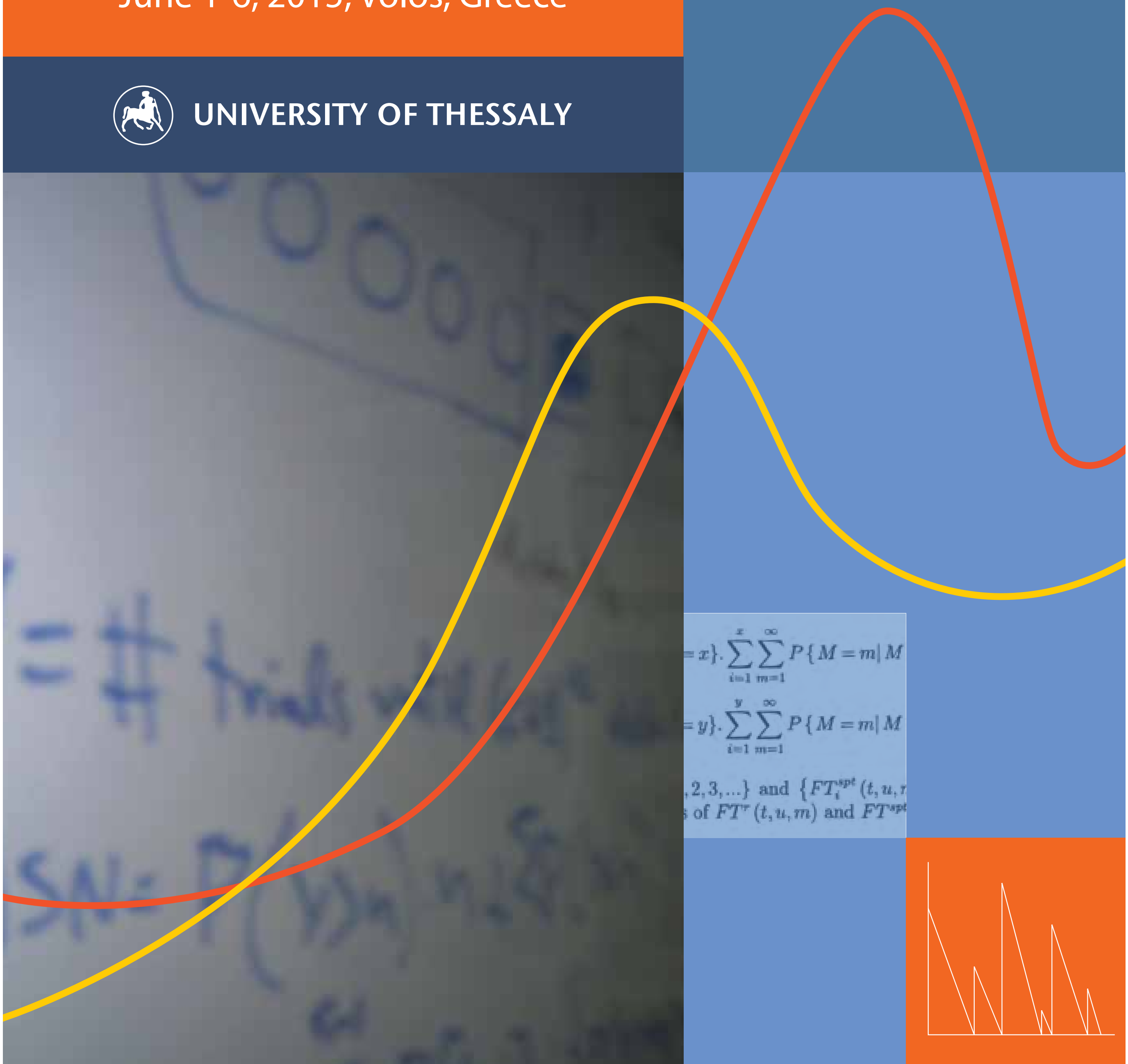
SMMSO 2015

10th conference on stochastic models of manufacturing and service operations

June 1-6, 2015, Volos, Greece

UNIVERSITY OF THESSALY



SMMSO 2015

10th Conference on Stochastic Models of Manufacturing and Service Operations

June 1-6, 2015
Volos, Greece

UNIVERSITY OF THESSALY

Πανεπιστημιακές
Εκδόσεις Θεσσαλίας

Table of Contents

Using shadow prices in a linear programing representation of kanban system dynamics to maximize system throughput

George Liberopoulos
Department of Mechanical Engineering, University of Thessaly, Volos, Greece, email: glib@mie.uth.gr

Kostas Takoumis
Department of Mechanical Engineering, University of Thessaly, Volos, Greece, email: kostast137@gmail.com

Dimitrios Pandelis
Department of Mechanical Engineering, University of Thessaly, Volos, Greece, email: d_pandelis@mie.uth.gr

We report numerical experimentation with the simple idea of using shadow prices in a linear programming representation of the dynamics of a serial kanban control system to obtain indicative sample path derivative estimates of system throughput with respect to the (integer) number of kanbans per stage, and drive a rudimentary gradient-based procedure to optimally allocate a fixed number of kanbans among different stages for maximum throughput.

Keywords: discrete event dynamic system simulation; linear programming; sample path gradient estimator; kanban system

1 Introduction

Recently, Schruben (2000) and Chan and Schruben (2008) proposed a methodology for modeling and simulating *Discrete Event Dynamic Systems* (DEDS) based on representing the dynamics of a DEDS by an *Event Relationship Graph* (ERG), converting the ERG into a *Mathematical Programming* (MP) problem, and solving the MP problem. This methodology was applied, among others, to multistage production systems, and in particular pull production control systems (Alfieri and Matta, 2012b).

Modeling the sample path of a DEDS as the solution to an MP problem allows the use the well-developed MP algorithms for performance evaluation purposes. It also paves the way for exploiting MP theory, such as duality, for the detection of system properties (e.g., Chan and Schruben, 2003), and using MP techniques, such as sensitivity analysis, for design and optimization purposes. Chan and Schruben (2008) proposed using the dual variables of the MP problem to compute sample path derivatives with respect to continuous system parameters. Chan and Schruben (2006) and Chan and Closser (2013) further demonstrated the use of dual variables for computing gradient estimators of system performance with respect to continuous parameters (for which *Infinitesimal Perturbation Analysis* (IPA) generates consistent gradient estimators), such as service times, for G/G/1 and G/G/m queues, respectively.

Matta (2008) and Alfieri and Matta (2012a) modeled the dynamics of a multi-stage tandem queuing system as a *Mixed Integer Linear Programming* (MILP) problem which can be used to simultaneously simulate the system and optimize its integer parameters (buffer capacities). They also proposed a procedure for relaxing the MILP problem into a computationally more tractable *Linear Programming* (LP) problem. Matta et *al*. (2014) further pursued the idea of simultaneously simulating and optimizing DEDSs by proposing a methodology based on representing both the dynamics of a DEDS and as well as the constraints and decision variables of the underlying optimization problem by a special class of ERGs called “ERG Lite” (ERGL), converting the resulting ERGL into an MILP problem, and solving the MILP problem. They demonstrated the methodology on a kanban control system.

In this paper, we experiment with the simple idea of using the shadow prices (dual variable values) of an LP representation of the dynamics of a serial kanban control system to obtain indicative sample path derivative estimates of system throughput with respect to the (integer) number of kanbans per stage, and drive a rudimentary gradient-based procedure to optimally allocate a fixed number of kanbans among different stages for maximum throughput. Chan and Schruben (2006) and Chan and Closser (2013) used dual variables to compute gradient estimators with respect to continuous parameters that appear as constants in the MP formulation. Here, we use them to compute gradient estimators with respect to integer parameters that appear as indexes of continuous decision variables.

The remainder of this paper is organized as follows. In Section 2, we present the LP model of the kanban system and the gradient-based procedure, and in Section 3, we report on numerical experimentation with this procedure for kanban systems with 3, 5, 6, and 10 stages. We conclude in Section 4.

2 LP representation of kanban control system

We consider an I-stage serial kanban system, where each stage contains a single-machine workstation. Figure 1 shows a queueing network representation of such a system, for $I = 4$. The notation used in this figure is given in Table 1. The detailed description of the operation of the system can be found in Liberopoulos (2013).

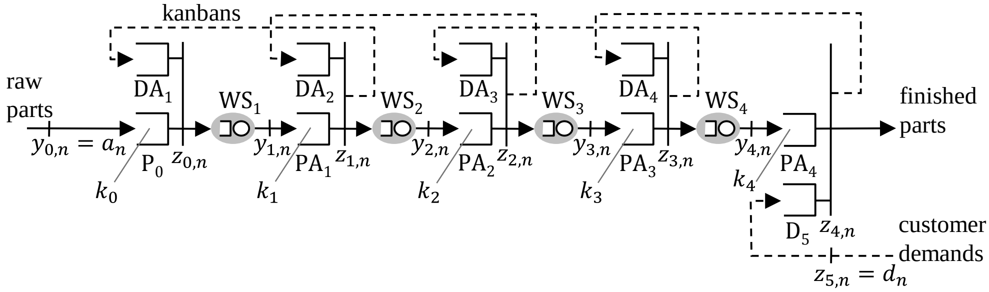


Figure 1. 4-stage kanban system driven by the arrival of customer demands and raw parts.

Table 1. Notation for an I-stage kanban system.

Indices	
i	Stage index: $i = 1, \dots, I$
n	Event index: $n = 1, \dots, N$
System components	
WS_i:	Workstation of stage i modelled as a single-server queue containing stage-i in-process parts (WIP) with stage-i kanbans attached to them, $i = 1, \dots, I$
PA_i:	Queue containing stage-i finished parts with stage-i kanbans attached to them, $i = 1, \dots, I$
DA_i:	Queue containing demands for stage-i parts with stage-i kanbans attached to them, $i = 1, \dots, I$
P_0:	Queue containing raw parts
D_{I+1}:	Queue containing backordered customer demands
Parameters	
k_i:	Total number of stage-i kanbans, $i = 1, \dots, I$
k_0:	Initial number of raw parts in the system
Event times	
a_n:	Arrival time of n^{th} raw part to the system, $n = 1, \dots, N$
d_n:	Arrival time of n^{th} customer demand to the system, $n = 1, \dots, N$
$t_{i,n}$:	Processing time of n^{th} part in station $i, i = 1, \dots, I,\ n = 1, \dots, N$
$y_{i,n}$:	Completion time of n^{th} part in station $i, i = 1, \dots, I,\ n = 1, \dots, N$
$z_{i,n}$:	Departure time of n^{th} part from stage $i, i = 1, \dots, I,\ n = 1, \dots, N$
$z_{0,n}$:	Release time of n^{th} part into stage 1, $n = 1, \dots, N$
$z_{I+1,n}$:	Arrival time of n^{th} customer demand to the system ($= d_n$), $n = 1, \dots, N$

Assuming that initially queue P_0 contains k_0 raw parts and queue PA_i contains k_i stage-i finished parts, each with a stage-i kanban attached to it, $i = 1, \dots, I$, the event times of the kanban system are related by the following "max +" equations:

$$y_{i,n} = t_{i,n} + \max\{y_{i,n-1}, z_{i-1,n}\}, \qquad i = 1, \dots, I,\ n = 1, \dots, N \tag{1}$$
$$y_{0,n} = a_n, \qquad n = 1, \dots, N \tag{2}$$
$$y_{i,n-k_i} = 0, \qquad i = 0, \dots, I,\ n = 0, \dots, k_i \tag{3}$$
$$z_{i,n} = \max\{y_{i,n-k_i}, z_{i+1,n}\}, \qquad i = 0, \dots, I,\ n = 1, \dots, N \tag{4}$$
$$z_{I+1,n} = d_n, \qquad n = 1, \dots, N \tag{5}$$

The above equations can be used to generate a sample path of the kanban system for N events (arrivals of raw parts, arrival of customer demands, and processing of parts) in a traditional DEDS simulation approach. Alternatively, the sample path can be generated by solving the following LP problem (see Alfieri and Matta, 2012):

$$\min \sum_{n=1}^{N}\left(\sum_{i=1}^{I} y_{i,n} + \sum_{i=0}^{I} z_{i,n}\right) \quad (6)$$

subject to

$$y_{i,n} \geq t_{i,n} + y_{i,n-1}, \quad i = 1,\dots,I,\ n = 1,\dots,N \quad (7)$$

$$y_{i,n} \geq t_{i,n} + z_{i-1,n}, \quad i = 1,\dots,I,\ n = 1,\dots,N \quad (8)$$

$$y_{0,n} = a_n, \quad n = 1,\dots,N \quad (9)$$

$$y_{i,n-k_i} = 0, \quad i = 0,\dots,I,\ n = 0,\dots,k_i \quad (10)$$

$$z_{i,n} \geq y_{i,n-k_i}, \quad i = 0,\dots,I,\ n = 1,\dots,N \quad (11)$$

$$z_{i,n} \geq z_{i+1,n}, \quad i = 0,\dots,I,\ n = 1,\dots,N \quad (12)$$

$$z_{I+1,n} = d_n, \quad n = 1,\dots,N \quad (13)$$

$$y_{i,n} \geq 0, \quad i = 0,\dots,I,\ n = 1,\dots,N \quad (14)$$

$$z_{i,n} \geq 0, \quad i = 0,\dots,I+1,\ n = 1,\dots,N \quad (15)$$

LP (6)-(15) aims to minimize the completion times in all stations and the departures times from all stages over all parts, thus eliminating any unnecessary idle times. Once LP (6)-(15) is solved, various performance measures can be computed from the solution. For example, the average flow time of a part in the system, denoted by FT_{aver}, is computed as:

$$FT_{aver} = \left[\sum_{n=1}^{N-\sum_{i=1}^{I} k_i}\left(z_{I,n+\sum_{i=1}^{I} k_i} - y_{o,n}\right)\right] / \left(N - \sum_{i=1}^{I} k_i\right) \quad (16)$$

The average throughput of the system, denoted by TH_{aver}, is computed as:

$$TH_{aver} = \left(N - \sum_{i=1}^{I} k_i\right) / \left(z_{I,N} - z_{I,\sum_{i=1}^{I} k_i}\right) \quad (17)$$

Finally, the average work in process, denoted by WIP_{aver}, is computed from Little's formula as:

$$WIP_{aver} = FT_{aver} \cdot TH_{aver} \quad (18)$$

The kanban system shown in Figure 1 is driven by the arrival of customer demands and raw parts which in turn trigger the processing of parts in the workstations. Figure 2 shows the same system assuming it has been "flooded" with an infinite number of customer demands and raw parts. We call the resulting system "saturated" kanban system. Studying the saturated system is important, because its throughput defines the upper limit of the average customer demand rate that the regular kanban system can satisfy.

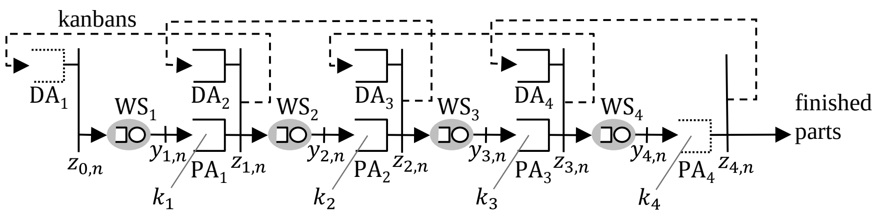


Figure 2. 4-stage saturated kanban system.

A sample path of the saturated kanban system for N events (processing of parts) can be generated by solving the following LP problem:

$$\min \sum_{n=1}^{N}\left(\sum_{i=1}^{I} y_{i,n} + \sum_{i=0}^{I} z_{i,n}\right) \quad \text{(shadow price)} \quad (19)$$

subject to

$$y_{i,n} \geq t_{i,n} + y_{i,n-1}, \quad i = 1,\dots,I,\ n = 1,\dots,N \quad (20)$$

$$y_{i,n} \geq t_{i,n} + z_{i-1,n}, \quad i = 1,\dots,I,\ n = 1,\dots,N \quad (21)$$

$$y_{i,n-k_i} = 0, \quad i = 1,\dots,I,\ n = 0,\dots,k_i \quad (22)$$

$$z_{i,n} \geq y_{i,n-k_i}, \quad (\delta_{i,n}) \quad i = 1,\dots,I,\ n = 1,\dots,N \quad (23)$$

$$z_{i,n} \geq z_{i+1,n}, \qquad i = 0, \dots, I-1, \; n = 1, \dots, N \tag{24}$$
$$y_{i,n} \geq 0, \qquad i = 1, \dots, I, \; n = 1, \dots, N \tag{25}$$
$$z_{i,n} \geq 0, \qquad i = 0, \dots, I, \; n = 1, \dots, N \tag{26}$$

LP (19)-(26) is obtained from LP (6)-(15) after setting $a_n = d_n = 0, \; n = 1, \dots, N$. The objective of LP (19)-(26) is the same as that of LP (6)-(15), namely, to minimize the completion times in all stations and the departures times from all stages over all parts, thus eliminating any unnecessary idle times. Minimizing these times results in minimizing the departure time of the last part from the last station, $z_{I,N}$, and therefore maximizes the average throughput of the system, which is given by (17).

The shadow price $\delta_{i,n}$ of constraint (23) gives the marginal improvement (decrease) in objective function (19) if the rhs of this constraint is (slightly) decreased. Note that the rhs of (23) can be decreased if k_i is increased, since $y_{i,n-k_i}$ is a non-increasing function of k_i; therefore, $\delta_{i,n}$ indirectly signals the decrease in the objective function caused by an increase in k_i, assuming that this increase affects only $y_{i,n-k_i}$. With this in mind, the sum of $\delta_{i,n}$ over all parts n, denoted by Δ_i, i.e., $\Delta_i = \sum_{n=1}^{N} \delta_{i,n}$, signals the overall decrease in the objective function, and hence the increase in system throughput, caused by an increase in k_i; therefore, it can be used as an indicator of the "derivative estimate" of the system throughput with respect to k_i, where we use the word "derivative" in a loose sense, since k_i can only take integer values.

Now, consider the problem of optimally allocating a fixed number of kanbans, say K, among different stages to maximize throughput. To solve this problem, we propose using the following rudimentary gradient-based iterative procedure: Start with some initial allocation, e.g., uniform allocation of kanbans among stages. In each iteration, increase by one the number of kanbans of the stage with the highest value of Δ_i and respectively decrease by one the number of kanbans of the stage with the lowest value of Δ_i (stages with only one kanban are skipped). The procedure stops if it yields a kanban allocation where all but one stages have a single kanban, or an allocation that has been found in a previous iteration. The allocation that yields the maximum throughput is chosen as the solution. The pseudo-code of the iterative procedure follows. We use index j as a superscript to denote the iteration number.

Pseudo-code of iterative procedure to optimally allocate *K* kanbans among *I* stages in a serial kanban control system

1. Set $j = 0$.
2. Choose initial number of kanbans $k_i^0, i = 1, \dots, I$, such that $\sum_{i=1}^{I} k_i^0 = K$.
3. Solve LP (19)-(26) with $k_i = k_i^j, i = 1, \dots, I$, and extract TH_{aver}^j and $\delta_{i,n}^j, i = 1, \dots, I, n = 1, \dots, N$.
4. Compute $\Delta_i^j = \sum_{n=1}^{N} \delta_{i,n}^j$; $i_{max}^j = \arg\max \Delta_i^j$; $i_{min}^j = \arg\min_{i:\, k_i^j > 1} \Delta_i^j$.
5. If $i_{max}^j = i_{min}^j$, GOTO step 10.
6. Set $k_{i_{max}}^{j+1} = k_{i_{max}}^j + 1$; $k_{i_{min}}^{j+1} = k_{i_{min}}^j - 1$.
7. If $\left[k_i^{j+1}, i = 1, \dots, I\right] = \left[k_i^{j'}, i = 1, \dots, I\right]$ for some $j' = 0, \dots, j$, GOTO step 10.
8. Set $j = j + 1$.
9. GOTO step 3.
10. END.

The above procedure is very crude and does not guarantee finding the optimal allocation, because the shadow price-based gradient estimates that it uses are only indicative, the throughput value produced by the LP is itself an estimate, and the decision variables are integers. Once the procedure comes to completion and yields a solution, we may further search locally around that solution to find a better one.

3 Numerical Results

In this section, we report on numerical experimentation with the iterative procedure outlined in the previous section, for kanban systems with $I = 3$, 5, 6, and 10 stages. In all instances examined, we assumed that the processing times in the workstation of stage i are independent, exponentially distributed random variables with mean $1/\mu_i$, $i = 1, \dots, I$. We modelled LP (19)-(26) using GAMS 24.1.3 and solved it using CPLEX 12.5.1 on an Intel Core i5 at 2.67GHz, with 6GB RAM. The results are shown in Tables 2-5. The first I columns show the processing rates μ_i; the next column shows the total number of kanbans K; the next $I + 1$ columns show the initial number of kanbans k_i^0 and the respective average throughput TH_{aver}^0; the next $I +$

1 columns show the optimal number of kanbans k_i^* and the respective maximum average throughput TH_{aver}^*; the last column shows the percent increase in throughput, $100 \cdot (TH_{aver}^* - TH_{aver}^0)/TH_{aver}^0$. Figures 3-6 show plots of the processing rates μ_i and the optimal number of kanbans k_i^* vs. stage $i, i = 1, \dots, I$.

Table 2. Results for the 3-stage kanban system (N = 60,000).

μ_1	μ_2	μ_3	K	k_1^0	k_2^0	k_3^0	TH_{aver}^0	k_1^*	k_2^*	k_3^*	TH_{aver}^*	% incr
1	1	1	10	3	4	3	0.8215	1	8	1	0.8324	1.33
			15	5	5	5	0.8705	1	13	1	0.8822	1.34
1	2	1	6	2	2	2	0.8336	1	4	1	0.8373	0.44
			8	3	2	3	0.8764	1	5	2	0.8794	0.34
			10	3	4	3	0.9045	1	6	3	0.9047	0.02
			15	5	5	5	0.9380	6	3	6	0.9382	0.02
2	1	2	6	2	2	2	0.9440	1	4	1	0.9643	2.15
			8	3	2	3	0.9703	1	6	1	0.9878	1.80
			10	3	4	3	0.9905	1	8	1	0.9952	0.47
			15	5	5	5	0.9971	1	12	2	0.9978	0.07
3	2	1	6	2	2	2	0.9718	1	4	1	0.9866	1.52
			8	3	2	3	0.9899	1	6	1	1.0020	1.22
			10	3	4	3	1.0019	1	7	2	1.0058	0.39
			15	5	5	5	1.0065	1	5	9	1.0072	0.07
1	2	3	6	2	2	2	0.9684	2	3	1	0.9832	1.53
			8	3	2	3	0.9869	4	3	1	0.9988	1.21
			10	2	2	2	0.9998	2	7	1	1.0027	0.29
			15	5	5	5	1.0033	4	10	1	1.0038	0.05

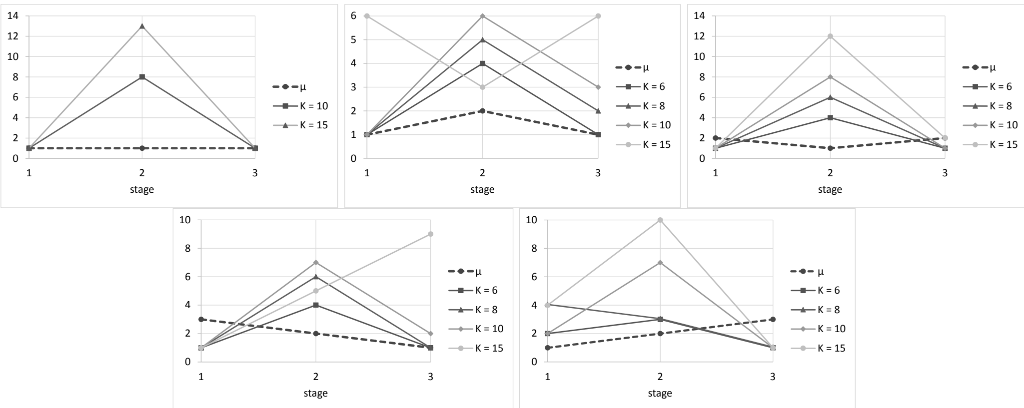


Figure 3. Plots of μ_i and k_i^* vs. i for the 3-stage kanban system (N = 60,000).

Table 3. Results for the 5-stage kanban system (N = 50,000).

μ_1	μ_2	μ_3	μ_4	μ_5	K	k_1^0	k_2^0	k_3^0	k_4^0	k_5^0	TH_{aver}^0	k_1^*	k_2^*	k_3^*	k_4^*	k_5^*	TH_{aver}^*	% incr
1	1	1	1	1	6	1	2	1	1	1	0.5280	1	1	2	1	1	0.5397	2.22
					7	1	2	2	1	1	0.5767	1	2	1	2	1	0.5844	1.34
					8	1	2	2	2	1	0.6258	1	2	2	2	1	0.6258	0.00
					10	2	2	2	2	2	0.6653	1	3	2	3	1	0.6815	2.43
					15	3	3	3	3	3	0.7512	1	5	3	5	1	0.7655	1.90
1	1	4	1	1	6	1	2	1	1	1	0.6005	1	2	1	1	1	0.6005	0.00
					7	1	2	2	1	1	0.6214	1	2	1	2	1	0.6559	5.55
					8	1	2	2	2	1	0.6740	1	3	1	2	1	0.6864	1.84
					10	2	2	2	2	2	0.7228	1	4	1	3	1	0.7414	2.57
					15	3	3	3	3	3	0.8009	1	6	1	6	1	0.8196	2.33

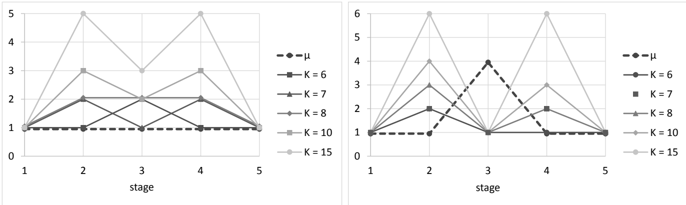


Figure 4. Plots of μ_i and k_i^* vs. i for the 5-stage kanban system (N = 50,000).

Table 4. Results for the 6-stage kanban system (N = 30,000).

μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	K	k_1^0	k_2^0	k_3^0	k_4^0	k_5^0	k_6^0	TH_{aver}^0	k_1^*	k_2^*	k_3^*	k_4^*	k_5^*	k_6^*	TH_{aver}^*	% incr
1	2	3	3	2	1	18	3	3	3	3	3	3	0.9446	3	5	2	2	4	2	0.9469	0.24
1	1	1	1	1	1	7	1	1	2	1	1	1	0.5076	1	1	1	2	1	1	0.5083	0.14
						8	1	1	2	2	1	1	0.5459	1	1	2	1	2	1	0.5469	0.18
						12	2	2	2	2	2	2	0.6482	1	3	2	2	3	1	0.6605	1.90
						18	3	3	3	3	3	3	0.7374	1	5	3	3	5	1	0.7497	1.67
3	2	1	1	2	3	18	3	3	3	3	3	3	0.8542	1	1	7	7	1	1	0.9265	8.46

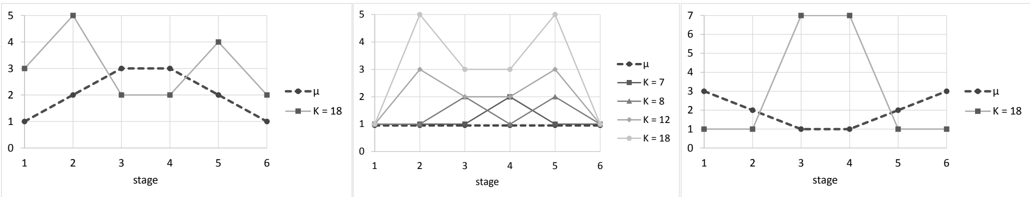


Figure 5. Plots of μ_i and k_i^* vs. i for the 6-stage kanban system (N = 30,000).

Table 5. Results for the 10-stage kanban system (N = 10,000).

μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8	μ_9	μ_{10}	K	k_1^0	k_2^0	k_3^0	k_4^0	k_5^0	k_6^0	k_7^0	k_8^0	k_9^0	k_{10}^0	TH_{aver}^0	k_1^*	k_2^*	k_3^*	k_4^*	k_5^*	k_6^*	k_7^*	k_8^*	k_9^*	k_{10}^*	TH_{aver}^*	% incr
1	1	1	1	1	1	1	1	1	1	12	1	1	1	1	2	2	1	1	1	1	0.4692	1	1	1	2	1	1	2	1	1	1	0.4765	1.56

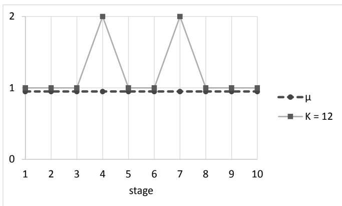


Figure 6. Plots of μ_i and k_i^* vs. i for the 10-stage kanban system (N = 10,000).

The results indicate that for the 3-stage kanban system, it is optimal to allocate the majority of kanbans at the middle stage and only one or a few kanbans at the end stages. This is expected, because the middle stage has a smaller system efficiency that the end stages, as it is subject to both blockage and starvation, whereas the end stages may only be blocked (first stage) or starved (last stage). For large values of K, the average throughput is insensitive to the kanban allocation; hence, many different allocations yield similar average throughput estimates. It is for this reason that in some instances the "optimal" kanban allocation shown in Table 2 deviates from the "inverse bowl" shape which is characteristic of kanban systems with more restricted values of K. If the production system is unbalanced (i.e., the processing rates at different stages are unequal), then stages with lower processing rates tend to need more kanbans than in the balanced system. Finally, the average throughput improvement seems to be modest.

For the 5-, 6-, and 10-stage kanban systems, the results indicate that the optimal kanban allocation has a "Λ" or "M" or more generally a "Panama hat" shape, where the crease in the middle is more pronounced for larger values of K. More specifically, for small values of K, it is optimal to allocate the kanbans evenly at the interior stages, starting with the middle stage. As K becomes larger, then it is optimal to allocate slightly more kanbans at the stages that are adjacent to the middle stage than in the middle stage itself. As in the 3-stage system, if the production system is unbalanced, then stages with lower processing rates tend to need more kanbans than in the balanced system.

Figure 7 shows plots of the kanban levels k_i^j, shadow price-based gradient estimates Δ_i^j, and average throughput estimates TH_{aver}^j vs. the iteration number j and stage $i, i = 1, \dots, 6$, for the 6-stage kanban system with K = 18 and $(\mu_1, \dots, \mu_6)$ = (3,2,1,1,2,3), listed in the last row of Table 4. Initially, the kanbans are uniformly allocated among the 6 stages. For this allocation, the shadow price-based gradient estimates yielded by LP (19)-(26) are an order of magnitude higher for the middle stages than for the other stages. Thus, in the first iteration, the kanban levels of the middle stages are augmented by one, while those of the end stages are diminished by one. For this new allocation, the gradient estimates yielded by the LP are still one order of magnitude higher for the middle stages than for the other stages. Thus, in the second iteration, the kanban levels of the middle stages are once again augmented by one, while those of the end stages are diminished by one. When the kanban levels at the end stages reaches the value of one, it is the turn of the kanban levels of the stages next to the end stages to start diminishing, while the kanban levels of the middle stages keep augmenting. This process is repeated until all but the middle stages end up with one kanban. As is shown in the bottom row of Table 4 and the bottom plot of Figure 7, the average throughput increases by 8.46% from 0.8542 to 0.9265. This is the largest increase recorded among all the problem instances evaluated. For all other problem instances presented in Tables 2-5, the detailed results of the iterations of the optimization procedure leading to the optimal kanban allocation can be found in Takoumis (2014).

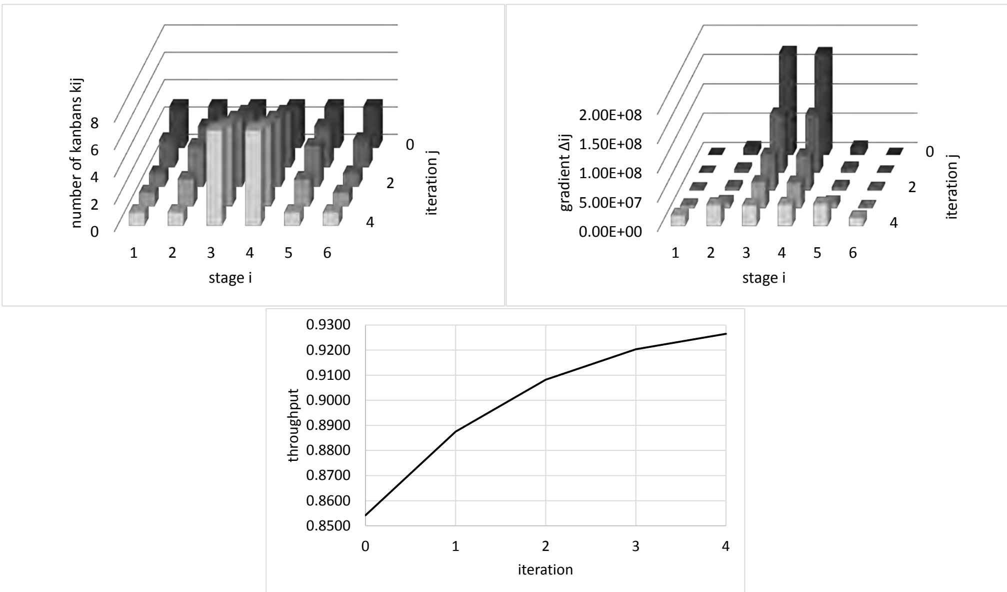


Figure 7. Plots of k_i^j, Δ_i^j, and TH_{aver}^j vs. stage i and iteration j, for the 6-stage kanban system with $(\mu_1, \dots, \mu_6)$ = (3,2,1,1,2,3).

4 Conclusions

Our numerical results indicate that the shadow prices of constraint (23) of the LP representation of kanban system dynamics LP (19)-(26) point to the right direction for improving system throughput. The optimization procedure proposed at the end of Section 2 is very crude and may not always lead to the optimal solution, especially if the objective function is insensitive to the design parameters (kanban levels), as is the case when the total number of kanbans K is large. However, it seems to be sufficient for finding good enough solutions for practical purposes or for use as initial solutions for more sophisticated algorithms.

Acknowledgements

The work in this paper was supported by grant MIS 379526 “ODYSSEUS: A holistic approach for managing variability in contemporary global supply chain networks,” which is co-financed by the EU-ESF and Greek national funds through NSRF – Operational Program “Education and Lifelong Learning” – “THALES: Reinforcement of the Interdisciplinary and/or Inter-Institutional Research and Innovation”.

References

Alfieri, A., A. Matta. 2012a. Mathematical formulations for approximate simulation of multistage production systems. *Eur. J. Oper. Res.* **219** (3) 773-783.

Alfieri, A., A. Matta. 2012b. Mathematical programming representation of pull controlled single-product serial manufacturing systems. *J. Intell. Manuf.* **23** (1) 23-35.

Chan, W. K. V., N. Closser. 2013. Sensitivity analysis of linear programming formulations for G/G/M queue. *Proc. 2013 Winter Simulation Conf.* IEEE, Piscataway, NJ, 667-677.

Chan, W. K. V., L. W. Schruben. 2003. Properties of discrete-event systems from their mathematical programming representations. *Proc. 2003 Winter Simulation Conf.* IEEE, Piscataway, NJ, 496-502.

Chan, W. K. V., L. W. Schruben. 2006. Response gradient estimation using mathematical programming models of discrete-event system sample paths. *Proc. 2006 Winter Simulation Conf.* IEEE, Piscataway, NJ, 272-278.

Chan, W. K. V., L. Schruben. 2008. Optimization models of discrete-event system dynamics. *Oper. Res.* **56** (5) 1218-1237.

Liberopoulos, G. 2013. Production release control: Paced, WIP-based, or demand-driven? Revisiting the push/pull and make-to-order/make-to-stock distinctions. J. M. Smith, B. Tan, eds. Handbook of Stochastic Models and Analysis of Manufacturing System Operations, International Series in Operations Research & Management Science, Vol. 192. Springer, New York, NY, 211-247.

Matta, A. 2008. Simulation optimization with mathematical programming representation of discrete event systems, *Proc. 2008 Winter Simulation Conf.* IEEE, Piscataway, NJ, 1393-1400.

Matta, A., G. Pedrielli, A. Alfieri. 2014. ERG Lite: Event based modeling for simulation-optimization of control policies in discrete event systems, *Proc. 2014 Winter Simulation Conf.* IEEE, Piscataway, NJ, 3983-3994.

Schruben, L. W. 2000. Mathematical programming models of discrete event system dynamics. *Proc. 2000 Winter Simulation Conf.* IEEE, Piscataway, NJ, 381-385.

Takoumis, K. 2014. *Performance Evaluation and Optimization of Kanban-Type Production-Inventory Systems via Simulation with the Use of Linear Programming*. M.Sc. Dissertation, Department of Mechanical Engineering, University of Thessaly, Volos, Greece (in Greek).

Sponsors

PAPADOPOULOS
A wonderful world of taste since 1922

ATHENIAN BREWERY

EURO
The Association of European Operational Research Societies

ChemiX
XOXACOS CHEMICALS

ΤΡΑΙΝΟΣΕ

UNIVERSITY OF THESSALY

eleN sa

Springer

PIRAEUS BANK GROUP CULTURAL FOUNDATION

ISBN: 978-960-9439-33-6