Analysis of Multi-Class Queueing Systems with Finite Buffers and Setup Times Using Decomposition Methods

Georg N. Krieg and Heinrich Kuhn

Faculty of Business Administration and Economics
Catholic University of Eichstätt-Ingolstadt
85049 Ingolstadt, Germany
e-mail: georg.krieg@ku-eichstaett.de
e-mail: heinrich.kuhn@ku-eichstaett.de

Abstract:
In this paper, we consider two variants of a multi-class queueing system with a single server, finite buffers, and setup times. In the first variant, a setup occurs every time the server turns to a different class of customers, even if the queue of the scanned customer class is empty. In the second variant, the server skips empty queues and a setup is performed only at queues that contain at least one customer. We propose decomposition methods for the analysis of the steady-state behavior of both system variants. We also show that multi-product kanban systems with setup times and lost sales are possible applications of the discussed models and we give numerical results that indicate the accuracy of the proposed approximation methods.

Keywords:
Queueing systems, Multiple customer classes, Finite buffers, Setups, Polling systems, Decomposition method, Kanban

1 Introduction

Queueing systems with multiple classes of customers are difficult to evaluate analytically, especially if buffers are finite and setup times require other service disciplines than first-come-first-served (FCFS). Recently, Anupindi and Tayur (1998), Olsen (1999), and Kim and Van Oyen (2000) investigated scheduling methods for multi-class manufacturing systems with setup times. Simulation was used in the first two studies to evaluate system alternatives, the authors of the latter employed numerical analysis of a Markov decision process model. As a consequence, they were restricted to the analysis of small systems with mainly two and three customer classes.

In this paper, we propose decomposition methods for two variants of a multi-class queueing system with a single server, finite buffers, and setup times. The performance measures generated by both approximation procedures are sufficiently accurate, and the algorithms converge fast and reliably. In the first system variant, setup, or change-over, times occur every time the server turns to a different class of customers. If the queue is empty upon completion of the setup, the server turns to the next class of customers so that it never idles. In the second variant, the server skips empty queues and a setup is performed only at queues that contain at least one customer. If the system is completely empty, the server idles at the current queue conserving the present setup. In both variants, queues are served exhaustively and they are considered in a fixed sequence that is repeated cyclically. Customers are rejected if all positions are occupied in the queue of their class.

Decomposition methods are frequently used to analyze multi-stage manufacturing systems with one product class (see, for example, Bonvik et al. 2000; Dallery and Frein 1993; Dallery and Gershwin 1992; Di Mascolo et al. 1996 and the textbooks by Buzacott and Shanthikumar 1993, Gershwin 1994, and Papadopoulos et al. 1993). The basic principle is to decompose a system with K stages into K single-stage subsystems. We transfer this idea to multi-class systems and decompose a system with r customer classes into r single-class subsystems. This has been done before, for example, by Altiok and Shiue (1994), Federgruen and Katalan (1994), Jung and Un (1994), Kofman (1993), and Takagi (1991). What may be new in our approach is that we model the subsystems as continuous-time Markov chains that are then solved numerically. By doing this, we gain considerable flexibility for the design of the subsystems.

Systems like the one considered here are called *polling systems* in the computer science literature (see, for example, the survey by Takagi 1990). Both variants are special and extremely difficult examples of this class. Variant A is a finite-capacity polling system with cyclic-exhaustive service and a continuously roving server, variant B is a finite-capacity polling system with cyclic-exhaustive service, zero switch-over times, state-dependent setups, and a server that idles at the most recently served queue if the system is empty (*patient server*).

The literature on finite-capacity polling models (with more than one buffer) is severely limited. Ganz and Chlamtac (1988) and Tran-Gia and Raith (1988) propose approximate evaluation methods for cyclic polling and nonexhaustive service, Takagi (1991) and Kofman (1993) derive exact procedures for cyclic polling and exhaustive and nonexhaustive service disciplines, Jung and Un (1994) describe an exact method for cyclic polling and exhaustive service. The exact methods require the solution of sets of linear equations where the number of unknowns grows rapidly when the number of customer classes and the buffer capacities are increased.

In the next section, we propose a decomposition method for the first variant and in § 3 a decomposition method for the second variant of the multi-class queueing system. Some details, though, have to be omitted for the sake of brevity. They may be found in Krieg and Kuhn (2001a, 2001b). In § 4, multi-product kanban systems with setup times and lost sales are shown to be a possible application of the described models in manufacturing and we give numerical results for several example systems that indicate the accuracy of the two evaluation procedures.

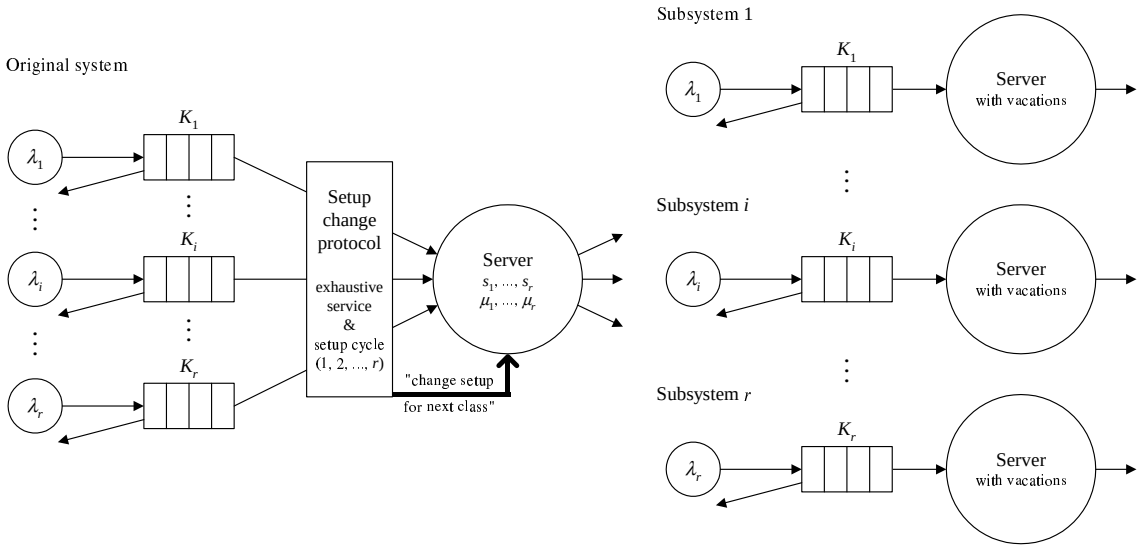

Figure 1: Decomposition of a system with r classes into r single-class subsystems

2 Variant A: Mandatory setups and a continuously roving server

Consider first the variant with mandatory setups and a continuously roving server. At most K_i customers of any class i, $i = 1, \ldots, r$, may be in the system at any time. For ease of presentation, we restrict ourselves to setup cycles in which each queue appears exactly once (*rotation cycles*). Without loss of generality, the server considers the customer classes $1, ..., r$ in ascending order of indices. For modeling purposes, we assume that customers of class i arrive according to a Poisson process with rate λ_i and independently of the customers of any other class. The length of a setup and the service time per customer of class i are independent and identically distributed random variables that follow exponential distributions with mean s_i and μ_i^{-1}, respectively. Thus, we have a multi-class $M/M/1/\{K_i\}_1^r$ queueing system with exhaustive-cyclic service and exponentially distributed setup times.

Since the Markov chain model of this system suffers from state space explosion with increasing number of customer classes and buffer capacities (Ibe and Trivedi 1990), we propose a decomposition method that splits up the original system with r classes into r single-class subsystems (see figure 1). The subsystems are represented by continuous-time Markov chains that are solved numerically. The occupation of the server with customers of other classes in the original system is modeled as a temporary shutdown, absence, or *vacation*, of the server in the subsystems. Thus, the server in any subsystem i, $i = 1, ..., r$, may, at any time, be in one of the three states "setup"(S_i), "busy period" (B_i), and "vacation" (V_i) (see figure 2).

If the setup is viewed as the last phase of the vacation, then the subsystems are $M/M/1/K_i$ queueing systems with cyclic-exhaustive service and multiple vacations (for literature on queueing systems with vacations see, for example, the surveys by Doshi 1986 and 1990, Teghem Jr. 1986, Takagi 1991, Chapter 2, and Takagi 1993).

In each subsystem, one unknown parameter has to be determined. This pa-

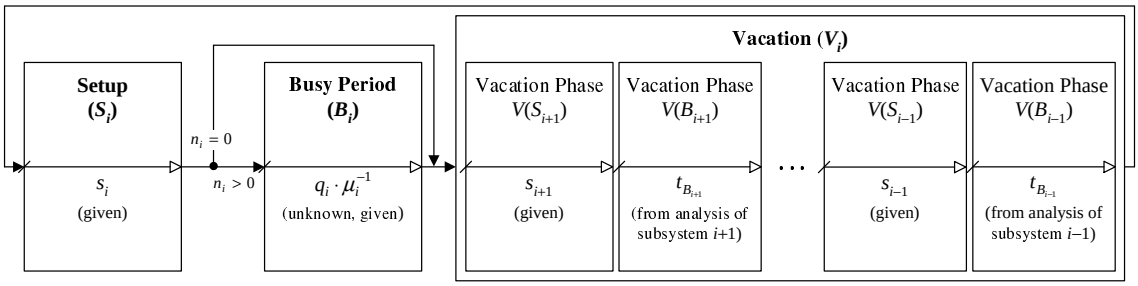

Figure 2: States of the server in subsystem i (class i), where n_i is the number of customers (of class i), s_i is the average setup time, q_i is the average number of served customers, μ_i^{-1} is the average service time per customer, and $t_{B_j}, j \in \{1, \ldots, r\} \backslash \{i\}$, is the average length of the busy period in subsystem j.

rameter is the average length of the busy period, denoted by t_{B_i} for the server in subsystem i. In subsystem $i+1$, as in any other subsystem except subsystem i, this parameter value is used as the average length of vacation phase $V(B_i)$, that is, the part of the "downtime" or "absence" of the server in subsystem $i+1$ that represents the occupation of the server in the original system with customers of class i. The details of the calculations that have to be done to obtain the current value of the unknown parameter t_{B_i} are given in Krieg and Kuhn (2001a). Here, we may only sketch the general procedure.

The average length of the busy period in subsystem i equals the product of the average number of served customers per cycle (q_i) and the average service time per customer (μ_i^{-1}). The value of q_i, however, is not known. But, after finding the steady-state probability vector of the Markov chain for subsystem i, it is possible to determine how the average duration of the cycle "setup-busy period-vacation" distributes percentage-wise among the three states of the server. Since the average duration of the setup is given by s_i and therefore known, the average length of the cycle and the average length of the busy period (t_{B_i}) may now be computed.

Figure 3 illustrates the scheme of the algorithm. After guessing rough estimates for the average duration of the busy periods $B_1, B_3, ..., B_r$, initial values are determined by analyzing subsystems 2 through r. Then the first rotation starts with the analysis of subsystem 1. A new value for t_{B_i} is computed and performance measures relating to class 1 are obtained. After that, the algorithm continues with equivalent calculations for subsystems 2 through r. Additional rotations of the same kind follow until a stopping criterion is simultaneously met by all subsystems. This may happen after the analysis of any subsystem during the last rotation.

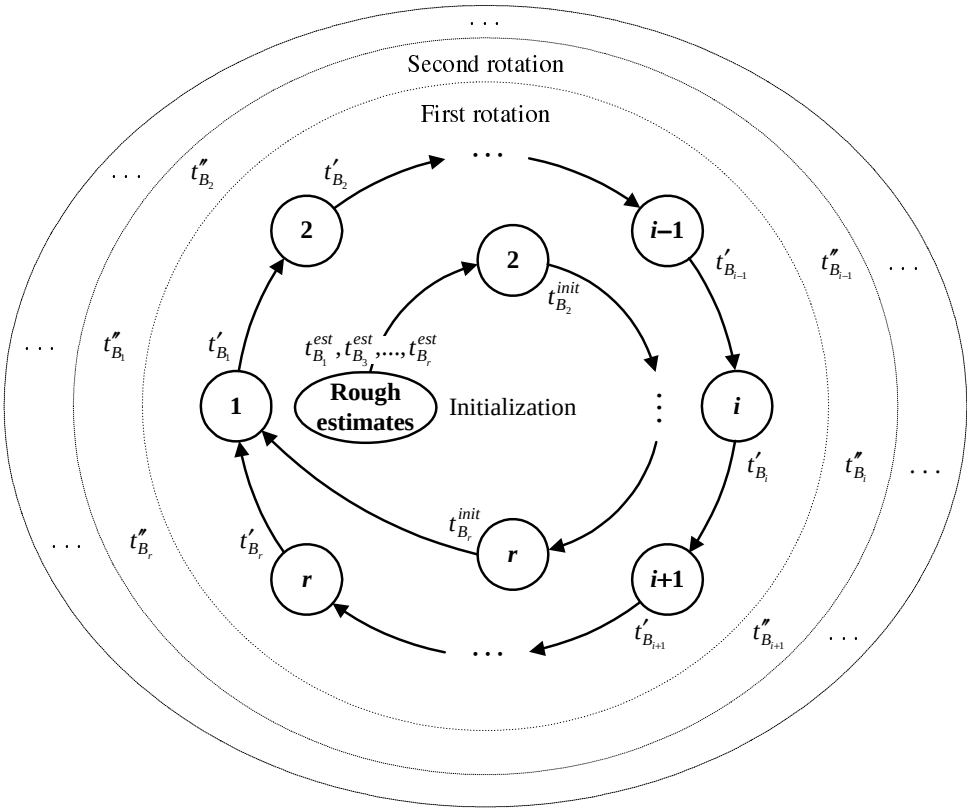

Figure 3: Scheme of the algorithm

3 Variant B: State-dependent setups and a patient server

Consider now the variant where setups occur only if there are customers waiting at the time the server turns to a queue and where the server idles at the current queue conserving the present setup if the system is completely empty. Other than that, the system is identical to the one analyzed before. An approach similar to the one presented above may be used for this variant. There is, however, an additional state called "idle period" that the server of any subsystem i may be in and the sequence of the states is slightly different (see figure 4). At the end of the busy period, the server may idle for a certain period of time before taking the vacation. The average length of this idle period is denoted by t_{I_i}. If the server finds no waiting customers upon return from a vacation, he instantly begins another vacation (*multiple vacations*).

The idle period in subsystem i represents the condition of the server in the original system when at the end of the busy period for class-i customers the system is completely empty. Before subsystem i may be analyzed, the average length of the idle period t_{I_i} has to be estimated (details are given in Krieg and Kuhn 2001b).

Other than above, we cannot use the steady-state probability of the setup state for class-i customers and the corresponding average setup time to determine the average cycle length because the setup state need not be attended by the server in every cycle. Rather, the average length of the setup state per cycle has also to be

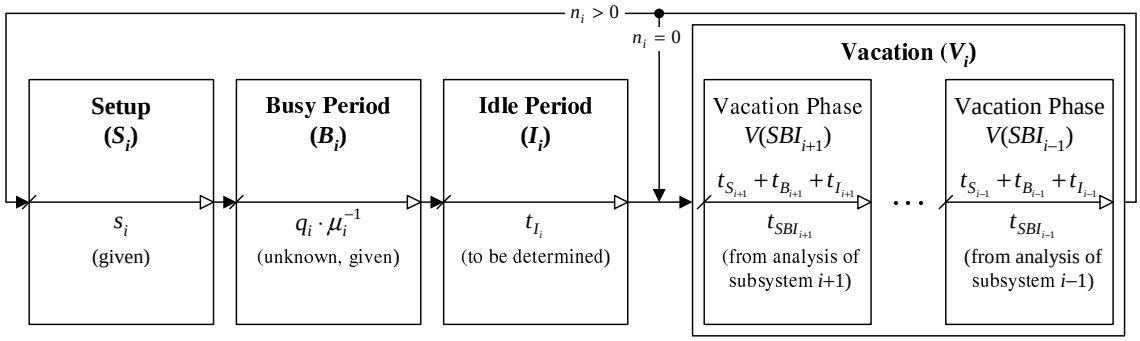

Figure 4: States of the server in subsystem i (class i), where n_i is the number of customers (of class i), s_i is the average setup time, q_i is the average number of served customers, μ_i^{-1} is the average service time per customer, and $t_{S_j}(t_{B_j}, t_{I_j}), j \in \{1, \ldots, r\}\backslash\{i\}$, is the average length of the setup state (busy period, idle period) per cycle in subsystem j.

determined in addition to the average length of the busy period. For the sake of brevity, we refer to Krieg and Kuhn (2001b) for further details.

4 Numerical results: Multi-product kanban systems

A possible application of the discussed models are multi-product kanban systems with setup times and lost sales. Subtracting the blocking probabilities of the queueing system from one yields the fill rates of the kanban system, that is, the (long-run) fraction of demand filled immediately. A second important set of performance measures for a kanban system, the average inventory levels, may be obtained with $K_i - L_i$, where K_i is the number of kanbans for product i and L_i is the average number of class-i customers in the queueing system.

To indicate the accuracy of the procedures we give numerical results for several kanban systems. We generated sets of example systems by systematically changing the total traffic intensity (ρ) and the number of products (r) of a generally defined base system (see table 1). Parameter values for products $2, ..., r-1$ were chosen to yield equal differences between the traffic intensities, the processing rates, and the setup times of consecutive products, for example, $\rho_1 - \rho_2 = \rho_2 - \rho_3 = ... = \rho_{r-1} - \rho_r$. The reference values for the determination of the relative approximation errors were obtained with the exact Markov chain model (for systems with three products) and via simulation.

Tables 2 through 5 contain two values for each set of performance measures: the maximum and the average of the absolute percentage errors of all products (err_{max} and err_{avg}, respectively). To indicate if the maximum error resulted from under- or over-estimating the actual value, we used the negative or positive number, respectively. Tables 2 and 3 show the relative approximation errors for variant A, tables 4 and 5 the equivalent results for variant B. In these examples, most relative errors are close to or below 1.0%.

Table 1: Parameter Values and Ratios of the Base System

Total traffic intensity, ρ	0.8
Traffic intensity ratio ρ_1/ρ_r	1.6
Processing rate of product 1, μ_1	2.0
Processing rate ratio μ_1/μ_r	0.9
Setup to processing time ratio of product 1, s_1/μ_1^{-1}	2.0
Setup time ratio s_1/s_r	0.8
Required fill rates, f_i	95.0%

Table 2: Variant A: Relative Approximations Errors I

		Avg. Inv. Levels		Fill Rates	
ρ	K_1, K_2, K_3	err_{max}	err_{avg}	err_{max}	err_{avg}
0.50	$5, 5, 4$	1.88	1.86	1.16	1.08
0.55	$6, 6, 5$	1.91	1.89	1.02	0.96
0.60	$7, 6, 6$	2.03	1.78	1.18	0.91
0.65	$7, 7, 6$	1.57	1.51	0.93	0.81
0.70	$8, 8, 7$	1.36	1.26	0.71	0.56
0.75	$10, 10, 9$	1.23	1.06	0.40	0.21
0.80	$12, 12, 11$	1.11	0.86	−0.33	0.21
0.85	$15, 15, 13$	1.19	0.90	−0.87	0.68
0.90	$20, 20, 18$	2.01	1.54	−1.39	1.17
0.95	$33, 32, 29$	4.25	3.81	−1.73	1.53

Table 3: Variant A: Relative Approximations Errors II

		Avg. Inv. Levels		Fill Rates	
r	$K_1, \ldots, K_r$	err_{max}	err_{avg}	err_{max}	err_{avg}
3	$12, 12, 11$	1.11	0.86	−0.33	0.21
4	$12, 12, 11, 10$	0.79	0.58	−0.42	0.34
5	$12, 12, 11, 11, 10$	0.53	0.38	−0.42	0.35
6	$12, 12, 12, 11, 11, 10$	0.42	0.32	−0.41	0.33
7	$12, 12, 12, 11, 11, 10, 10$	0.31	0.26	−0.41	0.33
8	$12, 12, 12, 11, 11, 11, 10, 10$	0.29	0.23	−0.41	0.33
9	$12, 12, 12, 12, 11, 11, 11, 10, 10$	0.27	0.18	−0.31	0.25
10	$12, 12, 12, 12, 11, 11, 11, 10, 10, 10$	0.16	0.13	−0.31	0.31

Table 4: Variant B: Relative Approximations Errors I

		Avg. Inv. Levels		Fill Rates	
ρ	K_1, K_2, K_3	err_{max}	err_{avg}	err_{max}	err_{avg}
0.50	$5, 5, 5$	-0.87	0.41	0.58	0.31
0.55	$6, 6, 5$	0.46	0.34	0.46	0.30
0.60	$7, 6, 6$	0.55	0.39	0.41	0.24
0.65	$8, 7, 7$	0.67	0.45	0.20	0.10
0.70	$9, 8, 8$	0.61	0.40	-0.29	0.22
0.75	$10, 10, 9$	0.39	0.31	-0.63	0.58
0.80	$13, 12, 11$	0.36	0.29	-1.02	0.94
0.85	$16, 16, 14$	0.63	0.47	-1.40	1.30
0.90	$21, 21, 19$	1.47	1.28	-1.80	1.66
0.95	$33, 32, 29$	3.90	3.75	-2.00	1.84

Table 5: Variant B: Relative Approximations Errors II

		Avg. Inv. Levels		Fill Rates	
r	$K_1, \ldots, K_r$	err_{max}	err_{avg}	err_{max}	err_{avg}
3	$13, 12, 11$	0.36	0.29	-1.02	0.94
4	$13, 13, 12, 11$	0.27	0.22	-1.13	1.03
5	$13, 13, 12, 11, 11$	0.14	0.08	-1.14	1.10
6	$13, 13, 12, 12, 11, 10$	0.13	0.04	-1.14	1.09
7	$13, 13, 12, 12, 11, 11, 10$	0.00	0.09	-1.14	1.01
8	$13, 13, 12, 12, 12, 11, 11, 10$	0.00	0.05	-1.04	0.96
9	$13, 13, 12, 12, 12, 11, 11, 11, 10$	0.00	0.01	-1.03	0.91
10	$13, 13, 12, 12, 12, 11, 11, 11, 10, 10$	0.00	0.05	-0.94	0.85

5 Conclusion

We have presented decomposition methods for two variants of a multi-class queueing system with a single server, finite buffers, and setup times. Whereas in variant A setups are mandatory and the server constantly switches queues performing setup after setup if the system is empty, variant B is characterized by state-dependent setups and a patient server. In both variants, queues are served exhaustively and they are considered in a fixed sequence that is repeated cyclically. Due to the finite buffers, arriving customers are rejected if all positions are occupied in the queue of their class.

Both decomposition methods were observed to approximate the performance measures with small relative errors, and the algorithms converged fast and reliably. Therefore, they are well-suited for analyzing and optimizing, for example, multi-product kanban systems with setup times and lost sales. In Krieg and Kuhn (2001c) we discuss an optimization procedure built around the proposed decomposition method for variant A that finds optimal or near-optimal solutions to the *Multi-Product Kanban System Configuration Problem* (MPKSCP).

References

Altiok, T. and G. A. Shiue (1994). Single-stage, multi-product production/inventory systems with backorders. *IIE Transactions 26*(2), 52–61.

Anupindi, R. and S. Tayur (1998). Managing stochastic multiproduct systems: model, measures, and analysis. *Operations Research 46*(3), S98–S111. Supp.

Bonvik, A. M., Y. Dallery, and S. B. Gershwin (2000). Approximate analysis of production systems operated by a CONWIP/finite buffer hybrid control policy. *International Journal of Production Research 38*(13), 2845–2869.

Buzacott, J. A. and J. G. Shanthikumar (1993). *Stochastic Models of Manufacturing Systems*. Englewood Cliffs, N.J.: Prentice-Hall.

Dallery, Y. and Y. Frein (1993). On decomposition methods for tandem queueing networks with blocking. *Operations Research 41*(2), 386–399.

Dallery, Y. and S. B. Gershwin (1992). Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems 12*, 3–94. Special issue.

Di Mascolo, M., Y. Frein, and Y. Dallery (1996). An analytical method for performance evaluation of kanban controlled production systems. *Operations Research 44*(1), 50–64.

Doshi, B. (1990). Single server queues with vacations. In H. Takagi (Ed.), *Stochastic Analysis of Computer and Communication Systems*, pp. 217–265. Amsterdam: Elsevier Science Publishers B.V. (North-Holland).

Doshi, B. T. (1986). Queueing systems with vacations – a survey. *Queueing Systems 1*, 29–66.

Federgruen, A. and Z. Katalan (1994). Approximating queue size and waiting time distributions in general polling systems. *Queueing Systems 18*, 353–386.

Ganz, A. and I. Chlamtac (1988). Queueing analysis of finite buffer token networks. *Performance Evaluation Review 16*, 30–36.

Gershwin, S. B. (1994). *Manufacturing Systems Engineering*. Englewood Cliffs, N.J.: Prentice Hall.

Ibe, O. C. and K. S. Trivedi (1990). Stochastic Petri net models of polling systems. *IEEE Journal on Selected Areas in Communications 8*(9), 1649–1657.

Jung, W. Y. and C. K. Un (1994). Analysis of a finite-buffer polling system with exhaustive service based on virtual buffering. *IEEE Transactions on Communications 42*(12), 3144–3149.

Kim, E. and M. P. Van Oyen (2000). Finite-capacity multi-class production scheduling with set-up times. *IIE Transactions 32*(9), 807–818.

Kofman, D. (1993). Blocking probability, throughput and waiting time in finite capacity polling systems. *Queueing Systems 14*(3-4), 385–411.

Krieg, G. N. and H. Kuhn (2001a). A decomposition method for multi-product kanban systems with setup times and lost sales. Working paper, Faculty of Business Administration and Economics, Catholic University of Eichstätt, 85049 Ingolstadt, Germany.

Krieg, G. N. and H. Kuhn (2001b). A decomposition method for multi-product kanban systems with state-dependent setups and lost sales. Working paper, Faculty of Business Administration and Economics, Catholic University of Eichstätt, 85049 Ingolstadt, Germany.

Krieg, G. N. and H. Kuhn (2001c). Production planning of multi-product kanban systems with significant setup times. In B. Fleischmann, U. Derigs, W. Domschke, R. Lasch, and U. Rieder (Eds.), *Operations Research Proceedings 2000: Selected Papers of the Symposium on Operations Research (SOR'00)*, Dresden, Germany, September 9–12, 2000. Springer, Berlin.

Olsen, T. L. (1999). A practical scheduling method for multiclass production systems with setups. *Management Science 45*(1), 116–130.

Papadopoulos, H. T., C. Heavey, and J. Browne (1993). *Queueing Theory in Manufacturing Systems Analysis and Design*. London: Chapman & Hall.

Takagi, H. (1990). Queueing analysis of polling models: an update. In H. Takagi (Ed.), *Stochastic Analysis of Computer and Communication Systems*, pp. 267–318. Amsterdam: Elsevier Science Publishers B.V. (North-Holland).

Takagi, H. (1991). Analysis of finite-capacity polling systems. *Advances in Applied Probability 23*, 373–387.

Takagi, H. (1993). *Queueing Analysis, Volume 2: Finite Systems*. Amsterdam: North-Holland.

Teghem Jr., J. (1986). Control of the service process in a queueing system. *European Journal of Operational Research 23*, 141–158.

Tran-Gia, P. and T. Raith (1988). Performance analysis of finite capacity polling systems with nonexhaustive service. *Performance Evaluation 9*, 1–16.