NETWORK MODELS IN MANUFACTURING AND COMMUNICATIONS

I MITRANI

Rapporteur: R de Lemos

Network Models in Manufacturing and Communications

Debasis Mitra
AT&T Bell Laboratories
Murray Hill, NJ 07974
USA

&

Isi Mitrani
University of Newcastle upon Tyne
Newcastle Upon Tyne NE1 7RU
UK

1. Introduction

Consider a tandem network of N service stations, or cells, with jobs coming from the outside into cell 1, moving from cell k to cell $k+1$ ($k = 1, 2, \ldots, N-1$), and leaving after cell N. Cell k contains one or more servers, and a finite buffer space where a limited number of jobs may reside before, during, and after receiving service (see Figure 1.1).

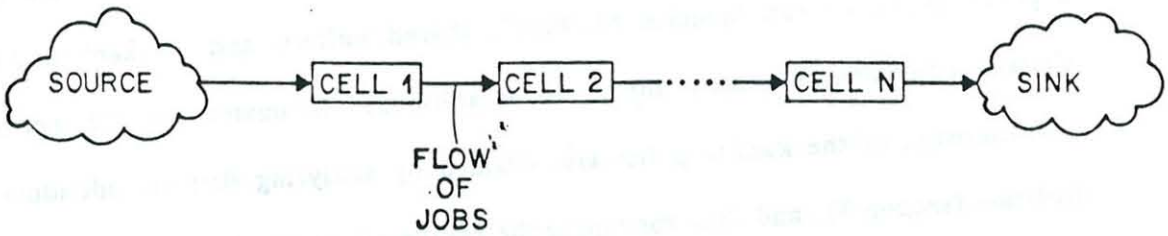

Figure 1.1 A tandem network of cells with finite buffers

Clearly, the space restrictions imply that the operation of any cell is influenced by that of cells both 'upstream' and 'downstream' from it in the network. Thus the policy governing the flow of jobs between cells is an important factor affecting the performance of the system. Our object is to examine, compare and evaluate a number of these policies.

The original motivation for the study was the modeling of production lines in manufacturing facilities. Of particular interest is a cell coordination scheme pioneered by the Toyota company, and called 'kanban', after the Japanese word for card [1-9]. The model which we abstract from the substantial literature on manufacturing practices uses a fixed number of kanbans, or cards, circulating within the confines of each cell, in order to

signal the status of the cell's inventory to its neighbors. Those signals may trigger job transfers between cells independently of the service processes that are in progress.

It should be pointed out, however, that the usefulness of our models is not restricted to the manufacturing area. Another field of applications is that of computer communications, where messages are buffered, processed and transferred from node to node. Here too, given appropriate information and data handling facilities, one could envisage different buffering policies that would influence the performance of the system (see, for instance, [10-14]).

Several existing and novel policies for the control and coordination of cells in tandem are described in Section 2. These include the classical transfer or manufacturing blocking, a policy which we call 'minimal blocking', 'shared buffers' and the kanban discipline. Certain equivalences among them are established. Estimates for the performance characteristics of the kanban policy are obtained by analyzing first an individual cell in isolation (section 3), and then combining the results of that analysis with an approximate model of the interaction between cells (Section 4). The accuracy of the approach and its application to performance evaluation is examined in Section 5.

This paper shares with references 13-20 the general approach of decomposing the given system into smaller, tractable subsystems and then approximately expressing the interaction between the subsystems in a coupling procedure. The contexts and the contents are however very diverse.

2. Description and Comparison of Policies

2.1 Transfer or manufacturing blocking

We start with what is perhaps the simplest and best-known buffering policy [10,13], see Figure 2.1.

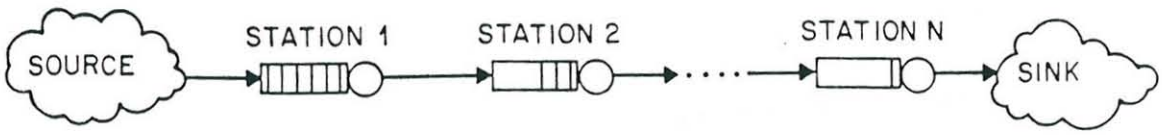

Figure 2.1. Transfer or manufacturing blocking.

There are N stations, numbered $1, 2, \ldots, N$. Assume that station 1 is preceded by an inexhaustible pool of jobs requiring processing, while station N is followed by a similar sink for completed jobs. Station k contains a single server and a buffer capable of holding up to C_k jobs ($k = 1, 2, \ldots, N$). If, at any time, there are jobs present in the buffer, one of them is occupying the server while the others are waiting to do so. When a service terminates, the completed job moves to station $k + 1$ (for $k < N$), provided that the latter's buffer is not full. If that buffer is full, then the job is blocked: it remains in station k, continuing to engage the server and preventing the start of a new service. As soon as a space becomes available in station $k + 1$, the blocked job moves there, freeing the server. All moves are instantaneous.

Note that there can never be an empty space in buffer 1 (because of the inexhaustible source), nor a blocked job in buffer N (because of the infinite sink). In all other stations a buffer location may be empty, occupied by a job waiting for or receiving service, or occupied by a blocked job (at most one location can be in this last state). Another feature of this system is that a service completion event in one station may trigger a number of simultaneous movements of blocked jobs in preceding stations. Information about the state of a station is assumed to be instantaneously available to its neighbors.

Thus, under the transfer blocking policy, a server that has just completed a service is

unable to start a new one when either

(i) there are no jobs requiring service in the attached buffer, or

(ii) the following (downstream) buffer is full.

A more formal description of the evolution of the sample paths for this model can be given by means of a set of recurrence relations. Let A_n^k and T_n^k be the instants when the n^{th} job arrives at the k^{th} station, and completes service there, respectively ($k = 1, 2, \ldots, N$; $n = 1, 2 \ldots$). Obviously, the instant when the n^{th} job departs from the k^{th} station is A_n^{k+1} if $k < N$, and T_n^k if $k = N$. Also, denote the service time of job n at server k by s_n^k.

For the transfer blocking discipline, we have

$$A_{n+1}^k = \max(T_{n+1}^{k-1}, A_{n+1-C_k}^{k+1}), \tag{2.1}$$

$$T_{n+1}^k = \max(A_{n+1}^k, A_n^{k+1}) + s_{n+1}^k, \tag{2.2}$$

for $k = 1, 2, \ldots, N$ and $n = 1, 2, \ldots$. By definition, $T_n^0 = 0$ and $A_n^{N+1} = T_n^N$. To see that (2.1) and (2.2) hold, note that the $(n+1)^{\text{th}}$ job enters station k either when it completes service in station $k-1$ (if it is not blocked there), or when the $(n+1-C_k)^{\text{th}}$ job enters station $k+1$. Similarly, the $(n+1)^{\text{th}}$ job starts service in station k either immediately upon entry (if the buffer there is empty), or when the n^{th} job enters station $k+1$.

Relations of the above type are useful in comparing different control policies [21].

A related discipline is sometimes referred to in the literature as 'communication blocking' [10]. Under it, the server in station k cannot start a new service if buffer $k+1$ is full. Conceptually, manufacturing blocking and communication blocking are similar (although their performance characteristics are different), since in both cases at most one job may be blocked in each station.

2.2 Minimal blocking

Consider the following less restrictive, blocking rule: when a completed job in station k is unable to move to station $k + 1$ (because the buffer there is full), it remains in buffer k but does not continue to engage the server. The latter is free to start a new service, provided that there is a job requiring one. Thus, of the jobs present in buffer k, some may be waiting for service, some may be blocked, (having completed their service but being unable to move) and one may be in service. When a departure from station $k + 1$ occurs, one of the blocked jobs in station k moves instantaneously to station $k + 1$, without disturbing the service that might be in progress.

We call this control policy 'minimal blocking'. In justification of the name, note that now a server is idle only when there are no jobs requiring service in its buffer.*

The core of this control policy is the functional separation of the tasks of *serving* and *moving*. The first is associated with the actual processing or manufacturing and the second with moving jobs from station to station, i.e. material handling in the manufacturing context. These two become separate, concurrent processes.

The minimal blocking policy can be described by a set of sample path evolution equations analogous to (2.1) and (2.2). In fact, (2.1) applies to this system without modification, while (2.2) is replaced by

$$T^k_{n+1} = \max(A^k_{n+1}, T^k_n) + s^k_{n+1} \ . \tag{2.3}$$

The change is explained by noting that, under minimal blocking, the $(n + 1)^{\text{th}}$ job starts service in station k either immediately upon arrival (if the server is idle), or when the n^{th}

* While this paper was being prepared, it came to our notice that the same policy was introduced and analyzed approximately in [14]. We use a different approximation method that reflects more closely the interaction between cells.

job completes service (rather than when it departs).

It seems intuitively obvious that transfer blocking causes more server idleness, and should therefore lead to lower throughput of jobs, than minimal blocking. Indeed, a stronger statement concerning the sample paths of the two systems can be established rigorously by induction on n and k in the evolution equations.

2.3 Systems with shared buffers

A more efficient use of storage space is achieved, if the job handling facilities allow it, by sharing buffers among several servers. Such a sharing increases resource utilization by reducing the likelihood that servers are deprived of work through either blocking or starvation [11,12].

Suppose that in a multi-station system such as the one depicted in Figure 2.2, the servers are grouped into *centers* indexed $1, 2, \ldots, N$. Center k contains the servers $(k, 1)$, $(k, 2), \ldots, (k, m_k)$ which cooperate by pooling their space allocations and sharing the resulting buffer space of capacity C_k.

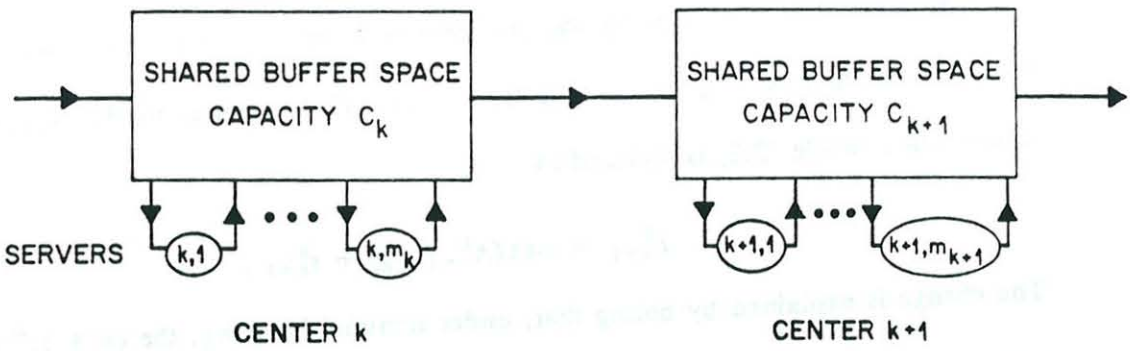

Figure 2.2: Centers, each with several servers and a shared buffer.

A job is able to enter center k if a free location is available. Having entered, the job waits and receives service first by server $(k, 1)$, then by server $(k, 2)$ and so on, until it is served

by server (k, m_k). At that point the job either leaves (if the following buffer is not full), or becomes blocked. The blocking may be of the transfer type, whereby the job continues to occupy server (k, m_k) and prevents it from starting a new service, or it may be minimal, with the job remaining in the buffer but releasing server (k, m_k).

Consider first a shared buffer system with transfer blocking. To describe the evolution of its sample path, introduce the instants $A_n^{k,j}$, when the n^{th} job 'arrives' at server (k, j) in center k $(j = 1, 2, \ldots, m_k)$. Of these, only $A_n^{k,1}$ is truly an arrival point: the job enters the buffer then. After that, arriving at a server means being ready to receive service by it. Also, let $T_n^{k,j}$ be the instant when the n^{th} job completes service at server j in station k. Then, by the argument leading to equation (2.1), we can write

$$A_{n+1}^{k,1} = \max(T_{n+1}^{k-1,m_{k-1}}, A_{n+1-C_k}^{k+1,1}) \; ; \; k = 1, 2, \ldots, N, \tag{2.4}$$

where $T_n^{0,m_0} = 0$ and $A_n^{N+1,1} = T_n^{N,m_N}$ by definition. For servers $2, 3, \ldots, m_k$ in station k, an arrival coincides with a service completion at the preceding server:

$$A_n^{k,j} = T_n^{k,j-1} \; ; \; k = 1, 2, \ldots, N, \;\; j = 2, 3, \ldots, m_k \, . \tag{2.5}$$

On the other hand, a job starts service at a server either immediately upon arrival (if the server was idle) or then the preceding job vacates it. Hence, denoting the service time of job n at server j in station k by $s_n^{k,j}$, we can write an equation similar to (2.2) for the completion instants:

$$T_{n+1}^{k,j} = \max(A_n^{k,j+1}, A_{n+1}^{k,j}) + s_{n+1}^{k,j} \; ; \; k = 1, 2, \ldots, N, \;\; j = 1, 2, \ldots, m_k \, . \tag{2.6}$$

Here, A_n^{k,m_k+1} is defined as being equal to $A_n^{k+1,1}$ if $k < N$, and to T_n^{N,m_N} if $k = N$.

2.4 Shared buffers with minimal blocking

The minimal blocking shared buffer system differs from the transfer blocking one only in the handling of a blocked job at the last server in a center. Now every server may start a new service as soon as the old one is completed (provided that there is a job requiring

service). The evolution equations (2.4) and (2.5) continue to hold without modification, while (2.6) is replaced by an equation similar to (2.3):

$$T^{k,j}_{n+1} = \max(T^{k,j}_{n}, A^{k,j}_{n+1}) + s^{k,j}_{n+1}\,;\ k = 1, 2, \ldots, N,\quad j = 1, 2, \ldots, m_k\,. \qquad (2.7)$$

It should be pointed out that the tandem arrangement of servers within each station is not the only one that can be envisaged. One could imagine shared buffer centers where jobs follow more complicated routes among the servers, including feedbacks, before exiting. For any given routing and blocking policy, it would be possible to write a set of evolution equations similar to (2.4)-(2.6) or (2.4), (2.5) and (2.7).

2.5 Kanban

The kanban policy was introduced in connection with manufacturing production lines and is still discussed primarily in that context. We shall therefore use the manufacturing terminology in describing our model.

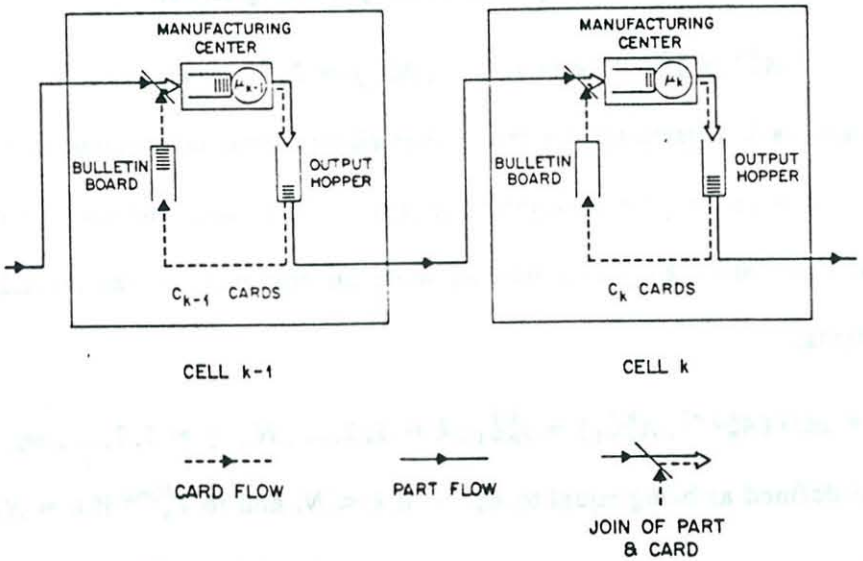

Figure 2.3

As shown in Figure 2.3, a kanban cell consists of

i. a manufacturing center (in the figure this contains a single server and its queue; however, we also allow networks of several servers and queues in series);

ii. an output hopper, where jobs are placed after completing service;

iii. a bulletin board, where requests for new jobs are posted.

In cell k there is a fixed number, C_k, of cards, or kanbans ($C_k \geq 1$). A job must acquire one of these cards in order to enter the cell, and must continue to hold it throughout its sojourn there. Thus, the total number of jobs in the cell can never exceed C_k. Unattached cards may be found only on the bulletin board, and are then considered as requests for new jobs. We assume that each server is the manufacturing center of cell k has buffer space large enough to hold all C_k jobs ($k = 1, 2, \ldots, N$). Hence, importantly, no incremental blocking due to finite buffers is introduced in the manufacturing centers of cells.

Suppose that Q is a job that has just left the manufacturing network in cell $k - 1$, and R is the card attached to it. Both Q & R move to the output hopper of cell $k - 1$, whereupon there are two possible courses of action:

a. if the bulletin board in cell k is empty, then Q & R wait in the output hopper of cell $k - 1$;

b. if that board is not empty, then the following moves occur instantaneously: Q is transferred from cell $k - 1$ to cell k, where it picks a waiting card and goes to the manufacturing network; R goes to the bulletin board of cell $k - 1$;

In the case (a), Q & R wait until a card appears in the bulletin board of cell k, at which point the moves described in (b) occur to the leading (job & card) pair in the output hopper (which may not be Q & R).

The above rules imply that the output hopper in cell $k - 1$ and the bulletin board in cell k cannot be nonempty simultaneously. In particular, the bulletin board in cell 1 is always empty because that cell is preceded by an inexhaustible pool of jobs. Similarly, the

output hopper in cell N is always empty because it is followed by an inexhaustible pool of demands.

We shall now show that the difference between the kanban and the minimal blocking shared buffer policy is apparent, rather than real. Although the equivalence holds more generally, we restrict our demonstration to the specific, simple context which we have followed so far in the paper, namely, the tandem configuration. Here the manufacturing centers of the kanban cells contain tandem networks with m_k servers in series in cell k. The contrasting minimal blocking shared buffer system (see Figure 2.2) has several centers arranged in tandem and service is given by the servers in each center in the order that they are indexed.

Proposition 1.

A kanban system where the k^{th} cell has C_k cards and a manufacturing center of m_k servers in series ($k = 1, 2, \ldots, N$), is equivalent to a minimal blocking shared buffer system in which the k^{th} center has m_k servers sharing a buffer of size C_k ($k = 1, 2, \ldots, N$; see section 2.3).

Proof.

To establish the proposition, it suffices to consider the following mapping between the states of kanban cell k and center k of the minimal blocking shared buffer system: To a card in the cell's bulletin board corresponds an empty location in the center; to a (job & card) pair in queue (k, j) of the manufacturing center of the cell corresponds a job waiting for, or receiving service at server (k, j); to a (job & card) pair in the kanban cell's output hopper, corresponds a blocked job in the buffer center. With this mapping in mind, a glance at the appropriate scheduling and transfer rules shows that conditions under which servers may work, and jobs may move between cells or stations, are the same.

Alternatively, it is easily seen that the sample path evolution equations for the two systems are the same. The equations for the kanban system are precisely (2.4), (2.5) and (2.7) with $A_n^{k,j}$ and $T_n^{k,j}$ respectively denoting the time of arrival at the queue of the server (k, j) and the time of service completion there of the n^{th} job at the manufacturing center of cell k ($k = 1, 2, \ldots, N$; $j = 1, 2, \ldots, m_k$; $n = 1, 2, \ldots$).

■

Observe that in the case of a single server in each manufacturing center, i.e. $m_k \equiv 1$ ($k = 1, 2, \ldots, N$), the kanban system is equivalent to the minimal blocking policy applied to buffered stations in tandem, the case treated in Section 2.2. The evolution equations are (2.1) and (2.3).

It is clear from the above results that the kanban cell model provides a rather general framework for studying systems consisting of service stations in tandem. In the following sections, our task will be the performance analysis of the kanban policy or, equivalently, the minimal blocking one. For conciseness and simplicity, we shall concentrate on the case of a single server in each manufacturing center, i.e. $m_k = 1$ ($k = 1, 2, \ldots, N$). It should be emphasized, however, that in the procedure that we give below servers with state-dependent service rates are no more difficult to handle than fixed-rate ones. On the other hand, a network of several servers in series can be approximated by a single state-dependent server. Hence, our methods can be applied to the general kanban cell model (or the general minimal blocking shared buffer system) by simply adding another level of decomposition. Thus, the presentation that follows involves no significant loss of generality.

3. AN ISOLATED KANBAN CELL

From now on, our aim is to devise efficient and reasonably accurate methods for

determining various steady-state performance measures of the kanban system. To complete the specification of the model, the service times in cell k will be assumed to be i.i.d. random variables distributed exponentially with mean $1/\mu_k$ ($k = 1, 2, \ldots, N$). Then the system state, defined as the vector of integers representing the numbers of cards and jobs in all bulletin boards and output hoppers, would be a Markov process to which existing solution techniques could be applied.

It should be pointed out, however, that the exact solution of a large kanban system is likely to be numerically intractable. This is because the size of the state space grows exponentially with the number of cells. For example, it is not difficult to show that, if there is a single card in each cell, then the number of states is approximately equal to $0.447(2.618)^{N+1}$, for large N.

Clearly, a good approximate solution is desirable. To obtain it, we shall decompose the model into semi-autonomous components. Each component consists of a single kanban cell supplied with jobs on one hand, and with demands for finished products on the other, by two independent Poisson processes. The rates of these processes, for cell k, are denoted by ρ_k and σ_k, respectively. At the moment we shall treat these rates as given. Later, they will be determined as functions of the basic system parameters, μ_k and C_k ($k = 1, 2, \ldots, N$).

Such a cell is said to be 'isolated'. The flow of jobs and cards through the isolated cell k is illustrated in Figure 3.1.

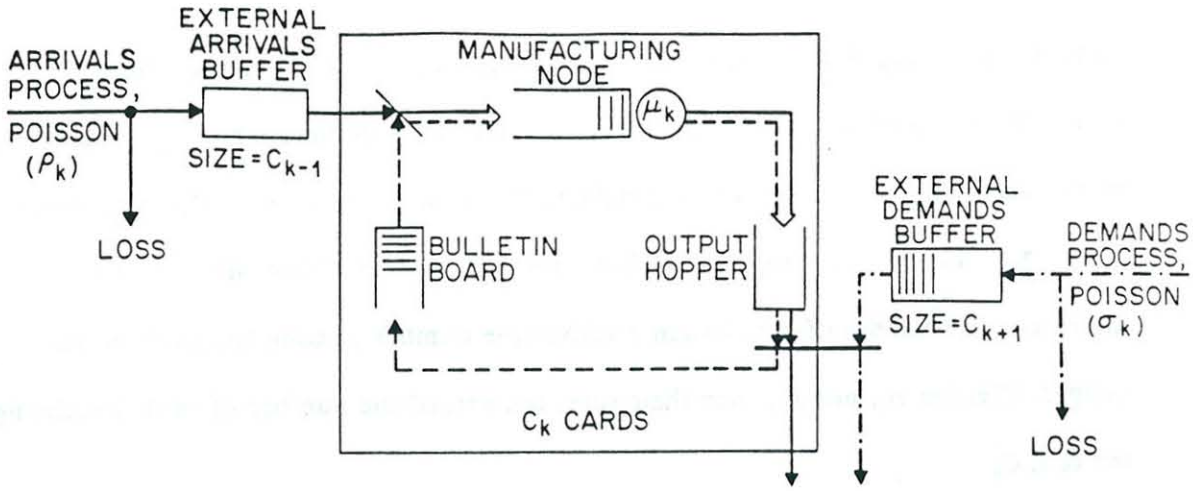

Figure 3.1. Isolated k^{th} kanban cell

There is an external buffer for the incoming jobs, of size C_{k-1}; a job which finds that buffer full is lost. Similarly, there is an external buffer for the demands for finished products, of size C_{k+1}; an incoming demand which finds it full is lost. Note that those buffer sizes are chosen to coincide exactly with the capacities of the output hopper in cell $k - 1$ and the bulletin board in cell $k + 1$, respectively.

Within the isolated cell, the scheduling rules are the same as in the original kanban model. A job (from the external arrivals buffer) enters the cell by picking up a card from the bulletin board, and together they join the manufacturing buffer. A completed job (in the output hopper) leaves the cell by being matched with a demand from the external demands buffer; then they both disappear, while the released card goes on the bulletin board.

These rules imply that the external arrivals buffer and the bulletin board cannot be simultaneously non-empty. Similarly, the external demands buffer and the output hopper cannot be simultaneously non-empty. Hence, the state of the isolated kanban cell k at time t can be described completely by a pair of integers, $(I_{k,t}, J_{k,t})$, where $-C_{k-1} \leq I_{k,t} \leq C_k$

and $-C_{k+1} \le J_{k,t} \le C_k$; moreover, $I_{k,t} + J_{k,t} \le C_k$. When $I_{k,t}$ is negative, $-I_{k,t}$ represents the number of jobs in the external arrivals buffer; when $I_{k,t}$ is positive, it represents the number of cards in the bulletin board; when $I_{k,t} = 0$, both buffer and board are empty. In the same way, $J_{k,t}$ (when negative) represents the number of demands in the external demand buffer or (when positive) the number of completed jobs in the output hopper. Neither $I_{k,t}$ nor $J_{k,t}$, nor their sum, can exceed the number of cards circulating in the cell, C_k.

The assumptions that have been made ensure that $M = \{I_{k,t}, J_{k,t}) ; t \ge 0\}$ is a finite Markov process. The state space, S_k, of that process, is of a size $|S_k| = (C_{k-1} + C_k + 1)(C_{k+1} + C_k + 1) - C_k(C_k + 1)/2$. The process M is irreducible, and therefore it has a steady-state distribution, which we shall denote by $\{p_{ij}\}$:

$$p_{ij} = \lim_{t \to \infty} P(I_{k,t} = i, J_{k,t} = j); \quad (i, j) \in S_k . \tag{3.1}$$

These probabilities satisfy the following set of balance equations:

$$\begin{aligned}
&[\mu_k 1(i < C_k \wedge j < C_k \wedge i + j < C_k) + \rho_k 1(i > -C_{k-1}) + \sigma_k 1(j > -C_{k+1})]p_{ij} \\
&= \mu_k[p_{i,j-1} 1(j > 0) + p_{i-1,j-1} 1(j \le 0)] \\
&\quad + \sigma_k[p_{i-1,j+1} 1(j \ge 0) + p_{i,j+1} 1(j < 0)] + \rho_k p_{i+1,j}, \quad (i, j) \in S_k ,
\end{aligned} \tag{3.2}$$

where $1(c)$ is equal to 1 if the condition c holds, 0 otherwise; by definition, $p_{ij} = 0$ if (i, j) is not in S_k. Equation (3.2), together with the normalizing equation, can be solved numerically for reasonably large values of C_{k-1}, C_k and C_{k+1}.

Two special cases of isolated cells deserve a separate mention. We know that the leftmost cell, $k = 1$, does not ever have to wait for the arrival of new job. This is achieved in our model by setting $\rho_1 = \infty$ and, say, $C_0 = 1$. The only feasible states are those of the form $(-1, j)$, for $j = -C_2, \ldots, 0, \ldots, C_1$. It is readily seen that the marginal distribution of $J_{1,t}$ for $t \to \infty$ is of the truncated geometric type:

$$p_{\cdot j} = (1 - \alpha)\,\alpha^{j+C_2}[1 - \alpha^{C_1+C_2+1}]^{-1}, \quad j = -C_2,\ldots, 0,\ldots, C_1, \quad k = 1, \quad (3.3)$$

where $\alpha = \mu_1/\sigma_1$. Similarly, the rightmost cell is modelled by setting $\sigma_N = \infty$ and $C_{N+1} = 1$; then the marginal distribution of $I_{N,t}$ for $t \to \infty$ is given by

$$p_{i\cdot} = (1 - \beta)\,\beta^{i+C_{N-1}}[1 - \beta^{C_{N-1}+C_N+1}]^{-1}, \quad i = -C_{N-1},\ldots, 0,\ldots, C_N, \quad k = N, \quad (3.4)$$

where $\beta = \mu_N/\rho_N$.

Let T_k be the steady-state throughput of parts in isolated cell k. That quantity is equal to the average number of jobs that enter the cell per unit time. Hence we can write, for $k > 1$,

$$T_k = \rho_k[1 - P(I_k = -C_{k-1})], \quad k = 2, 3,\ldots, N. \quad (3.5)$$

Alternatively, since the cell is in steady-state, the throughput is equal to the average number of demands that are accepted per unit time. When $k < N$, this can be written as

$$T_k = \sigma_k[1 - P(J_k = -C_{k+1})], \quad k = 1, 2,\ldots, N-1. \quad (3.6)$$

Equations (3.5)-(3.6) will be referred to as the 'fundamental identities'. They will be used in the derivation of the approximate solution for the whole system. These identities have also been obtained directly from the balance equations (3.2).

The scheme to be described in the next section is constituted from basic building blocks which are the individual, isolated kanban cells described in this section. In particular the viewpoint taken there is that, for each cell index k, the quantities T_k, $P(I_k = C_k)$, $P(I_k = -C_{k-1})$, $P(J_k = C_k)$ and $P(J_k = -C_{k+1})$ are each functions of two variables, ρ_k and σ_k. The reader should keep this important fact in mind since the notation to express it is cumbersome and will not be used. We do not have explicit representations of these functions (except for the case of $C_k \equiv 1$) but they are implicitly defined by the following procedure: for given values of the arguments (ρ_k, σ_k) solve the balance equations (3.2) to obtain the values taken by these functions.

4. EQUATIONS FOR THE COMPLETE SYSTEM

Let us return now to the full kanban system consisting of N cells in tandem, with parameters μ_k and $C_k (k = 1, 2, \ldots, N)$. Our aim is to approximate that system by a sequence of isolated kanban cells, where the streams of incoming jobs and demands for finished products at cell k are provided by cells $k - 1$ and $k + 1$, respectively (Figure 4.1). The parameters of these isolated cells must be such that the performance characteristics of the resulting sequence are as close as possible to those of the kanban system.

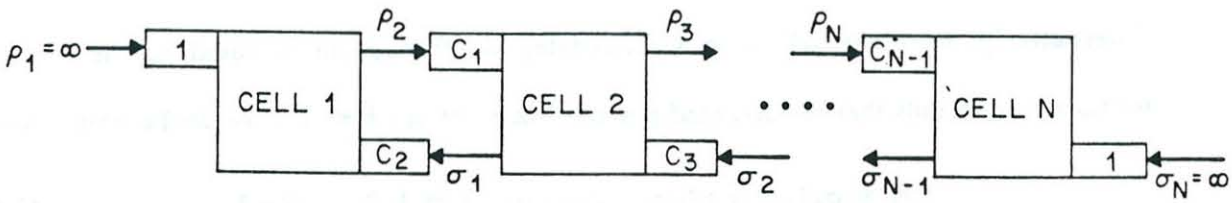

Figure 4.1. Model of production line constituted from models of isolated kanban cells.

The important facts in constituting the analysis of the production line from the isolated kanban cells are the following: the external arrivals buffer of the isolated cell k (see Figure 3.1) is in fact the output hopper of cell $k - 1$, and the external demands buffer of the isolated cell k is the bulletin board of cell $k + 1$. Indeed it was with this in mind that the external buffer capacities of cell $k (k = 1, 2, \ldots, N)$ were set to C_{k-1} and C_{k+1}, respectively, with the convention $C_0 = C_{N+1} = 1$.

Our present task is to choose appropriately the arrival rate vectors $\boldsymbol{\rho} = (\rho_2, \rho_3, \ldots, \rho_N)$, and $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_{N-1})$. It has already been decided that $\rho_1 = \sigma_N = \infty$. Several considerations may guide the choice of $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$. For instance, since the system is in steady-state, the throughputs of all the cells must be equal: $T_k = T_{k-1}, k = 2, 3, \ldots, N$. In view of relation (3.5), this requirement can be restated as

$$\rho_k[1 - P(I_k = -C_{k-1})] = T_{k-1} ,$$

or,

$$\rho_k = T_{k-1}/[1 - P(I_k = -C_{k-1})], \quad k = 2, 3, \ldots, N . \tag{4.1}$$

Alternatively, relation (3.6) implies that the equality of throughputs can be achieved by requiring that

$$\sigma_k = T_{k+1} / [1 - P(J_k = -C_{k+1})], \quad k = 1, 2, \ldots, N - 1 . \tag{4.2}$$

Another consideration arises from the fact that, in reality, the external arrivals buffer of the isolated cell k is the output hopper of cell $k - 1$. This consideration implies in particular that the following equations hold:

$$P(I_k = -C_{k-1}) = P(J_{k-1} = C_{k-1}), \quad k = 2, 3, \ldots, N .$$

Substituting this into (3.5) we get the equations

$$\rho_k = T_k / [1 - P(J_{k-1} = C_{k-1})], \quad k = 2, 3, \ldots, N . \tag{4.3}$$

Similarly, equating $P(J_k = -C_{k+1})$ with $P(I_{k+1} = C_{k+1})$, together with (3.6), yields

$$\sigma_k = T_k / [1 - P(I_{k+1} = C_{k+1})], \quad k = 1, 2, \ldots, N - 1 . \tag{4.4}$$

Now, consider the set of equations (4.1) and (4.4). Their right-hand sides are obtained from the solutions of the isolated cell models, and are therefore functions of the vectors $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$. Hence, denoting those (vector valued) functions by $\mathbf{f}$ and $\mathbf{g}$, respectively, (4.1) and (4.4) take the form

$$\begin{aligned} \boldsymbol{\rho} &= \mathbf{f}(\boldsymbol{\rho}, \boldsymbol{\sigma}) \\ \boldsymbol{\sigma} &= \mathbf{g}(\boldsymbol{\rho}, \boldsymbol{\sigma}) . \end{aligned} \tag{4.5}$$

This constitutes a set of fixed-point equations for the unknown vectors $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$. In fact, since (4.1) ensures that all throughputs are equal, (4.4) can be rewritten as

$$\sigma_k = T_{k+1} / [1 - P(I_{k+1} = C_{k+1})], \quad k = 1, 2, \ldots, N-1 . \tag{4.6}$$

This form of the function **g** is more convenient, both conceptually and numerically, because it enables σ_k to be expressed entirely in terms of the solution of cell $k+1$.

More than one set of fixed-point equations may be obtained. Note that if $T_k = T_{k-1}$, we obtain an expression for ρ_k entirely in terms of the solution of cell $k-1$:

$$\rho_k = T_{k-1} / [1 - P(J_{k-1} = C_{k-1})], \quad k = 2, 3, \ldots, N . \tag{4.7}$$

Ideally, we would like to have all three requirements satisfied: (i) $T_{k-1} = T_k$ $(k > 1)$, (ii) $P(I_k = -C_{k-1}) = P(J_{k-1} = C_{k-1})$ $(k > 1)$ *and* (iii) $P(J_k = -C_{k+1}) = P(I_{k+1} = C_{k+1})$ $(k < N)$. However, that ideal is unattainable because those three requirements yield a total of $3(N-1)$ equations for only $2(N-1)$ unknowns. The proposed approximation based on equations (4.1) and (4.6) achieves objectives (i) and (iii), while that based on (4.2) and (4.7) achieves (i) and (ii). Both approaches are reasonable, but it turns out that the latter is slightly more accurate than the former. Other approximations are also possible. For example, using (4.6) and (4.7) as the fixed-point equations yields quite acceptable results, although none of the above three requirements are exactly satisfied.

To solve numerically a set of fixed-point equations of the type (4.1), one would normally use an iterative procedure. Starting with some initial estimate of $\boldsymbol{\rho}$ and $\boldsymbol{\sigma}$, say $\boldsymbol{\rho}_0 = (\mu_1, \mu_2, \ldots, \mu_{N-1})$ and $\boldsymbol{\sigma}_0 = (\mu_2, \mu_3, \ldots, \mu_N)$, the iterations proceed according to

$$\boldsymbol{\rho}_{n+1} = \mathbf{f}(\boldsymbol{\rho}_n, \boldsymbol{\sigma}_n)$$

$$\boldsymbol{\sigma}_{n+1} = \mathbf{g}(\boldsymbol{\rho}_n, \boldsymbol{\sigma}_n) .$$

The rate of convergence depends, in general, on the particular approximation adopted.

The convergence of the iterative procedure to a solution does not, a priori, preclude the existence of other solutions. However, we conjecture that in the cases that we are

considering, the solution of the fixed-point equations is unique. That belief is based on certain properties of an isolated kanban cell, for which there are strong intuitive arguments, but no formal proof.

The existence of a solution also seems difficult to establish (except in one special case, where it follows from Brower's fixed-point theorem). We can report that, in the course of a rather extensive experimentation program, the iterations have never failed to converge.

To summarize, the results presented in the next section, if not from simulations, are obtained from the solution of the following system of fixed-point equations, (4.2) and (4.7), in the unknowns $\rho_2, \rho_3, \ldots, \rho_N$ and $\sigma_1, \sigma_2, \ldots, \sigma_{N-1}$:

$$\left.\begin{aligned} \rho_k &= \frac{T_{k-1}}{1 - P(J_{k-1} = C_{k-1})}, \quad k = 2, 3, \ldots, N, \\ \sigma_k &= \frac{T_{k+1}}{1 - P(J_k = -C_{k+1})}, \quad k = 1, 2, \ldots, N-1. \end{aligned}\right\}$$

In these equations it is tacitly understood that T_k, $P(J_k = C_k)$ and $P(J_k = -C_{k+1})$ are each function of ρ_k and σ_k; these functions have been defined in connection with the k^{th} isolated kanban cell and are evaluated by the procedure summarized at the end of the preceding section.

5. NUMERICAL RESULTS

Here we report on some experiments performed with our models. Figure 5.1 illustrates several features from the analytic solution of two separate production lines which differ only in that one has 1 card while the other has 2 cards in each cell; the two lines are identical in having 20 cells each and the service rate in every cell is identically 4. Among the features displayed in Figure 5.1 which prevail commonly is, first, the tendency of the inventory in individual cells to decrease with increasing distance from the head of

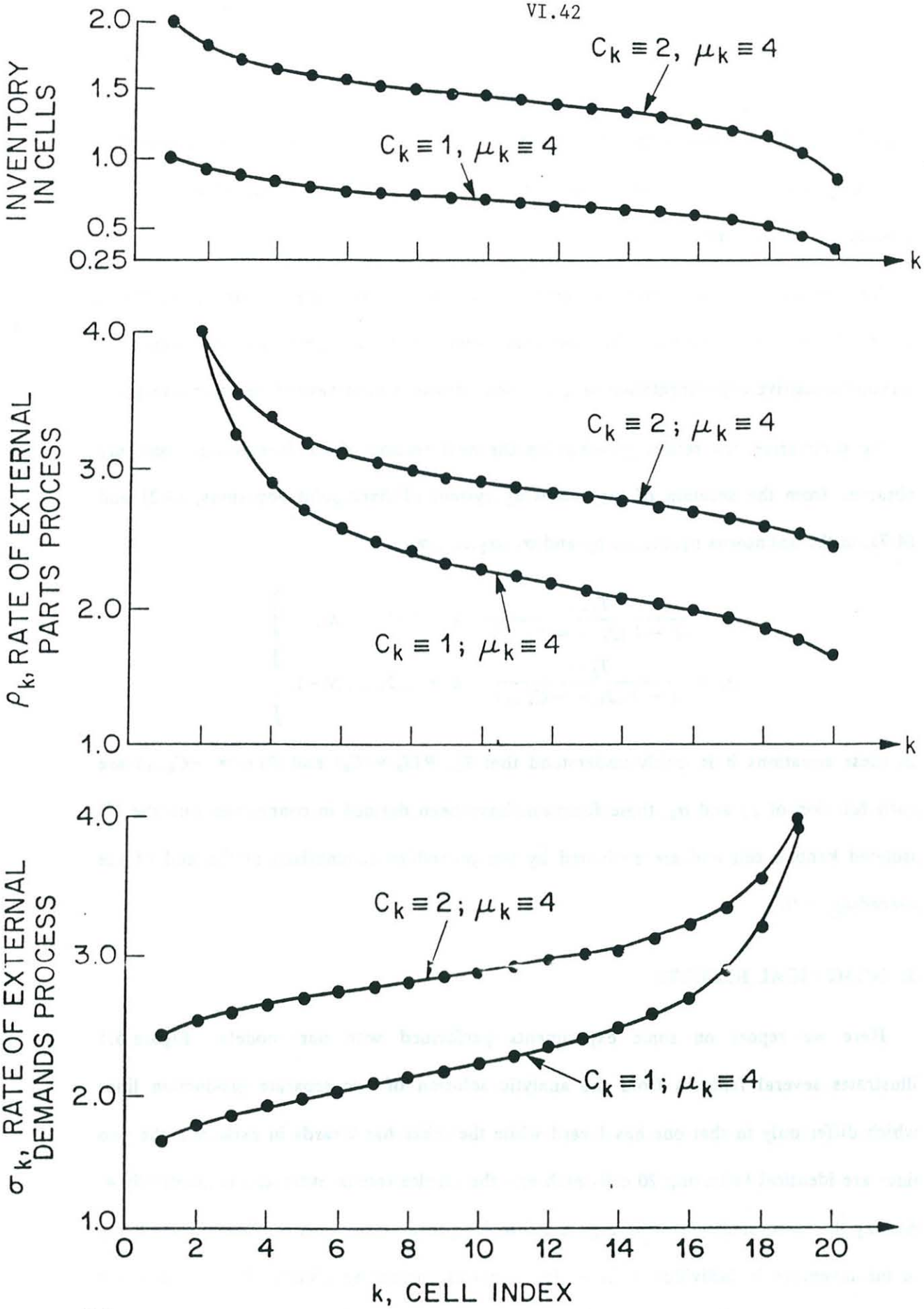

Figure 5.1 Results from computations for analytic (fixed-point) approximation for the kanban system in a production line with 20 cells. Throughput is 1.49 for $C_k \equiv 1$ and 2.30 for $C_k \equiv 2$. Total Inventory is 13.93 for $C_k \equiv 1$ and 29.01 for $C_k \equiv 2$.

the line. The numerical agreement between simulations and analysis for the individual cell inventories is rather good and in line with the agreement in total inventory which is examined in detail later. The analytic method generates solutions $\rho_2, \rho_3, \ldots, \rho_N$ which monotonically decrease, while $\sigma_1, \sigma_2, \ldots, \sigma_{N-1}$ monotonically increase. In uniform lines, i.e. where C_k and μ_k do not depend on k, a natural symmetry exists: $\sigma_{N-k} \approx \rho_{k+1}$, $k = 1, 2, \ldots, N-1$.

The number of iterations required for solving the fixed point equations in ρ and σ depends on N, the number of cells in the production line, and on the convergence or termination criterion. A typical criterion required that the L_∞-norm of the difference of two consecutive vectors be less than 10^{-5}. For this criterion, the number of iterations ranged from 37 for $N = 6$ to 70 for $N = 24$.

Figure 5.2 shows the analytic approximations, and the simulated values, of the throughput and the total inventory in a system of 6 identical kanban cells. The service rates are held constant while the storage capacity (number of cards in each cell) is varied. If the simulation results are treated as exact (the half width of the confidence intervals for the throughput values is 0.01) then the relative error of the approximation is seen to vary in the range 0.2%-5%. Moreover, the larger the number of cards per cell, the smaller that relative error becomes.

Figure 5.3 displays similar plots, except that this time the number of cards per cell is held constant, while the number of cells varies. Again there is a good agreement between approximations and simulations, with relative errors in the range 0.3%-7%. However, a reverse trend is observed: the larger the system, the larger the relative error. Note that the system throughput is an increasing function of the number of cards per cell, but a decreasing function of the number of cells. This is because the presence of more cards reduces the opportunities for blocking, whereas that of more cells increases them.

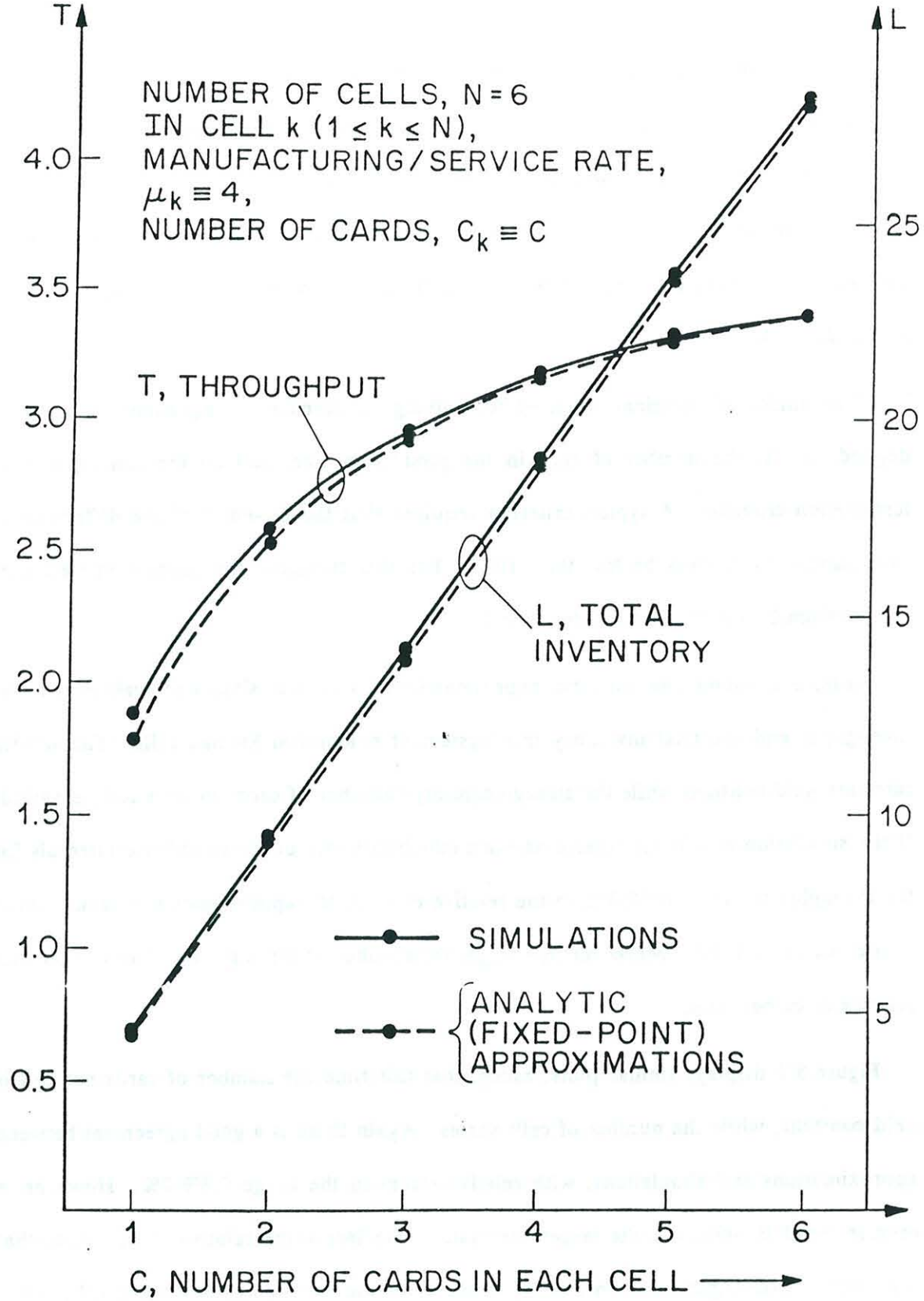

Figure 5.2 Throughput and Total Inventory of a production line, for various numbers of cards uniformly distributed over the cells in the line.

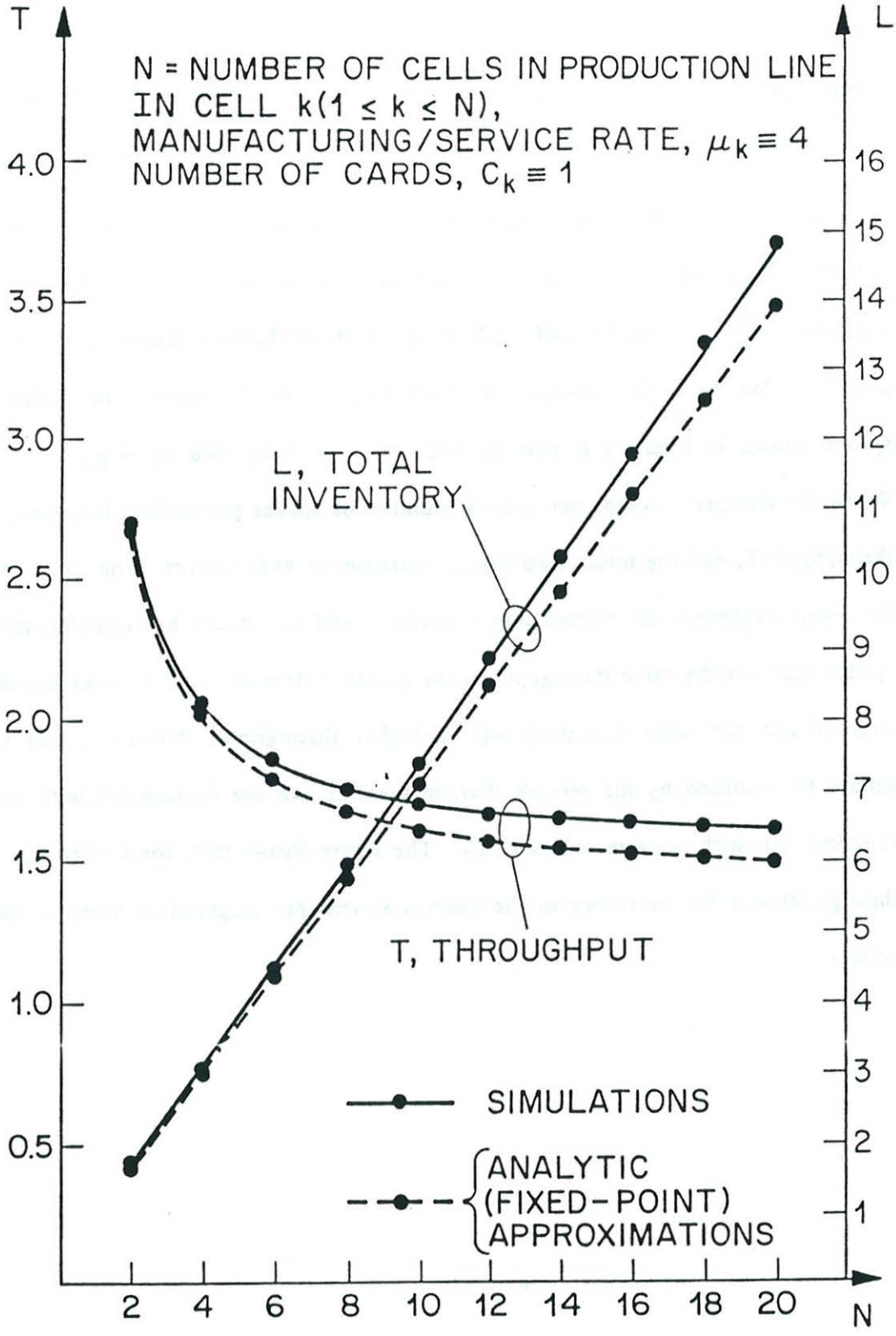

Figure 5.3 Throughput and Total Inventory of production lines with various numbers of cells.

Another point of interest is that, whereas the inventory is an almost linear function of the system size (number of cards or number of cells), the throughput is highly non-linear.

Figure 5.4 shows a comparison between a 10-cell kanban system and the corresponding sequence of ten buffers in tandem under transfer or manufacturing blocking. In this comparison C_k is the number of cards in cell k of the kanban system shown in Figure 1.1 and it is also the buffer capacity (inclusive of the server's position) of station k in the system shown in Figure 2.1; also $C_1 = C_2 = \cdots = C_N$ and $\mu_1 = \mu_2 = \cdots = \mu_N$. When the number of cards per cell (or number of spaces per buffer) increases, both the throughput, T, and the total inventory, L, increase, in each system. The curves plotted in the figure represent the relationship between T and L. It can be seen that the kanban system achieves the same throughput as the buffer system but with a lower inventory, or, alternatively, the same inventory with a higher throughput. However, that statement should be qualified by the remark that, in practice, we are dealing not with continuous functions but with discrete values of C_k. The figure shows that, for a fixed C_k, both the throughput and the inventory in the kanban system are larger than those in the buffer system.

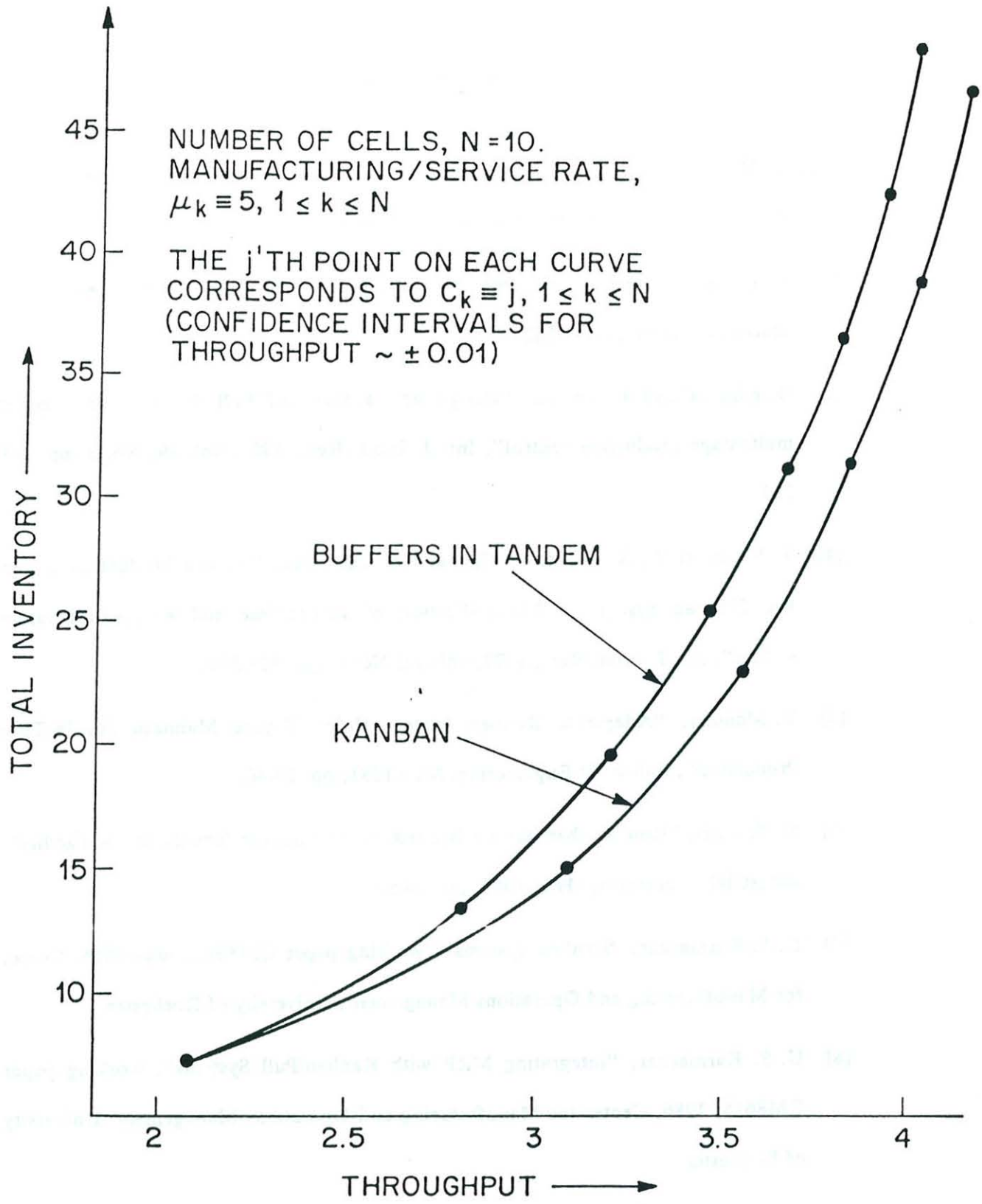

Figure 5.4 Throughput-inventory relationship compared for buffers in tandem (manufacturing blocking) and the kanban system in a production line with 10 cells.

REFERENCES

[1] Y. Monden, "*Toyota Production System*", Industrial Engineering and Management Press, Institute of Industrial Engineers, Atlanta, 1983.

[2] R. J. Schonberger, "*Japanese Manufacturing Techniques*", The Free Press, Macmillan, New York, 1982.

[3] O. Kimura and H. Terada, "Design and analysis of Pull System, a method of multi-stage production control", Int. J. Prod. Res., 1981, vol. 19, No. 3, pp. 241-253.

[4] Y. Sugimori, K. Kusunoki, F. Cho and S. Uchikawa, "Toyota Production System and Kanban System — Materialization of Just-in-time and Respect-for-human system", Int. J. Prod. Res., 1977, vol. 15, No. 6, pp. 553-564.

[5] Y. Monden, "Adaptable Kanban System Helps Toyota Maintain Just-In-Time Production", Industrial Engineering, May 1981, pp. 29-46.

[6] M. Sepehri, "How Kanban System is Used in an American Toyota Motor Facility", Industrial Engineering, Feb. 1985, pp. 50-56.

[7] U. S. Karmarkar, "Kanban Systems", working paper QM8612, June 1986, Center for Manufacturing and Operations Management, University of Rochester.

[8] U. S. Karmarkar, "Integrating MRP with Kanban/Pull Systems", working paper QM8615, 1986, Center for Manufacturing and Operations Management, University of Rochester.

[9] L. J. Krajewski, B. E. King, L. P. Ritzman and D. S. Wong, "Kanban, MRP and Shaping the Manufacturing Environment", Management Science, vol. 33, No. 1, January 1987, pp. 39-57.

[10] T. Altiok and S. Stidham, "A Note on Transfer Lines with Unreliable Machines, Random Processing Times and Finite Buffers", AIIE Trans. **14**, 1982, pp. 125-127.

[11] J. A. Buzacott and J. G. Shantikumar, "Models for Understanding Flexible Manufacturing Systems," A.I.I.E. Trans., vol. 12, 1980, pp. 339-350.

[12] E. Koenigsberg and J. Mamer, "The Analysis of Production Systems".

[13] R. Suri and G. W. Diehl, "A Variable Buffer-Size Model and its Use in Analyzing Closed Queueing Networks with Blocking," Mgt. Sci., vol. 32, No. 2, Feb. 1986, pp. 206-224.

[14] K. M. Rege, "Approximate Analysis of Serial Manufacturing Lines with Buffer Control", private communications, to appear in Large Scale Systems.

[15] B. A. Sevastyanov, "Influence of Storage Bin Capacity on the Average Standstill Time of a Production Line", Theory of Probability and Applications, vol. 7, 1962, pp. 429-438.

[16] F. S. Hillier and R. W. Boling, "Finite Queues in Series with Exponential or Erlang Service Times — a Numerical Approach," Oper. Res., vol. 15 No. 2, April 1967, pp. 254-265.

[17] S. B. Gershwin, "An Efficient Decomposition Method for the Approximate Evaluation of Tandem Queues with Finite Storage Space and Blocking", Operations Research, vol. 35, No. 2, 1987, pp. 291-305.

[18] H. G. Perros and T. Altiok, "Approximate Analysis of Open Networks of Queues with Blocking: Tandem Configuration", IEEE Trans. Soft. Eng., vol. SE-12, 1986, pp. 450-461.

[19] I. F. Akyildiz, "On the Exact and Approximate Throughput Analysis of Closed

Queueing Networks with Blocking", IEEE Trans. Soft. Eng., vol. 14, No. 1, Jan. 1988, pp. 62-70.

[20] A. Brandwajn and Y.-L.L. Jow, "An Approximation Method for Tandem Queues with Blocking", Oper. Res., **36**, No. 1, Jan. 1988, pp. 73-83.

[21] P. Tsoucas and J. Walrand, "On the Monotonicity of Throughput in Production Line Models", to appear in J. Appl. Prob.

DISCUSSION

Rapporteur: Rogério de Lemos

During the lecture Professor Littlewood asked what purpose the cards served in the Kanban policy. Dr Mitrani answered that they control the flow of parts into a cell in terms of how many parts and in what points. Mr McCue also asked if twice as much buffer space was required per cell. Dr Mitrani replied that the cards are a conceptual notion used to make the picture clearer - cards do not occupy space.

Mr Hughes asked whether the ranking of the minimal blocking (Kanban) policy as better than the transfer blocking policy, in terms of throughput, was based on any numerical simulation. Dr Mitrani answered negatively. He also pointed out that only the basic mathematical relations were used in the induction analysis, and stated that the use of numerical simulation methods are necessary only in the case where a quantified throughput is needed.

After the lecture, Professor Rogers asked if the transfer times between two nodes were taken into account. Dr Mitrani replied that there were two possibilities in considering these times, either you assume that they are part of the service of the node, or one could introduce artificial nodes between any two nodes and consider these nodes to have a very large buffer space. Still concerning this latter approach, no approximation was needed, and its only disadvantage was that the analysis should be realized with twice the number of nodes.

Professor Marie made the observation that constant service times were assumed to be a good assumption in manufacturing, and asked whether they could be used in such analysis. Dr Mitrani answered that as far as the sample path description equations are concerned, any type of distributive function could be considered.

Professor Littlewood asked Dr Mitrani for an intuitive explanation of the bell shape like curve that arises through the optimisation of space allocation. Dr Mitrani explained that the left hand extreme nodes only influence the nodes to the right of it, and as you move along, more dependencies will be introduced, more interactions will take place, and thus implies the need of more buffer space. Related to this issue Dr Holt asked if a symmetric bell shape could always be expected. Dr Mitrani replied saying that in this case symmetry was obtained because it was assumed equal service times for every node, but in general it not to be.

Professor Girault asked if such work has been applied in more complicated networks. Dr Mitrani answered negatively, and said that most of the examples presented during the lecture were based mainly on manufacturing and not on protocols, but such studies will be considered in the future. Dr Mitrani went on to say that both applications have some distinct characteristics, such as the transfer time between nodes; in manufacturing the transfer times are much smaller than the service times.