions systems. International Journal of Operations and Production Management 9(4): 5–18
25. Liberopoulos G, Dallery Y (2000) A unified framework for pull control mechanisms in multi-stage manufacturing systems. Annals of Operations Research 93: 325–355
26. Muckstadt JA, Tayur SR (1995) A comparison of alternative kanban control mechanisms. IIE Transactions 27: 140–161
27. Orth MJ, Coskunoglu O (1995) Comparison of push/pull hybrid manufacturing control strategies. In: Proceedings of Industrial Engineering Research, Nashville, TN, pp 881–890. IIE, Norcross, GA
28. Pandey PC, Khokhajaikiat P (1996) Performance modelling of multistage production systems operating under hybrid push/pull control. International Journal of Production Economics 43(1): 17–28
29. Spearman ML (1988) An analytical congestion model for closed production systems. Technical Report 88-23, Dept. of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL
30. Spearman ML, Woodruff DL, Hopp WJ (1990) CONWIP: A pull alternative to kanban. International Journal of Production Research 28(5): 879–894
31. Spearman ML, Zazanis MA (1992) Push and pull production systems: Issues and comparisions. Operations Research 40(3): 521–532
32. Takahashi K, Hiraki S, Soshiroda M (1991) Integration of push type and pull type production ordering systems. In: Proceedings of 1st China-Japan International Symposium on Industrial Management, Beijing, China, pp 396–401
33. Takahashi K, Hiraki S, Soshiroda M (1994) Push-pull integration in production ordering systems. International Journal of Production Economics 33(1–3): 155
34. Takahashi K, Soshiroda M (1996) Comparing integration strategies in production ordering systems. International Journal of Production Economics 44(1–2): 83–89
35. Wang D, Xu CC (1997) Hybrid push/pull production control strategy simulation and its applications. Production Planning and Control 8(2): 142–151
36. Zipkin P (1989) A kanban-like production control system: analysis of simple models. Working paper 89-1, Graduate School of Business, Columbia University, New York

Section IV: Stochastic Production Planning and Assembly

Planning order releases for an assembly system with random operation times

Sven Axsäter

Department of Industrial Management and Logistics, Lund University, Sweden
(e-mail: Sven.Axsater@iml.lth.se)

Abstract. A multi-stage assembly network is considered. A number of end items should be delivered at a certain time. Otherwise a delay cost is incurred. End items and components that are delivered before they are needed will cause holding costs. All operation times are independent stochastic variables. The objective is to choose starting times for different operations in order to minimize the total expected costs. We suggest an approximate decomposition technique that is based on repeated application of the solution of a simpler single-stage problem. The performance of our approximate technique is compared to exact results in a numerical study.

Keywords: Multi-stage production/inventory systems – Decomposition

1 Introduction

In this paper we consider the planning of interrelated assembly operations with independent stochastic operation times. One or more end items should according to a given contract be delivered at a certain time. The delivery cannot take place until all end items are ready. In case the given delivery requirement cannot be satisfied there is a delay cost that is proportional to the length of the delay. If the end items are ready at different times the delay cost is based on the time when all items are ready. Furthermore, if end items are ready earlier than the delivery time, holding costs are incurred. A final assembly operation can normally not start unless a set of preceding operations, also with stochastic durations, has been completed. Delays for such preceding operations will not result in any direct delay costs but may indirectly result in additional delay costs for the end items. If such preceding operations are finished before the corresponding final operations start, holding costs are incurred. The operations preceding the final operations can, in turn, have preceding operations and so on. Our purpose is to find starting times for the different

operations that minimize the total expected costs, i.e., in other words we are looking for optimal safety times.

The considered problem with several stages is, in general, too difficult to be solved exactly. We therefore suggest a heuristic that is based on successive applications of the solution of a simpler one-stage problem. Different versions of such simpler problems with one or two stages have been studied in several papers before. Examples of this research are Yano (1987a), Kumar (1989), Hopp and Spearman (1993), Chu et al. (1993), and Shore (1995). Song et al. (2000) consider stochastic operation times as well as stochastic demand. In a recent overview Song and Zipkin (2003) consider a more general class of stochastic assembly problems. There are also a number of papers dealing with similar problems for other types of systems. Gong et al. (1994) consider a serial system and show that the problem of choosing optimal lead-times is equivalent to the well-known model in Clark and Scarf (1960). Yano (1987b) deals also with a serial system, while Yano (1987c) considers a distribution-type system. Examples of papers analyzing related problems for single-stage systems are Buzacott and Shanthikumar (1994), Hariharan and Zipkin (1995), Chen (2001), and Karaesmen et al. (2002).

The outline of this paper is as follows. We first give a detailed problem formulation in Section 2. In Section 3 we consider the simpler single-stage system that is the basis for our heuristic. The approximate procedure is then described in Section 4. In Section 5 we apply our technique to two sets of sample problems, and finally we give some concluding remarks in Section 6.

2 Problem formulation

We consider an assembly network (see Fig. 1). The arcs represent the operations. The node where operation i starts is denoted node i. The operation times are independent random variables with continuous distributions. Our purpose is to plan production so that the expected holding and delay costs are minimized.

Let us introduce the following notation:

$\mathrm{t_i}$ = starting time for operation i,
τ_i = stochastic duration time of operation i,
$\mathrm{f_i(x)}$ = density for τ_i,
$\mathrm{F_i(x)}$ = cumulative distribution function for τ_i, (It is assumed that $\mathrm{F_i(x)} < 1$ for any finite x.)
$\mathrm{t_d}$ = requested delivery time for the assembly,
$\mathrm{e_i}$ = positive echelon holding cost associated with operation i,
h = sum of all echelon holding costs, i.e., holding cost for all end items,
b = positive delay cost per time unit.

The delay costs at node 0 are obtained as $\mathrm{b}(\mathrm{t}_1 + \tau_1 - \mathrm{t_d})^+$, where $\mathrm{x}^+ = \max(\mathrm{x}, 0)$. If there are several end items, the delay cost is based on the maximum delay. There are no delay costs associated with other nodes. However, delays at other nodes may affect the delay at node 0. Consider node i, which is the starting point for operation i. Note first that we must have $\mathrm{t_i} \geq \max(\mathrm{t_j} + \tau_\mathrm{j}, \mathrm{t_k} + \tau_\mathrm{k})$. After starting operation i, the echelon holding cost $\mathrm{e_i}$ is incurred until the final delivery,

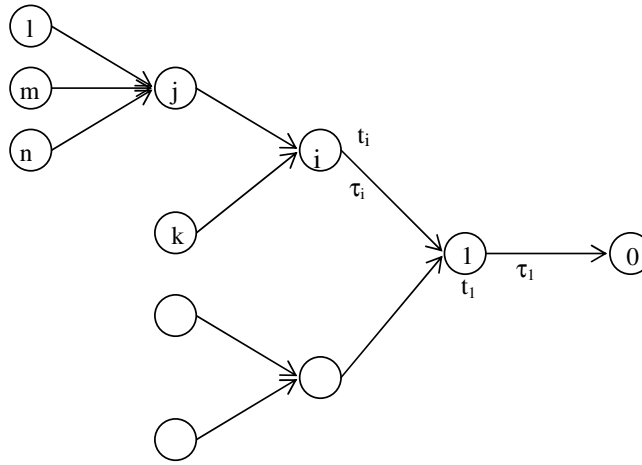


Fig. 1. Assembly network

which will take place at the requested delivery time $\mathrm{t_d}$, or later in case of a delay. However, we disregard the holding costs during the operations, because they are not affected by when the operations are carried out. It is assumed that raw material can be obtained instantaneously from an outside supplier. This means that initial operations like l, m, and n in Figure 1 can start at any time.

3 Single-stage system

We shall derive an approximate solution by successively applying the exact solution for a single-stage system. Consider therefore first the system in Figure 2.

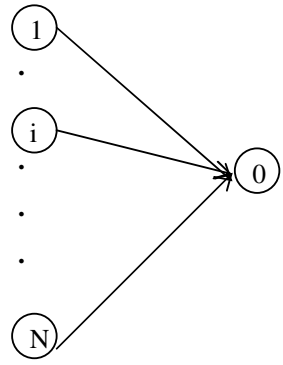


Fig. 2. Single-stage system

We can express the delay d as

$$d = \max_{1 \le \mathrm{i} \le \mathrm{N}} \left\{ (\mathrm{t_i} + \tau_\mathrm{i} - \mathrm{t_d})^+ \right\}. \tag{1}$$

The average costs C can then be expressed as

$$C = \sum_{i=1}^{N} e_i E(t_d + d - t_i - \tau_i) + bE(d)$$
$$= \sum_{i=1}^{N} e_i(t_d - t_i - E(\tau_i)) + \left(\sum_{i=1}^{N} e_i + b\right) E(d). \quad (2)$$

We shall optimize C with respect to the starting times $t_i (i = 1, 2, ..., N)$. This problem was solved by Yano (1987a), who also showed that C is a convex function of the starting times for $N = 2$. It is easy to see that C is convex in the starting times also for larger values of N. Consider the right-hand side of (2). The first term is linear so we only need to show that E(d) is convex. Because the operation times are independent it is enough to demonstrate that d in (1) is convex in the starting times for given operation times. Note that x^+ is convex. Let $0 \le \alpha \le 1$, and t' and t'' be two set-ups of starting times. We have

$$\max_{1\le i\le N} \left\{(\alpha t_i' + (1-\alpha)t_i'' + \tau_i - t_d)^+\right\}$$
$$\le \max_{1\le i\le N} \left\{\alpha(t_i' + \tau_i - t_d)^+ + (1-\alpha)(t_i'' + \tau_i - t_d)^+\right\}$$
$$\le \alpha \max_{1\le i\le N} \left\{(t_i' + \tau_i - t_d)^+\right\} + (1-\alpha) \max_{1\le i\le N} \left\{(t_i'' + \tau_i - t_d)^+\right\}. \quad (3)$$

Furthermore, it is evident that $C \to \infty$ as $t_i \to \infty$ or $t_i \to -\infty$. It follows that C has a unique minimum for finite values of the starting times t_i. See also e.g., Hopp and Spearman (1993) and Song et al. (2000).

Let $G(x)$ be the cumulative distribution function of the delay d. We have

$$G(x) = P(d \le x) = \prod_{i=1}^{N} F_i(x + t_d - t_i). \quad (4)$$

We can now express C as

$$C = \sum_{i=1}^{N} e_i(t_d - t_i - E(\tau_i)) + \left(\sum_{i=1}^{N} e_i + b\right) \int_0^\infty \left(1 - \prod_{i=1}^{N} F_i(x + t_d - t_i)\right) dx. \quad (5)$$

Consequently we get the partial derivative of C with respect to t_i as

$$\frac{\partial C}{\partial t_i} = -e_i + \left(\sum_{i=1}^{N} e_i + b\right) \int_0^\infty f_i(x + t_d - t_i) \prod_{j\ne i} F_j(x + t_d - t_j) dx. \quad (6)$$

When evaluating $\partial C/\partial t_i$ numerically we need to carry out a numerical integration.

Assume now first that there are no constraints on the variables t_i. We will then obtain the minimum as the unique solution of $\partial C/\partial t_i = 0$. Note that for $N = 1$ the problem degenerates to the familiar Newsboy problem, and we obtain the optimum from the condition

$$F_1(t_d - t_1) = \frac{b}{b + e_1}. \quad (7)$$

Consider the general case again. For given values of the other t_j it is easy to find the t_i giving $\partial C/\partial t_i = 0$. Due to the convexity $\partial C/\partial t_i$ is increasing so we can

apply a simple search procedure. If we start with some initial values of the starting times, e.g., $t_i = t_d - E(\tau_i)$, and carry out optimizations with respect to one t_i at a time the costs are nonincreasing and we will ultimately reach the optimal solution due to the convexity.

Assume now that we have lower bounds for the starting times t_i.

$t_{i0} =$ lower bound for t_i.

We can then obtain the optimal solution in almost the same way. We start with feasible values of t_i, e.g., $t_i = \max(t_d - E(\tau_i), t_{i0})$. Then we optimize with respect to one t_i at a time. Consider a local optimization of t_i. We first check whether $t_i = t_{i0}$ is optimal. This is the case if $\partial C/\partial t_i > 0$ for $t_i = t_{i0}$. Otherwise we determine the t_i giving $\partial C/\partial t_i = 0$ as before. Again the costs are nonincreasing and we will ultimately reach the optimal solution due to the convexity.

4 Approximate procedure for a multi-stage system

Consider now a general assembly system. We shall describe our approximate planning procedure. Let

$P(i) =$ set of operations that are immediate predecessors of node i.

Consider first the immediate predecessors of the final node, i.e., $P(0)$. In Figure 1 we have a single predecessor, but in a more general case we may have multiple predecessors like in Figure 2. Assume that there are N operations in $P(0)$. Assume also that the operations that must precede these operations are finished, i.e., the N operations in $P(0)$ are ready to start. Consider first all operations, which do not belong to $P(0)$. The holding costs associated with these operations before time t_d cannot be affected by the starting times for the operations in P(0). So we disregard these holding costs in the first step. There are also holding costs associated with these operations during a possible delay, which should be included because they are affected by the starting times for the operations in $P(0)$. For the operations in $P(0)$ both the holding costs from their respective starting times to t_d, and during a possible delay are affected by the starting times for the operations in $P(0)$. There are also delay costs associated with a delay. This leads to the following cost expression for the operations in $P(0)$.

$$C(0) = \sum_{j=1}^{N} e_j(t_d - t_j - E(\tau_j)) + (h+b)\int_0^\infty \left(1 - \prod_{j=1}^{N} F_j(x + t_d - t_j)\right) dx. \quad (8)$$

Note that the only difference compared to (5) is that h includes also holding costs during a possible delay for operations preceding $P(0)$. Using the algorithm in Section 3 (without lower bounds for the starting times) we optimize (8) in our first step and get the corresponding starting times t_i^* for the operations in $P(0)$.

Assume then that i is one of the operations in $P(0)$, and consider its immediate predecessors, i.e., the operations in $P(i)$. We shall now consider the single stage system consisting of these operations and thereby interpret t_i^* as a requested delivery time. For an operation in $P(i)$ the starting time will affect the holding costs before

t_i^*. Let us also consider a delay cost $\hat{b}$ that replaces $h + b$ in (8). The resulting problem is to minimize

$$C(i) = \sum_{j=1}^{N} e_j(t_i^* - t_j - E(\tau_j)) + \hat{b} \int_0^\infty (1 - \prod_{j=1}^{N} F_j(x + t_i^* - t_j))dx. \tag{9}$$

Note that although we, for simplicity, are using a similar notation in (8) and (9), the considered operations are not the same. So the number of operations N, the holding costs e_j, and the distribution functions F_j are normally different.

It remains to determine $\hat{b}$. If there is a delay, this delay will affect the starting times of the operations in $P(0)$. This will increase the costs $C(0)$ in (8). Consider some given starting times for the operations in $P(i)$ and let the corresponding stochastic delay relative to t_i^* be δ. A reasonable approximate delay cost is

$$\begin{aligned}\hat{b} &= E_{\delta>0}\left\{\frac{dC(0)}{d\delta}(t_1^* + \delta, t_2^* + \delta, ..., t_N^* + \delta)\right\} \\ &= E_\delta\left\{\frac{dC(0)}{d\delta}(t_1^* + \delta, t_2^* + \delta, ..., t_N^* + \delta)\right\} / \Pr(\delta > 0).\end{aligned} \tag{10}$$

The second equality in (10) follows because $dC(0)/d\delta = 0$ for $\delta = 0$ due to the optimality of t_i^*. Because of our assumption concerning the distributions of the operation times we know that $\Pr(\delta > 0) > 0$. In (10) it is implicitly assumed that all operations in $P(0)$ are started δ time units later compared to the optimal solution, i.e., not only operation i. This is a reasonable assumption and will also simplify the computations. Using that

$$\frac{d}{d\delta}\prod_{j=1}^{N} F_j(x + t_d - t_j - \delta) = -\frac{d}{dx}\prod_{j=1}^{N} F_j(x + t_d - t_j - \delta), \tag{11}$$

we get from (10)

$$\begin{aligned}&\frac{dC(0)}{d\delta}(t_1^* + \delta, t_2^* + \delta, ..., t_N^* + \delta) \\ &= -\sum_{j=1}^{N} e_j + (h + b)\int_0^\infty \left(\frac{d}{dx}\prod_{j=1}^{N} F_j(x + t_d - t_j - \delta)\right) dx \\ &= -\sum_{j=1}^{N} e_j + (h + b)\left(1 - \prod_{j=1}^{N} F_j(t_d - t_j - \delta)\right)\end{aligned} \tag{12}$$

so it is relatively easy to evaluate $\hat{b}$ according to (10). Recall that we get the distribution of the delay from (4).

Because $\hat{b}$ in (10) depends on the starting times of the operations in $P(i)$, it is unknown. It is, however, still easy to determine an optimal solution corresponding to the delay cost (10). Note first that the $\hat{b}$ from (10) is bounded from below by 0 and from above by $h + b - \sum_{j=1}^{N} e_j$. The upper bound is easy to see from (12). It is also clear that the upper bound will lead to a finite optimal solution of (9). (Recall that h is the sum of all echelon holding costs.) Assume now that we start with a

certain $\hat{b}_{in}$ from the considered interval. There are now two possibilities. If $\hat{b}_{\mathrm{in}}$ is sufficiently small there is no finite optimum of (9). The resulting $\hat{b}_{\mathrm{out}}$ from (10) will approach the upper bound. If, on the other hand, there is a finite solution we know that $\hat{b}_{\mathrm{out}}$ is between the lower and upper bounds. Clearly, $\hat{b}_{\mathrm{out}}$ is a continuous function of $\hat{b}_{\mathrm{in}}$. Consequently, it follows from Brouwer's fixed point theorem that there exists a fixed point $\hat{b}_{\mathrm{out}} = \hat{b}_{\mathrm{in}}$, i.e., a solution corresponding to the delay cost (10). It is easy to find such a fixed point by a one-dimensional search.

Remark. Normally $\hat{b}_{\mathrm{out}}$ is a decreasing function of $\hat{b}_{\mathrm{in}}$. In that case it is very easy to find the unique fixed point.

We can then handle the predecessors of the operations in $P(i)$ in the same way, etc. Let j be one of the operations in $P(i)$. When dealing with $P(j)$ we let t_j^* be the requested delivery time for the single-stage system. In (10) we are using $C(i)$ instead of $C(0)$. A difference here is that the upper bound for $\hat{b}$ will not necessarily lead to a finite solution. This means in that case that the delay δ will approach infinity and, as a consequence, also the operations in $P(0)$ will be delayed. Consequently it is reasonable to use the costs in the preceding step, i.e., in this case $C(0)$ instead of $C(i)$. If necessary we can go one step further, and so on. This will always work because $C(0)$ will provide an upper bound leading to a finite optimal solution.

We will end up with starting times t_i^* for all operations and delay costs for all single-stage systems. When implementing the solution we will stick to the obtained starting times as long as they are possible to follow. However, delays may enforce changes. Consider, for example, operation j in Figure 1. Assume that operations l, m, and n are finished at some time $t_{j0} > t_j^*$. We then derive a new solution for the operations in $P(i)$. (The delay cost at node i is not changed.) In the solution we apply the constraint $t_j \geq t_{j0}$. Starting times that have already been implemented are regarded as given. If operation k has not yet started, its starting time may increase but cannot decrease. To see this consider (6) and note that $\partial C/\partial t_i$ is nonincreasing if some other t_j is increasing, i.e., t_i^* is nondecreasing.

Let us summarize our approximate procedure:

1. Optimize $C(0)$ as given by (8). Let K be the number of stages. Set $k = -1$.
2. Set $k = k + 1$.
3. For all operations i with k succeeding operations, optimize (9) for the operations of $P(i)$ under the constraint (10) with the cost function $C(i)$. If $k > 0$ it may occur that no finite optimum exists. If this is the case use the cost function for the successor of i. If necessary go to the successor of the successor, etc.
4. If $k < K - 1$, then goto 2.
5. Implement the solution. In case of a delay reoptimize free starting times without changing the delay cost.

5 Numerical results

To evaluate the suggested approximate procedure we used two sets of sample problems.

Problem set 1

Our first problem set concerns the two-stage network in Figure 3.

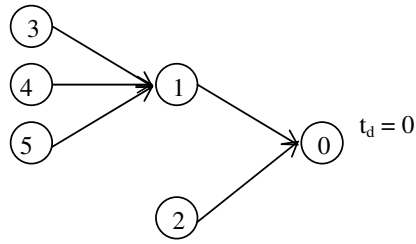


Fig. 3. Network for Problem set 1

The requested delivery time is $\mathrm{t_d} = 0$. All operation times have the same distribution. We denote this stochastic operation time by τ. This means that it is relatively easy to determine also the exact solution for comparison. By symmetry operations 3, 4, and 5 should have the same starting time $\mathrm{t_s} = \mathrm{t_3} = \mathrm{t_4} = \mathrm{t_5}$. As before consider first the single-stage network to the right. Let $\mathrm{t_1^*} = \mathrm{t_2^*}$ be the optimal solution of (8). Given $\mathrm{t_s}$ there is a certain stochastic delay d at node 1 relative to $\mathrm{t_1^*}$. If $\mathrm{d} = 0$ it is optimal to apply $\mathrm{t_1^*}$ and $\mathrm{t_2^*}$. If $\mathrm{d} > 0$ it is optimal to use the solution obtained with the constraint $\mathrm{t_1} \geq \mathrm{t_{10}} = \mathrm{t_1^*} + \mathrm{d}$. Recall that this leads to $\mathrm{t_2} \geq \mathrm{t_2^*}$. Let $\mathrm{c(d)}$ be the corresponding expected costs for the single-stage network according to (8). We obtain the total costs as

$$\mathrm{C(t_s)} = \sum_{\mathrm{i}=3}^{5} \mathrm{e_i(t_d - t_s - E(\tau))} + \mathrm{E_d\left\{c(d)\right\}}. \tag{13}$$

This is the case both for the exact and the approximate solution. The only difference between the exact and approximate solution is the determination of $\mathrm{t_s}$. In the approximate procedure we use the procedure described in Section 4. In the optimal solution we optimize (13) with respect to $\mathrm{t_s}$.

All echelon holding costs are kept equal, $\mathrm{e_i} = 1$, while we considered three different delay costs $\mathrm{b} = 5$, 25, and 50. Furthermore the expected operation time $\mathrm{E}(\tau) = 1$ in all considered cases. Two different types of distributions for the operation time were considered. For each distribution we considered the standard deviations $\sigma = 0.2, 0.5$, and 1. Both distributions are constructed as $\alpha + (1-\alpha)\mathrm{X}$, where α is a constant between 0 and 1 and X is a stochastic variable with its mean equal to 1. Distribution 1 is obtained by letting X have an exponential distribution with mean (and standard deviation) equal to 1. Distribution 2 is similarly obtained

Table 1. Optimal parameters and costs for Problem set 1

Distribution	Stand. dev. σ	Delay cost b	Optimal policy t_s	Optimal policy Costs	Approx. policy t_s	Approx. policy Costs	Cost increase %
1	0.2	5	−2.16	2.06	−2.22	2.08	1.0
1	0.2	25	−2.50	3.48	−2.58	3.52	1.1
1	0.2	50	−2.66	4.21	−2.75	4.26	1.1
1	0.5	5	−2.39	5.15	−2.54	5.19	0.8
1	0.5	25	−3.25	8.71	−3.45	8.81	1.1
1	0.5	50	−3.66	10.52	−3.88	10.64	1.1
1	1	5	−2.79	10.30	−3.09	10.40	1.0
1	1	25	−4.50	17.43	−4.90	17.62	1.1
1	1	50	−5.32	21.04	−5.75	21.27	1.1
2	0.2	5	−2.09	2.07	−2.17	2.10	1.4
2	0.2	25	−2.49	3.74	−2.57	3.77	0.8
2	0.2	50	−2.69	4.64	−2.78	4.68	0.9
2	0.5	5	−2.30	5.20	−2.43	5.23	0.6
2	0.5	25	−3.25	9.38	−3.43	9.45	0.7
2	0.5	50	−3.72	11.56	−3.95	11.68	1.0
2	1	5	−2.61	10.40	−2.86	10.47	0.7
2	1	25	−4.49	18.75	−4.87	18.90	0.8
2	1	50	−5.43	23.12	−5.90	23.35	1.0

by letting X be the square of a normally distributed random variable with mean 0 and standard deviation 1. In other words, X has a χ^2-distribution with one degree of freedom. This means that X has mean 1 and standard deviation $\sqrt{2}$. In both cases the mean is equal to 1 for any value of α, and by adjusting α we can obtain the considered standard deviations.

The results are shown in Table 1. The relative cost increase when using our approximate technique is quite small in all 18 cases. The maximum error is 1.4 %. The relative errors are fairly insensitive to the distribution type, the delay cost, and the standard deviation of the operation times. The approximate method results for Problem set 1 always in an earlier starting time t_s for the initial operations, i.e., the needed safety times are overestimated.

Problem set 2

Our second problem set concerns a more complicated network with three stages (see Fig. 4). Operations 1, 2, 3, 5, and 7 have the same stochastic operation time τ, while the times of operations 4 and 6 are 2τ. All times are independent. The time τ has the same distribution as distribution 1 in Problem set 1 with $E(\tau) = 1$ and the standard deviations $\sigma = 0.2$, 0.5, and 1. Furthermore, as for Problem set 1 all

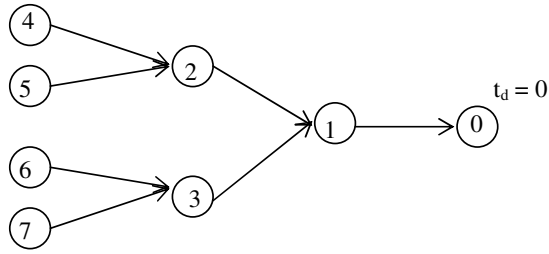


Fig. 4. Network for Problem set 2

Table 2. Optimal parameters and costs for Problem set 2

Stand. dev. σ	Delay cost b	Optimum by simulation t_{s1}	t_{s2}	t_m	Costs	Approximate policy t_{s1}	t_{s2}	t_m	Costs	Cost increase %
0.2	5	−4.3	−3.1	−2.3	3.68	−4.20	−3.20	−2.26	3.94	7.1
0.2	25	−5.0	−3.5	−2.5	6.46	−4.57	−3.53	−2.53	6.86	6.2
0.2	50	−5.3	−3.6	−2.7	7.82	−4.76	−3.70	−2.68	8.69	11.1
0.5	5	−4.6	−3.2	−2.6	9.18	−4.49	−3.49	−2.66	9.86	7.4
0.5	25	−5.8	−3.9	−3.4	16.16	−5.42	−4.32	−3.33	17.57	8.7
0.5	50	−7.1	−4.4	−3.8	19.12	−5.90	−4.75	−3.70	22.29	16.6
1	5	−5.2	−3.3	−3.3	17.89	−4.98	−3.99	−3.31	20.04	12.0
1	25	−8.3	−4.7	−4.6	31.85	−6.84	−5.64	−4.67	34.41	8.0
1	50	−9.9	−5.5	−5.4	38.79	−7.80	−6.49	−5.41	43.69	12.6

echelon holding costs are equal, $e_i = 1$, and we consider the delay costs $b = 5, 25$, and 50. Table 2 shows the results.

The approximate policy in Table 2 is obtained as described in Section 4. The costs for the approximate policy are obtained by simulation. The standard deviation is less than 0.02. The time t_{s1} is the starting time of the longer operations 4 and 6, and t_{s2} is the starting time of operations 5 and 7. The time t_m is the starting time for operations 2 and 3 if both operations are ready to start.

The optimum by simulation is obtained by a combination of simulation and analytical techniques. We omit the details. All times, both starting times and stochastic times were for simplicity rounded to multiples of 0.1. This does not affect the results much. More important is that in the determination of the optimal policy we carried out minimizations over several simulated costs. This means that the costs are somewhat underestimated. By considering the costs for the starting times suggested by the approximate policy we could, however, conclude that the error is not very significant. The cost increase of the approximate policy in Table 2 may be overestimated by $1-2$ % but hardly more.

We must conclude that the approximation errors for Problem set 2 are much larger than for Problem set 1. The average cost increase is nearly 10 %. Note that the intermediate time t_m is very accurate also in Table 2. We note that our

approximation underestimates the needed safety times for the long operations 4 and 6 while it overestimates the safety times for the shorter operations 5 and 7. To explain the large difference in errors, consider first the network for Problem set 1 in Figure 3. The delay at node 1 will determine the starting times t_1 and t_2 because we can initiate operation 2 at any time. A linear delay cost is then reasonable. Consider then the network for Problem set 2 in Figure 4 and the delay at node 2. A small delay may then not be that serious because there may any way be a delay at node 3. A long delay may, however, cause a long delay also for operation 3. This indicates that our linear delay cost may be less appropriate in this case. It also explains why the longer more stochastic operations are starting so early in the optimal solution, i.e., we wish to avoid long delays. A general, not very surprising, conclusion can be that our approximation works better for networks with a single "critical path".

6 Conclusions

We have considered the problem of minimizing expected delay and holding costs for a complex assembly network where the operation times are independent stochastic variables. An approximate decomposition technique for solving the problem has been suggested. The technique means repeated applications of the solution of a simpler single-stage problem. The approximate method has been evaluated for two problem sets. The results are very good for the first set of two-stage problems and the relative cost increase due to the approximation is only about 1 %. For the second set of three-stage problems the errors are about 10 % and cannot be disregarded. Although the numerical results show some promise, further research is needed for evaluation of the applicability of the suggested technique in more general settings. Because it is difficult to derive exact solutions for problems of realistic size it may be most fruitful to compare different heuristics for larger problems.

References

Buzacott JA, Shanthikumar JG (1994) Safety stock versus safety time in MRP controlled production systems. Management Science 40: 1678–1689

Chen F (2001) Market segmentation, advanced demand information, and supply chain performance. Manufacturing & Service Operations Management 3: 53–67

Chu C, Proth JM, Xie X (1993) Supply management in assembly systems. Naval Research Logistics 40: 933–950

Clark AJ, Scarf H (1960) Optimal policies for a multi-echelon inventory problem. Management Science 6: 475–490

Gong L, de Kok T, Ding J (1994) Optimal leadtimes planning in a serial production system. Management Science 40: 629–632

Hariharan R, Zipkin P (1995) Customer-order information, leadtimes, and inventories. Management Science 41: 1599–1607

Hopp W, Spearman M (1993) Setting safety leadtimes for purchased components in assembly systems. IIE Transactions 25: 2–11

Karaesmen F, Buzacott JA, Dallery Y (2002) Integrating advance order information in make-to-stock production systems. IIE Transactions 34: 649–662

Kumar A (1989) Component inventory costs in an assembly problem with uncertain supplier lead-times. IIE Transactions 21: 112–121

Shore H (1995) Setting safety lead-times for purchased components in assembly systems: a general solution procedure. IIE Transactions 27: 634–637

Song J-S, Yano CA, Lerssrisuriya P (2000) Contract assembly: Dealing with combined supply lead time and demand quantity uncertainty. Manufacturing & Service Operations Management 2: 287–296

Song J-S, Zipkin P (2003) Supply chain operations: Assemble-to-order systems, Ch. 11. In: Graves SC, De Kok T (eds) Handbooks in operations research and management science, Vol 11. Supply chain management: design, coordination and operation. Elsevier, Amsterdam

Yano C (1987a) Stochastic leadtimes in two-level assembly systems. IIE Transactions 19: 371–378

Yano C (1987b) Setting planned leadtimes in serial production systems with tardiness costs. Management Science 33: 95–106

Yano C (1987c) Stochastic leadtimes in two-level distribution-type networks. Naval Research Logistics 34: 831–843

A multiperiod stochastic production planning and sourcing problem with service level constraints*

Işıl Yıldırım[1], **Barış Tan**[2], **and Fikri Karaesmen**[1]

[1] Department of Industrial Engineering, Koç University, Rumeli Feneri Yolu, Sariyer, Istanbul, Turkey (e-mail: isil.yildirim@insead.edu.tr;fkaraesmen@ku.edu.tr)
[2] Graduate School of Business, Koç University, Rumeli Feneri Yolu, Sariyer, Istanbul, Turkey (e-mail: btan@ku.edu.tr)

Abstract. We study a stochastic multiperiod production planning and sourcing problem of a manufacturer with a number of plants and/or subcontractors. Each source, i.e. each plant and subcontractor, has a different production cost, capacity, and lead time. The manufacturer has to meet the demand for different products according to the service level requirements set by its customers. The demand for each product in each period is random. We present a methodology that a manufacturer can utilize to make its production and sourcing decisions, i.e., to decide how much to produce, when to produce, where to produce, how much inventory to carry, etc. This methodology is based on a mathematical programming approach. The randomness in demand and related probabilistic service level constraints are integrated in a deterministic mathematical program by adding a number of additional linear constraints. Using a rolling horizon approach that solves the deterministic equivalent problem based on the available data at each time period yields an approximate solution to the original dynamic problem. We show that this approach yields the same result as the base stock policy for a single plant with stationary demand. For a system with dual sources, we show that the results obtained from solving the deterministic equivalent model on a rolling horizon gives similar results to a threshold subcontracting policy.

Keywords: Stochastic production planning – Service level constraints – Subcontracting

* The authors are grateful to Yves Dallery for his ideas, comments and suggestions on the earlier versions of this paper.
Correspondence to: F. Karaesmen

1 Introduction and motivation

In this study, we consider a manufacturer that supplies products to a retailer. The manufacturer has a number of production sources that are either its own plants or its subcontractors. Each source has a different production cost, capacity, and lead time. The demand for each product in each period is random. The manufacturer has to meet the demand for multiple products taking into account the service level requirements set by the retailer.

In the production planning and the sourcing problem, the manufacturer's decision variables are how much to produce, when to produce, where to produce, and how much inventory to carry in each period. The objective is to minimize its total production and inventory carrying costs during the planning horizon subject to the service level requirements and other possible constraints.

This problem is motivated by the problems faced by suppliers of lean retailers in the textile-apparel-retail channel (Abernathy et al., 1999). Namely, adoption of lean retailing practices force suppliers of lean retailers to adopt new strategies to respond quickly to changing demand effectively. Using subcontractors emerge as a viable alternative to increase production capacity temporarily when it is needed. Additional cost of subcontracting can be justified by lowering inventories and improving the service. However, deciding on where to produce and how much to produce is a challenging task especially when the demand is volatile. A qualitative discussion of this problem can be found in Abernathy et al. (2000). Figure 1 below depicts the system which motivates this study.

We propose a solution methodology that is based on solving a deterministic mathematical problem at each time period on a rolling horizon basis. Randomness in the problem that comes from uncertain demand and service level constraints are integrated in a deterministic mathematical program by adding a number of additional linear constraints similar to the approach proposed by Bitran and Yanasse (1984). We propose using this approach to address the more relevant but also more

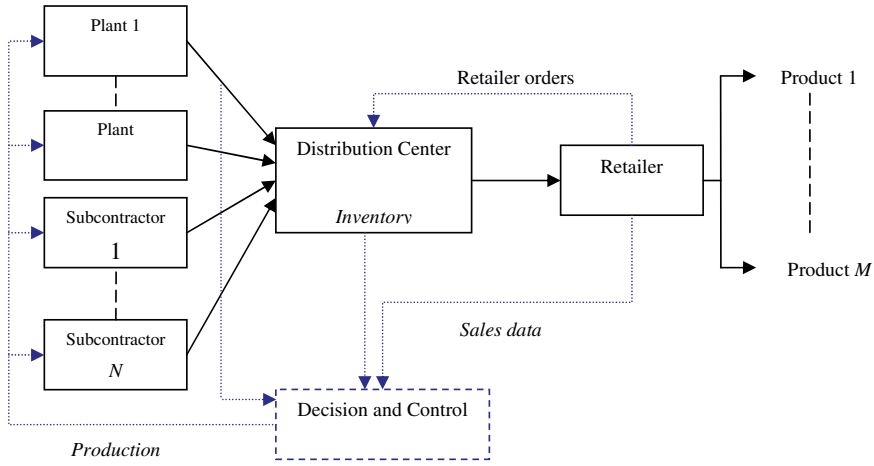


Fig. 1. A manufacturer with multiple plants that sells multiple products to a retailer

difficult dynamic problem where decisions can be updated over time. Since the equivalent deterministic problem is a well-structured mathematical programming problem, the proposed methodology can easily be integrated with the Advanced Planning and Optimization tools, such as the products of i2, Manugistics, etc., that are commonly used in practice.

The organization of the remaining parts of the paper is as follows: In Section 2, we review the literature on mathematical-programming-based stochastic production planning methodologies. The particular stochastic production planning and sourcing problem we investigate is introduced in Section 3. Section 4 presents the proposed solution methodology that is based on solving the deterministic equivalent problem at each time step on a rolling horizon basis. The performance of the rolling horizon approach is evaluated by considering a number of special cases in Section 5. Finally, conclusions are presented in Section 6.

2 Literature review

The classical deterministic production planning problem, its mathematical programming formulations and solution methodologies have received a lot of attention for many years (see Hax and Candea, 1984 for a number of well-known models). In this section, we only review the literature directly related to mathematical programming based approaches for stochastic production planning problems.

Bitran and Yanasse (1984) deal with a similar stochastic production planning problem with a service level requirement. They provide non-sequential (static) and deterministic equivalent formulations of the model and propose error bounds between the exact solution and the proposed approach. Their main focus is on the solution of the static problem, i.e., the solution at time zero for the whole planning horizon.

Bitran, Haas and Matsudo (1986) present a model that is motivated by a case in the consumer electronics and textile and apparel industry. In this model, the stochastic problem is transformed into a deterministic one by replacing the random demand with their average values. Then, the solution of the transformed problem provides answers to the questions of what to produce and when to produce. The complete solution is obtained by determining how much to produce from a newsboy-type formulation based on the solution of the deterministic problem.

Feiring and Sastri (1989) focus on production smoothing plans with rolling horizon strategies and confidence levels for the demand, which are set by the production planners. The probabilistic constraints in the demand-driven scheduling model are revised by Bayesian procedures and are transformed into deterministic constraints by inverse transformations of normally distributed demand.

Zäpfel (1996) claims that MRP II systems can be inadequate for the solution of production planning problems with uncertain demand because of the insufficiently supported aggregation/disaggregation process. The paper then proposes a procedure to generate an aggregate plan and a consistent disaggregate plan for the Master Production Schedule.

Kelle, Clendenen and Dardeau (1994) extend the economic lot scheduling problem for the single-machine, multi-product case with random demands. Their ob-

jective is to find the optimal length of production cycles that minimizes the sum of set-up costs and inventory holding costs per unit of time and satisfies the demand of products at the required service levels.

Clay and Grossman (1997) focus on a two-stage fixed-recourse problem with stochastic Right-Hand-Side terms and stochastic cost coefficients and propose a sensitivity-based successive disaggregation algorithm.

Sox and Muckstadt (1996) present a model for the finite-horizon, discrete-time, capacitated production planning problem with random demand for multiple products. The proposed model includes backorder cost in the objective function rather than enforcing service level constraints. A subgradient optimization algorithm is developed for the solution of the proposed model by using Lagrangian relaxation and some computational results are provided.

Beyer and Ward (2000) report a production and inventory problem of Hewlett-Packard's Network Server Division. The authors propose a method to incorporate the uncertainties in demand in an Advanced Planning System utilized by Hewlett-Packard.

Albritton, Shapiro and Spearman (2000) study a production planning problem with random demand and limited information and propose a simulation based optimization method. Qui and Burch (1997) study a hierarchical production planning and scheduling problem motivated by the fibre industry and propose an optimization model that uses logic of expert systems.

Van Delft and Vial (2003) consider multiperiod supply chain contracts with options. In order to analyze the contracts, they propose a methodology to formulate the deterministic equivalent problem from the base deterministic model and from an event tree representation of the stochastic process and solve the stochastic linear program by discretizing demand under the backlog assumption.

For the textile-apparel-retail problem discussed in Abernathy et al. (2000), a simulation model has also been developed (Yang et al., 1997). Then a simulation-based optimization technique that is referred as ordinal optimization, has been used to determine the parameters of a production and inventory control policy that gives a good-enough solution approximately (Yang et al., 1997; Lee, 1997). However, one needs to set a specific production and inventory control policy in the simulation. In addition to the difficulty of setting a plausible policy in a complicated case, as the number of sources and products increase, the number of parameters to be optimized also increases. As a result, finding an approximate solution requires a considerable time.

Simplified versions of the sourcing problem studied in this paper have been investigated in the past by using stochastic optimal control (Bradley, 2002; Tan and Gershwin, 2004; Tan, 2001). Bradley (2002) considers a system with a producer and a subcontractor and discrete flow of goods. In an M/M/1 setting without the service level requirements, he proves that the optimal control policy structure is a dual-base stock policy. In this policy when the number of customers in the queue reaches a certain level, then new incoming customers are sent to the subcontractor. When there are no customers waiting in the queue, then the producer continues production until a certain threshold is reached.

In Tan (2001) and Tan and Gershwin (2004), a producer with a single subcontractor is formulated with continuous flow of goods without the service level requirements. They also show that a threshold-type policy is optimal to decide when and how to use a subcontractor. In the threshold policy, the subcontractor is used when the inventory or the backlog is below a certain threshold level.

Our paper uses the idea of incorporating randomness in a deterministic mathematical program that is used in many of the above studies in different formats. We utilize the approach proposed by Bitran and Yanasee (1984) that shows the equivalence for the static problem. In contrast to this study where the main objective is determining error bounds for the optimal cost in the non-sequential case, our main focus is generating a production and sourcing plan, i.e. determining the values of the decision variables in the sequential (dynamic) problem where sourcing decisions are made (or updated) dynamically over time. We also compare the approximate solution of the dynamic problem with certain benchmark policies. Since the exact optimal solution of the dynamic problem is not known, we use two different benchmarks. It is proven that for a single source with lead time, the proposed approach yields the same production policy as the optimal base stock policy. For a dual-source, e.g. a producer with a subcontractor, a threshold-type subcontracting policy suggested by Bradley (2002), Tan (2001), Tan and Gershwin (2004) is utilized as a benchmark. After adopting the threshold policy to a more generalized case with lead time and service-level requirements, it is observed that the proposed approach yields very similar results to the threshold-based benchmark in the numerical examples considered.

3 Stochastic multiperiod sourcing problem with service level constraints

Assume that there is a single product and N different production sources (plants and subcontractors). The demand for this specific product at time t, d_t is random. The main decision variables are the production quantities at each production source at time t, $X_{i,t}, i = 1, \ldots, N$. The inventory level at the end of time period t is denoted by I_t. The number of periods in the planning horizon is T. The inventory holding cost per unit per unit time is h_t and the production cost at production source i at time t is $c_{i,t}$.

Constraints on the performance (related to backorders) of the system are imposed by requiring service levels. The frequently used *Type* 1 *Service Level* is defined to be the fraction of periods in which there is no stock out. It can be viewed as the plant's no-stock-out frequency. This service level measures whether or not a backorder occurs but is not concerned with the size of the backorder. In this study, we consider a *Modified Type* 1 *Service Level* requirement. The *Modified Type* 1 *Service Level* forces the probability of having no stock out to be greater than or equal to a service level requirement in each period. The service level requirement in period t is denoted by α_t.

The Stochastic Production Planning and Sourcing Problem (SP) is defined as:

$$Z^*(SP) = \text{Min}E\left[\sum_{t=1}^{T}\left(h_t(I_t)^+ + \sum_{i=1}^{N} c_{i,t}X_{i,t}\right)\right]$$

subject to

$$I_t = I_{t-1} + \sum_{i=1}^{N} X_{i,t} - d_t,\ t = 1, ..., T; \tag{1}$$

$$P\{I_t \geq 0\} \geq \alpha_t,\ t = 1, ..., T. \tag{2}$$

$$X_{i,t} \geq 0,\ i = 1, ..., N\ t = 1, ..., T. \tag{3}$$

where $(I_t)^+ = \text{Max}\{0, I_t\},\ t = 1, ..., T$.

The objective of the problem is to minimize the total expected cost, which is the expected value of the sum of the inventory holding and production costs in the planning horizon. The first constraint set defines the inventory balance equations for each time period. The next constraint imposes the service level requirement for each period. Finally, the last constraint states that the production quantities cannot be negative.

This formulation can easily be extended to multiple products and production sources with lead times. Moreover different service level definitions can also be considered by following the same approach.

4 An approximate solution procedure based on a rolling horizon procedure

The solution of the above problem at time 0 for the planning horizon [0, T] is referred as the static solution. The static solution is obtained by using the available information about the distribution of demand in the future periods and the initial inventory. A policy that sets (or updates) the future production quantities $X_{i,t}$ at time t based on the information available at that time, e.g., demand realizations, demand distributions in the future periods, and current inventory levels, is referred to as the dynamic solution.

In theory, the optimal policy which determines production quantities based on actual state information may be obtained by solving the stochastic dynamic program associated with this problem. In practice, however, there are several problems with the stochastic dynamic programming solution. First, the well-known curse of dimensionality makes numerical solutions challenging even for relatively small problems. Second, it is difficult to integrate constraints on the trajectory of the underlying stochastic processes such as service level requirements in inventory models. Therefore, we propose a rolling-horizon approach that is based on solving the static problem at each time period based on the available information. This, however, requires solving the static problem repeatedly which requires a transformation explained below.

4.1 Deterministic equivalent formulation for the static solution

Although obtaining the optimal dynamic solution is, in general, not tractable, the static solution can relatively easily be obtained by using deterministic mathematical programming as suggested by Bitran and Yanasse (1984).

In particular, Bitran and Yanasse show that the (Modified Type 1) service level constraint can be transformed into a deterministic equivalent constraint by specifying certain minimum cumulative production quantities that depend on the service level requirements.

To summarize this approach, let l_t denote the (deterministic equivalent) minimum cumulative production quantity in period t which is calculated by solving the probabilistic inequality:

$$P\left\{\sum_{\tau=1}^{t} d_\tau \leq l_t\right\} = \alpha_t, \quad t = 1, ..., T \text{ for } l_t(t = 1, ..., T)$$

that yields

$$l_t = F_t^{-1}(\alpha_t), \quad t = 1, ..., T$$

where $F_t(.)$ is the cumulative distribution function of the random sum $\sum_{\tau=1}^{t} d_\tau$. Then the probabilistic constraint $P\{I_t \geq 0\} \geq \alpha_t,\ t = 1, ..., T$ can be expressed equivalently by:

$$\sum_{\tau=1}^{t}\sum_{i=1}^{N} X_{i,\tau} + I_0 \geq l_t, \quad t = 1, ..., T \tag{4}$$

Now, the deterministic equivalent problem with service level constraints that has been mentioned in the previous sections can be modeled as below (Bitran and Yanasse, 1984):

Deterministic Equivalent Problem (DEP):

$$Z^*(DEP) = \text{Min} \sum_{t=1}^{T}\left(h_t(I_0 + \sum_{\tau=1}^{t}\sum_{i=1}^{N} X_{i,\tau}) + \sum_{i=1}^{N} c_{i,t}X_{i,t}\right)$$

subject to

$$\sum_{\tau=1}^{t}\sum_{i=1}^{N} X_{i,\tau} + I_0 \geq l_t, \quad t = 1, ..., T \tag{5}$$

$$X_{i,t} \geq 0, \quad i = 1, ..., N \quad t = 1, ..., T. \tag{6}$$

The optimal decision variable values in DEP are the same as the ones in the solution of SP at time 0.

The static solution is obtained by transforming the stochastic problem into a deterministic one and then solving the resulting mathematical program. The rolling horizon approach repeats this procedure by using the available information at each time period until time T.

5 Performance of the rolling horizon solution

It is known that the rolling-horizon approach yields good results for a number of dynamic optimization problems. In some special cases, the rolling horizon method may even yield the optimal solution. In this section, we evaluate the performance of the proposed method by comparing it to certain benchmark policies in two commonly encountered special cases in production planning.

5.1 A single source problem with stationary demand

We start with the special case of a single production source. When there is only one source, the objective function includes only the holding cost (since the expected total production costs must equal the total expected demand over the planning horizon). In this case, we use the base stock policy as the benchmark policy. The base stock policy is widely known and utilized in many applications. In addition, it is known to be optimal in a number of related inventory problems. It, therefore, constitutes a natural benchmark for comparison. The base stock policy has a single parameter which is a reorder level and a base lot size of one unit. It aims to maintain a pre-specified target inventory level. Under this policy, the sequence of events is as follows: the system starts with a pre-specified base stock level in the finished goods inventory. The arrival of the customer demand triggers the consumption of an end-item from the inventory and issuing of a replenishment order to the production facility. Using this policy, an order is placed (or the manufacturing facility operates) if and only if the inventory level drops below the base stock level. The comparison of these two models is performed for two cases with and without a lead time.

5.1.1 Single source without lead time

In this first scenario, there is a single product to be produced by a single production facility. It is assumed that the demand of this specific product stays stationary over the planning horizon. We propose that solving the deterministic equivalent model with modified service level constraints on a rolling horizon basis is equivalent to operating the system under the base stock policy. The next proposition establishes this equivalence:

Proposition 1. When the production facility has no lead time and the demand is stationary, using a base stock policy is equivalent to solving the deterministic equivalent model with service level constraints on a rolling horizon basis (either Modified Type 1 or Modified Type 2) in the following way: assume that the base stock level in the base stock policy equals $I_0(BS) = S_1$ and the initial inventory level in the deterministic equivalent problem equals $I_0(DEP) = l_1$. If $S_1 = l_1$, then the equivalent base stock policy gives the same total expected cost value, yields the same production plan and results in the same service level with the deterministic equivalent model with modified service level constraints solved on a rolling horizon basis.

Since this case is a special case of the next one with lead time, the proof of Proposition 1 is not given here but reported in (Yıldırım, 2004).

Corollary 1. The optimal base stock level is equal to l_1. Equivalently, the base stock level $S_1 = l_1$ ensures that the resulting production plan satisfies the required service levels.

Proof. If the initial inventory level is set to be $S_1 = l_1$, the resulting production plan is the same with that of the base stock policy which starts with a base stock level of $S_1 = l_1$. Although the base stock policy does not guarantee the assurance of the service levels, since we know that the deterministic equivalent model satisfies the

required service levels and the two policies are equivalent, we can say that the base stock level $S_1 = l_1$ ensures that the resulting production plan satisfies the required service levels. Note that $S_1 = l_1$ must be optimal because decreasing the base stock level from l_1 leads to an infeasible solution and increasing it above l_1 would lead to higher average inventory costs and therefore cannot be optimal. □

Even though a formal proof is lacking, it is highly likely that the base stock policy (with a stationary base stock level) is optimal for the single-plant single-product problem in an infinite horizon setting. Theorem 1 and Corollary 1 establish that for this problem, the rolling horizon approach yields the same solutions as the optimal base stock policy leading us to conclude that the rolling horizon procedure performs optimally in this case.

5.1.2 Single source with lead time

The deterministic equivalent model with service level constraints (DEP) can be extended to a case in which the production facility has a production lead time. Assume that there is a production lead time of *LT* periods and the initial scheduled receipts are denoted by $SR_t, t = 1, ..., LT$. Then, the problem can be modeled in the following way:

Deterministic Equivalent Production Planning Problem including Lead Time (DEPLT):

$$Z^*(\text{DEPLT}) = \text{Min} \sum_{t=1}^{LT} \left(h_t(I_0 + \sum_{\tau=1}^{t} SR_\tau) \right) + \sum_{t=LT+1}^{T} \left(h_t(I_0 + \sum_{\tau=1}^{LT} SR_\tau + \sum_{\tau=LT+1}^{t} \sum_{i=1}^{N} X_{i,\tau-LT}) \right)$$

subject to

$$\sum_{\tau=LT+1}^{t} X_{\tau-LT} + \sum_{\tau=1}^{LT} SR_\tau + I_0 \geq l_t, \quad t = (LT+1), ..., T; \tag{7}$$

$$X_t \geq 0, \quad t = 1, ..., T. \tag{8}$$

Our main result is as follows:

Proposition 2. When the production facility has a non-negative lead time *LT*, the demand is stationary and there are no scheduled receipts initially, using a base stock policy is equivalent to solving the deterministic equivalent model with service level constraints on a rolling horizon basis in the following manner: assume that the base stock level in the base stock policy including lead time equals $I_0(\text{BSLT}) = S_2$ and the initial inventory level in the deterministic equivalent model including lead time equals $I_0(\text{DEPLT}) = l_{LT+1}$. If $S_2 = l_{LT+1}$, then the equivalent base stock policy gives the same total expected cost value, yields the same production plan and results in the same service level with the deterministic equivalent model with service level constraints solved on a rolling horizon basis.

Proof. The proof of Proposition 2 is given in the Appendix. □

5.2 A dual source problem with stationary demand

Since the optimal solution of our dynamic problem is not known, a plausible benchmark is used to evaluate the performance of the proposed approach. We propose a threshold subcontracting model suggested in a number of studies in the literature (Bradley, 2002; Tan, 2001; Tan and Gershwin, 2004). Although the threshold policy is only shown to be optimal under specific assumptions including zero lead time, stationary demand, no service level requirements, etc., we think that it is a reasonable benchmark policy for our problem.

5.2.1 A threshold subcontracting policy

Now we explain the operation of the threshold policy for our benchmark case. We consider a dual source system with an in-house production facility and a subcontractor. We assume that the in-house facility has a capacity of C but the subcontractor has an infinite capacity. There is a lead time of one period. That is, production quantities scheduled at time t become available at time $t + 1$.

The threshold policy is characterized by two threshold levels S and Z. The in-house production facility operates when the inventory level is below S. That is, it starts producing when the inventory level drops below the target level S and stops producing when the inventory level again reaches S. The subcontractor is used when the inventory level decreases to a threshold level of Z.

When the inventory level is below S, but is still above Z, the in-house facility produces to cover the shortfall with respect to S. If there is not sufficient production capacity to cover the whole shortfall, the in-house facility operates at full capacity and the portion of demand that cannot be satisfied is backlogged for the next period.

Let $X_{1,t}$ and $X_{2,t}$ denote the production amounts of the in-house facility and the subcontractor in period t respectively. Then, the production amounts of each production facility in each time period can be determined for the threshold subcontracting model in the following way:

$$X_{1,t} = \text{Min}\{S - Z, S - I_{t-1}, C\}, \quad t = 1, ..., T; \tag{9}$$

$$X_{2,t} = \text{Max}\{0, Z - I_{t-1}\}, \quad t = 1, ..., T. \tag{10}$$

The following figure shows the evolution of $X_{1,t}$, $X_{2,t}$ and I_t under this policy for a Poisson arrival of demand with rate 10 and $S = 15, Z = 7$, and $C = 8$.

5.2.2 Comparison of the performance of the threshold policy and the rolling horizon approach

The deterministic equivalent model for this case is solved for a rolling horizon of 10 periods repeatedly throughout a planning horizon of 1000 periods. 5000 sample demand streams are generated and the realized inventory levels are integrated in the model accordingly. The production plans and the realized cost values between periods 451 and 550 are observed. All cost values are calculated on a per period basis.

The optimal values of the threshold values S and Z are determined by using a direct simulation-based numerical search. It is assumed that there are 1000 periods

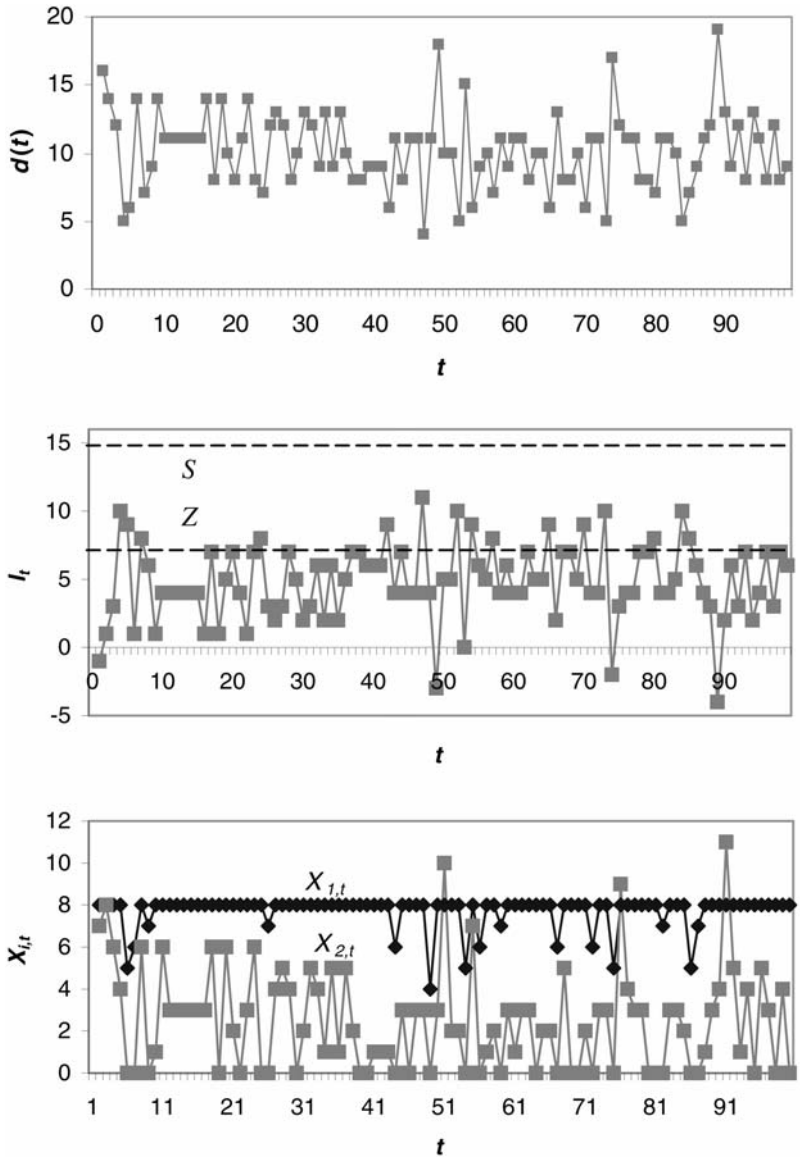


Fig. 2. Sample realization of d_t, $X_{1,t}$, $X_{2,t}$ and I_t under the threshold policy $S = 15$, $Z = 7, C = 8$

in the planning horizon and the same 5000 sample demand streams are utilized. The service level requirement is relaxed with the one-sided 95% confidence interval of the simulation result. That is whenever upper confidence level of the observed service level reaches the desired one, this case is accepted as satisfying the service level requirement. The underlying reasoning behind making this modification in service levels is that, the sample size we utilize might not be sufficient enough to make the realized service level equal exactly to the required one. Among the base

Table 1. The possible scenarios for which comparisons are made

Subcontracting cost	Holding cost	In-house production capacity	(Subcontracting cost)/(in-house prod. cost)	(Holding cost)/(in-house prod. cost)	(In-house prod. capacity)/(mean demand)
4	16	8	1	4	0.8
4	16	12	1	4	1.2
4	16	20	1	4	2
6	1	8	1.5	0.25	0.8
6	1	12	1.5	0.25	1.2
6	1	20	1.5	0.25	2
6	4	8	1.5	1	0.8
6	4	12	1.5	1	1.2
6	4	20	1.5	1	2

stock and threshold levels that satisfy the relevant service level requirements, the model aims to find the one with minimum total cost. The calculations are performed for periods between 451 and 550.

For the numerical examples reported below, the order arrivals are governed by a Poisson process with rate 10 products per period. The production cost is assumed to be \$4 per product for the in-house facility. The initial inventory level of the specific product is set to be zero. The service level requirement is set to be 95%.

The comparison between the deterministic equivalent model and the threshold subcontracting model is performed for nine combinations of subcontracting cost to in-house production cost, holding cost to in-house production cost and capacity to mean demand ratios. The combinations of subcontracting costs, holding costs and the in-house production capacities and therefore, the combinations of relevant subcontracting cost to in-house production cost, holding cost to in-house production cost and capacity to mean demand ratios for which the comparisons are made can be observed in Table 1. For each of the problem settings, the base stock and threshold levels observed in the threshold subcontracting model are reported in Table 2.

Note that, in some of the cases, the base stock and threshold pairs are observed to be the same. The reasoning behind this is, these pairs lead to the same average inventory levels and minimum cost values in these settings.

While comparing the two models, total expected cost, average production cost, average inventory holding cost values and the assignment of production to the plants (in percentages) are the key elements we focus on. Table 3 summarizes the total expected cost values of the deterministic equivalent model (DEM) and the threshold subcontracting model (TSM) for the nine different scenarios for each modified service level type.

The below tables display that the deterministic equivalent model gives very close solutions when compared with the threshold subcontracting model for both types of the modified levels. The deterministic equivalent model results in total expected cost values equal to or a little bit larger than those of the threshold subcontracting

Table 2. Base stock and threshold levels observed in each scenario

Subcontracting cost	Holding cost	In-house production capacity	Critical levels	
			Base stock	Threshold
4	16	8	15	7
4	16	12	15	3
4	16	20	15	$-\infty$
6	1	8	17	7
6	1	12	16	0
6	1	20	15	$-\infty$
6	4	8	15	7
6	4	12	15	3
6	4	20	15	$-\infty$

Table 3. The comparison of total expected cost values observed in each scenario

Subcontracting cost	Holding cost	In-house production capacity	Total expected cost		
			DEM	TSM	Percentage difference
4	16	8	121.66	121.66	0.00
4	16	12	121.66	121.66	0.00
4	16	20	121.66	121.62	0.03
6	1	8	49.97	49.89	0.16
6	1	12	46.16	45.65	1.12
6	1	20	45.10	45.10	0.02
6	4	8	65.33	65.33	0.00
6	4	12	61.47	61.47	0.00
6	4	20	60.42	60.40	0.03

model. For our set of numerical experiments, the deterministic equivalent model gives close results to the threshold subcontracting model when the service level requirement is of Modified Type 1.

Tables 4 and 5 display the comparison of average production and holding cost values. As can be seen, the deterministic equivalent model gives similar results to the threshold subcontracting model.

Table 6 summarizes the percentage of production assigned to the in-house production facility for both the deterministic equivalent model and the threshold subcontracting model. The results suggest that the production assignments of the deterministic model follow a similar pattern with the benchmark chosen.

Based on these figures, we can conclude that the proposed deterministic equivalent model solved on a rolling horizon basis performs as well as the threshold subcontracting model solved on a simulation-based optimization technique for the

Table 4. The comparison of average production cost values observed in each scenario

Subcontracting cost	Holding cost	In-house production capacity	Average production cost		
			DEM	TSM	Percentage difference
4	16	8	39.99	39.99	0.00
4	16	12	39.99	39.99	0.00
4	16	20	39.99	39.99	0.00
6	1	8	44.06	44.36	−0.68
6	1	12	41.05	40.24	2.03
6	1	20	40.00	39.97	0.06
6	4	8	44.91	44.91	0.00
6	4	12	41.05	41.05	0.00
6	4	20	40.00	39.99	0.01

Table 5. The comparison of average holding cost values observed in each scenario

Subcontracting cost	Holding cost	In-house production capacity	Average holding cost		
			DEM	TSM	Percentage difference
4	16	8	81.67	81.67	0.00
4	16	12	81.67	81.67	0.00
4	16	20	81.67	81.63	0.05
6	1	8	5.92	5.53	6.91
6	1	12	5.10	5.41	−5.67
6	1	20	5.10	5.10	0.05
6	4	8	20.42	20.42	0.00
6	4	12	20.42	20.42	0.00
6	4	20	20.42	20.41	0.05

Modified Type 1 service level. The total expected cost values of deterministic equivalent models for all nine different cases are equal to or a little bit larger than those of the threshold subcontracting model. However, we cannot reach the same conclusion for the average production and holding cost values. The deterministic equivalent model performs either worse for some cases or better for some other cases when the comparison is based on average production or holding cost values. However, the sum of these two terms, the total expected cost, is equal to a little bit larger than that of the threshold subcontracting model. Moreover, the proportion of production assigned to the in-house facility in the deterministic equivalent model resembles that in the simulation based threshold subcontracting model.

It is worth mentioning that the sample size utilized in the above numerical comparisons, 5000, might not be large enough to satisfy the service level requirements in each time period that the modified service level definitions necessitate.

Table 6. The percentage of production assignments to the in-house production facility observed in each scenario

Subcontracting cost	Holding cost	In-house production capacity	% In-house production	
			Base stock	Threshold
4	16	8	75.45	75.40
4	16	12	94.73	94.70
4	16	20	99.97	100.00
6	1	8	79.76	78.17
6	1	12	94.73	98.78
6	1	20	99.97	100.00
6	4	8	75.45	75.40
6	4	12	94.73	94.70
6	4	20	99.97	100.00

The coefficient of variation in the realized service level values might be larger than expected. To handle this problematic issue, we introduced one-sided confidence intervals. Although the threshold subcontracting model constitutes a lower bound in terms of total expected cost values for our set of numerical examples, it can not be generalized from our examples that the deterministic equivalent model always gives solutions worse than those of the threshold subcontracting model. Nevertheless, the proposed approach seems to give extremely promising results in this particular case as well.

6 Conclusions

In many practical situations, mathematical models of production planning/outsourcing problems have to deal with the randomness in demand. We present a systematic approach that enables the randomness in demand and the desired service levels to be incorporated in a mathematical programming framework.

We show that solving the deterministic equivalent problem on a rolling-horizon basis gives similar results to the performance of the benchmarks. Although the threshold-type policies are conceptually quite intuitive, it is very challenging to determine the optimal threshold levels by using simulation. The proposed algorithm is easier to implement and optimize by using available solvers.

This study can be extended in a number of ways. The same approach can be used to derive results for different service level definitions. Yıldırım (2004) reports preliminary results for Type 2 and Modified Type 2 service levels. The formulation of the multi-product case is also straightforward.

The effects of demand variability, production cost, and the lead time on the production and sourcing plans need further investigation. Since the optimal solution to the general problem is not known for the dynamic case, investigation of the static

case or a stylized model can yield insights regarding the interaction of demand variability, cost, and the lead time.

Appendix

Proof of Proposition 2

We use induction to show that

i. If the inventory levels at the beginning of the first period are equal, $I_0(\text{BSLT}) = I_0(\text{DEPLT}) = l_{LT+1}$, then production quantities in the first period and the inventory at the end of first period for both policies become equal, i.e. $X_1(\text{BSLT}) = X_1(\text{DEPLT}) = 0$ and $I_1(\text{BSLT}) = I_1(\text{DEPLT}) = l_{LT+1} - d_1$;

ii. If the inventory levels at the end of period t_1 such that $t_1 \leq LT$ are equal, $I_{t_1}(\text{BSLT}) = I_{t_1}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=1}^{t_1} d_\tau$, then the production quantities in period $(t_1 + 1)$ and the inventory levels at the end of period $(t_1 + 1)$ for both policies become equal; i.e. $X_{t_1+1}(\text{BSLT}) = X_{t_1+1}(\text{DEPLT}) = d_{t_1}$ and $I_{t_1+1}(\text{BSLT}) = I_{t_1+1}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=1}^{t_1+1} d_\tau$.
and

iii. If the inventory levels at the end of period (LT+1) are equal, $I_{LT+1}(\text{BSLT}) = I_{LT+1}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=1}^{LT+1} d_\tau$, then production quantities in period $(LT+2)$ and the inventory levels at the end of period $(LT+2)$ for both policies become equal, i.e. $X_{LT+2}(\text{BSLT}) = X_{LT+2}(\text{DEPLT}) = d_{LT+1}$ and $I_{LT+2}(\text{BSLT}) = I_{LT+2}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=2}^{LT+2} d_\tau$;

iv. If the inventory levels at the end of period t_2 such that $t_2 \geq LT$ are equal, $I_{t_2}(\text{BSLT}) = I_{t_2}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=t_2-LT}^{t_2} d_\tau$, then the production quantities in period (t_2+1) and the inventory levels at the end of period (t_2+1) for both policies become equal; i.e. $X_{t_2+1}(\text{BSLT}) = X_{t_2+1}(\text{DEPLT}) = d_{t_2}$ and $I_{t_2+1}(\text{BSLT}) = I_{t_2+1}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=t_2+1-LT}^{t_2+1} d_\tau$.

Assume that the initial inventory levels are equal such that $I_0(\text{BSLT}) = S_2, I_0(\text{DEPLT}) = l_{LT+1}$ and $S_2 = l_{LT+1}$. In the base stock policy, each demand observed is produced in the next period; therefore there is no production in the first period, $X_1(\text{BSLT}) = 0$. In the deterministic equivalent approach, the production quantity in the first period is determined according to the constraint $X_1(\text{DEPLT}) + \sum_{\tau=1}^{LT} SR_\tau(\text{DEPLT}) + I_0(\text{DEPLT}) = X_1(\text{DEPLT}) + 0 + l_{LT+1} \geq l_{LT+1}$ and therefore, $X_1(\text{DEPLT}) \geq 0$. Since the problem is of minimization type, the production quantity in the first period equals zero, i.e. $X_1(\text{DEPLT}) = 0$. Next, a customer demand of d_1 arrives. The end of period inventory for the base stock policy becomes $I_1(\text{BSLT}) = I_0(\text{BSLT}) + SR_1(\text{BSLT}) - d_1 = S_2 + 0 - d_1 = S_2 - d_1$ and the end of period inventory for the deterministic equivalent approach becomes $I_1(\text{DEPLT}) = I_0(\text{DEPLT}) + SR_1(\text{DEPLT}) - d_1 = l_{LT+1} + 0 - d_1 = l_{LT+1} - d_1$. Since we know that $S_2 = l_{LT+1}$, $I_1(\text{BSLT}) = I_1(\text{DEPLT})$.

In the second period, the base stock policy produces the demand of the first period, i.e. $X_2(\text{BSLT}) = d_1$. At the beginning of the second period, the deterministic equivalent model is rerun since it is solved on a rolling horizon basis.

The demand is assumed to be stationary over the planning horizon. Although solving the model on a rolling horizon basis throughout the planning horizon requires integration of the minimum cumulative production quantites for the number of periods in the rolling horizon into the model, only the minimum cumulative production quantity of period $(LT+1), l_{LT+1}$, is fully utilized. The production quantity of the deterministic equivalent model in the second period is determined by $X_2(\text{DEPLT}) + \sum_{\tau=2}^{LT+1} SR_\tau(\text{DEPLT}) + I_1(\text{DEPLT}) = X_2(\text{DEPLT}) + X_1(\text{DEPLT}) + I_1(\text{DEPLT}) = X_2(\text{DEPLT}) + 0 + l_{LT+1} - d_1 \geq l_{LT+1}$; therefore, $X_2(\text{DEPLT}) \geq d_1$. In order to minimize the production costs, the production quantity in the second period equals the demand of the first period, i.e. $X_2(\text{DEPLT}) = d_1$. After the arrival of a customer demand of d_2, the end of period inventory for the base stock policy becomes $I_2(\text{BSLT}) = I_1(\text{BSLT}) + SR_2(\text{BSLT}) - d_2 = S_2 - d_1 - d_2$ and the end of period inventory for the deterministic equivalent approach becomes $I_2(\text{DEPLT}) = I_1(\text{DEPLT}) + SR_2(\text{DEPLT}) - d_2 = l_{LT+1} - d_1 - d_2$. Since $S_2 = l_{LT+1}$, we can say that $I_2(\text{BSLT}) = I_2(\text{DEPLT})$.

Since demand during lead time cannot be satisfied no sooner than $(LT+1)$ periods of time, the inventory levels at the end of any period t_1 such that $t_1 \leq (LT-1)$ can be written as $I_{t_1}(\text{BSLT}) = S_2 - \sum_{\tau=1}^{t_1} d_\tau$, $I_{t_1}(\text{DEP}) = l_{LT+1} - \sum_{\tau=1}^{t_1} d_\tau$ and $S_2 = l_{LT+1}$. In period (t_1+1), the base stock policy produces $X_{t_1+1}(\text{BSLT}) = d_{t_1}$. In the deterministic equivalent approach, the production quantity is determined by the constraint $X_{t_1+1}(\text{DEPLT}) + \sum_{\tau=t_1+1}^{t_1+LT} SR_\tau(\text{DEPLT}) + I_{t_1}(\text{DEPLT}) = X_{t_1+1}(\text{DEPLT}) + \sum_{\tau=1}^{t_1} X_\tau(\text{DEPLT}) + I_{t_1}(\text{DEPLT}) = X_{t_1+1}(\text{DEPLT}) + \sum_{\tau=1}^{t_1-1} d_\tau + l_{LT+1} - \sum_{\tau=1}^{t_1} d_\tau \geq l_{LT+1}$; therefore, $X_{t_1+1}(\text{DEPLT}) \geq d_{t_1}$. Since the problem is of minimization type, $X_{t_1+1}(\text{DEPLT}) = d_{t_1}$. Then, a customer demand of d_{t_1+1} is observed. The end of period inventory for the base stock policy becomes $I_{t_1+1}(\text{BSLT}) = I_{t_1}(\text{BSLT}) + SR_{t_1+1}(\text{BSLT}) - d_{t_1+1} = S_2 - \sum_{\tau=1}^{t_1} d_\tau - d_{t_1+1} = S_2 - \sum_{\tau=1}^{t_1+1} d_\tau$ and the end of period inventory for the deterministic equivalent approach becomes $I_{t_1+1}(\text{DEPLT}) = I_{t_1}(\text{DEPLT}) + SR_{t_1+1}(\text{DEPLT}) - d_{t_1+1} = l_{LT+1} - \sum_{\tau=1}^{t_1} d_\tau - d_{t_1+1} = l_{LT+1} - \sum_{\tau=1}^{t_1+1} d_\tau$. Since $S_2 = l_{LT+1}$, $I_{t_1+1}(\text{BSLT}) = I_{t_1+1}(\text{DEPLT})$.

Similarly, d_{LT+1} is produced by the base stock policy in period $(LT+1)$, i.e. $X_{LT+1} = d_{LT+1}$. The constraint $X_{LT+1}(\text{DEPLT}) + \sum_{\tau=LT+1}^{2LT} SR_\tau(\text{DEPLT}) + I_{LT}(\text{DEPLT}) = X_{LT+1}(\text{DEPLT}) + \sum_{\tau=1}^{LT} X_\tau + I_{LT}(\text{DEPLT}) = X_{LT+1}(\text{DEPLT}) + \sum_{\tau=1}^{LT-1} d_\tau + l_{LT+1} - \sum_{\tau=1}^{LT} d_\tau \geq l_{LT+1}$; i.e. $X_{LT+1}(\text{DEPLT}) \geq d_{LT}$ determines the production quantity of the deterministic equivalent model in period $(LT+1)$. Then, $X_{LT+1}(\text{DEPLT}) = d_{LT}$. Next, a customer demand of d_{LT+1} arrives. The end of period inventory for the base stock policy becomes $I_{LT+1}(\text{BSLT}) = I_{LT}(\text{BSLT}) + SR_{LT+1}(\text{BSLT}) - d_{LT+1} = S_2 - \sum_{\tau=1}^{LT} d_\tau + X_1(\text{BSLT}) - d_{LT+1} = S_2 - \sum_{\tau=1}^{LT} d_\tau + 0 - d_{LT+1} = S_2 - \sum_{\tau=1}^{LT+1} d_\tau$ and the end of period inventory for the deterministic equivalent approach becomes $I_{LT+1}(\text{DEPLT}) = I_{LT}(\text{DEPLT}) + SR_{LT+1}(\text{DEPLT}) - d_{LT+1} = l_{LT+1} - \sum_{\tau=1}^{LT} d_\tau + X_1(\text{DEPLT}) - d_{LT+1} = l_{LT+1} - \sum_{\tau=1}^{LT} d_\tau + 0 - d_{LT+1} = l_{LT+1} - \sum_{\tau=1}^{LT+1} d_\tau$. Since $S_2 = l_{LT+1}$, $I_{LT+1}(\text{BSLT}) = I_{LT+1}(\text{DEPLT})$.

In period $(LT+2)$, the base stock policy produces $X_{LT+1}(\text{BSLT}) = d_{LT+2}$. For the deterministic equivalent approach, we know that $X_{LT+2}(\text{DEPLT}) + \sum_{\tau=LT+2}^{2LT+1} SR_\tau(\text{DEPLT}) I_{LT+1}(\text{DEPLT}) = X_{LT+2}(\text{DEPLT}) + \sum_{\tau=2}^{LT+1} X_\tau + I_{LT+1}(\text{DEPLT}) = X_{LT+2}(\text{DEPLT}) + \sum_{\tau=1}^{LT} d_\tau + l_{LT+1} - \sum_{\tau=1}^{LT+1} d_\tau \geq l_{LT+1}$; i.e. $X_{LT+2}(\text{DEPLT}) \geq d_{LT+1}$ and then, $X_{LT+2}(\text{DEPLT}) = d_{LT+1}$. After the arrival of d_{LT+2}, the following end of period inventory levels are observed $I_{LT+2}(\text{BSLT}) = I_{LT+1}(\text{BSLT}) + SR_{LT+2}(\text{BSLT}) - d_{LT+2} = S_2 - \sum_{\tau=1}^{LT+1} d_\tau + X_2(\text{BSLT}) - d_{LT+2} = S_2 - \sum_{\tau=1}^{LT+1} d_\tau + d_1 - d_{LT+2} = S_2 - \sum_{\tau=2}^{LT+2} d_\tau$ and $I_{LT+2}(\text{DEPLT}) = I_{LT+1}(\text{DEPLT}) + SR_{LT+2}(\text{DEPLT}) - d_{LT+2} = l_{LT+1} - \sum_{\tau=1}^{LT+1} d_\tau + X_2(\text{DEPLT}) - d_{LT+2} = l_{LT+1} - \sum_{\tau=1}^{LT+1} d_\tau + d_1 - d_{LT+2} = l_{LT+1} - \sum_{\tau=2}^{LT+2} d_\tau$. Since we know that $S_2 = l_{LT+1}$, $I_{LT+2}(\text{BSLT}) = I_{LT+2}(\text{DEPLT})$.

Now assume that at the end of any period t_2 such that $t_2 \geq (LT+1)$, $I_{t_2}(\text{BSLT}) = S_2 - \sum_{\tau=t_2-LT}^{t_2} d_\tau$, $I_{t_2}(\text{DEPLT}) = l_{LT+1} - \sum_{\tau=t_2-LT}^{t_2} d_\tau$ and $S_2 = l_{LT+1}$. In period (t_2+1), $X_{t_2+1}(\text{BSLT}) = d_{t_2}$ and $X_{t_2+1}(\text{DEPLT})$ is determined by the constraint $X_{t_2+1}(\text{DEPLT}) + \sum_{\tau=t_2+1}^{t_2+LT} SR_\tau(\text{DEPLT}) + I_{t_2}(\text{DEPLT}) = X_{t_2+1}(\text{DEPLT}) + \sum_{\tau=1}^{t_2} X_\tau + I_{t_2}(\text{DEPLT}) = X_{t_2+1}(\text{DEPLT}) + \sum_{\tau=1}^{t_2-1} d_\tau + l_{LT+1} - \sum_{\tau=1}^{t_2} d_\tau \geq l_{LT+1}$; $X_{t_2+1}(\text{DEPLT}) \geq d_{t_2}$ and since the model is of minimization type $X_{t_2+1}(\text{DEPLT}) = d_{t_2}$. Next, a customer demand of d_{t_2+1} arrives. The end of period inventory levels for both policies become $I_{t_2+1}(\text{BSLT}) = I_{t_2}(\text{BSLT}) + SR_{t_2+1}(\text{BSLT}) - d_{t_2+1} = S_2 - \sum_{\tau=t_2-LT}^{t_2} d_\tau + X_{t_2+1-LT}(\text{BSLT}) - d_{t_2+1} = S_2 - \sum_{\tau=t_2-LT}^{t_2} d_\tau + d_{t_2-LT} - d_{t_2+1} = S_2 - \sum_{\tau=t_2+1-LT}^{t_2+1} d_\tau$ and $I_{t_2+1}(\text{DEPLT}) = I_{t_2}(\text{DEPLT}) + SR_{t_2+1}(\text{DEPLT}) - d_{t_2+1} = S_2 - \sum_{\tau=t_2-LT}^{t_2} d_\tau + X_{t_2+1-LT}(\text{DEPLT}) - d_{t_2+1} = S_2 - \sum_{\tau=t_2-LT}^{t_2} d_\tau + d_{t_2-LT} - d_{t_2+1} = S_2 - \sum_{\tau=t_2+1-LT}^{t_2+1} d_\tau$. Since we know that $S_2 = l_{LT+1}$, $I_{t_2+1}(\text{BSLT}) = I_{t_2+1}(\text{DEPLT})$. This proves our proposition.

References

Abernathy FH, Dunlop JT, Hammond JH, Weil D (1999) A stitch in time. Oxford University Press, New York

Abernathy FH, Dunlop JT, Hammond JH, Weil D (2000) Control your inventory in a world of lean retailing. Harvard Business Review Nov-Dec: 169–176

Albritton M, Shapiro A, Spearman M (2000) Finite capacity production planning with random demand and limited information. Stochastic Programming E-Print Series

Beyer RD, Ward J (2000) Network server supply chain at HP: a case study. HP Labs Tech Report 2000-84

Bitran GR, Yanasse HH (1984) Deterministic approximations to stochastic production problems. Operations Research 32: 999–1018

Bitran GR, Haas EA, Maatsudo (1986) Production planning of style goods with high setup costs and forecast revisions. Operations Research 34(2): 226–236

Bradley JR (2002) Optimal control of a dual service rate M/M/1 production-inventory model. European Journal of Operational Research (2002) (forthcoming)

Candea D, Hax AC (1984) Production and inventory management. Prentice-Hall, New Jersey

Clay RL, Grossman IE (1997) A disaggregation algorithm for the optimization of stochastic planning models. Computers and Chemical Engineering 21(7): 751–774

Feiring BR, Sastri T (1989) A demand-driven method for scheduling optimal smooth production levels. Annals of Operations Research 17: 199–216

Kelle P, Clendenen G, Dardeau P (1994) Economic lot scheduling heuristic for random demands. International Journal of Production Economics 35: 337–342

Lee LH (1997) Ordinal optimization and its application in apparel manufacturing systems. Ph.D. Thesis, Harvard University, Cambridge, MA

Qiu MM, Burch EE (1997) Hierarchical production planning and scheduling in a multiproduct, multi-machine environment. International Journal of Production Research 35(11): 3023–3042

Sox CR, Muckstadt JA (1996) Multi-item, multi-period production planning with uncertain demand. IIE Transactions 28: 891–900

Tan B, Gershwin SB (2004) Production and subcontracting strategies for manufacturers with limited capacity and volatile demand. Annals of Operations Research (Special volume on Stochastic Models of Production/Inventory Systems) 125: 205–232

Tan B (2002) Managing manufacturing risks by using capacity options. Journal of the Operational Research Society 53(2): 232–242

Van Delft C, Vial J-PH (2003) A practical implementation of stochastic programming: an application to the evaluation of option contracts in supply chains. Automatica (to appear)

Yang MS, Lee LH, Ho YC (1997) On stochastic optimization and its applications to manufacturing. Lectures in Applied Mathematics 33: 317–331

Yıldırım I (2004) Stochastic production planning and sourcing problems with service level constraints. M.S. Thesis, Koç University, Industrial Engineering, Istanbul, Turkey

Zäpfel G (1996) Production planning in the case of uncertain individual demand extension for an MRP II concept. International Journal of Production Economics 46–47: 153–164