# DESIGN AND ANALYSIS OF MULTISTAGE TRANSFER LINES WITH INTERSTAGE BUFFERS

by

SERDAR ÖZER

B.S. in M.E., İ.T.Ü., 1983

Submitted to the Institute for Graduate Studies in Science and Engineering in Partial fulfillment of the requirements for the degree of

Master of Science

in

Industrial Engineering



Boğaziçi University

İstanbul - 1986

# ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my thesis advisor, Doç.Dr.Ali Rıza Kaylan, for his valuable guidance and his continuous support, encouragement and understanding throughout all phases of this study.

I also sincerely wish to thank Doç.Dr.M.Akif EYLER and Yrd.Doç.Dr.Vahan Kalenderoğlu for their valuable comments and suggestions.

I also would like to thank Mr.Türkay Peker, Technical Manager of Pfizer İlaçları A.Ş., and to Mr.Abdullah Kayral, Technical Manager of Fruko-Tamek Meyva Suları A.Ş., for their support and understanding throughout this study.

## ABSTRACT

Increasing importance of automation in manufacturing operations obliged the system designers to create large production systems that directly couple many stages of processing machinery.

Since high capital investment is necessary to realize such large production systems, the investment has to produce to its full potential. However, little flexibility is the character of complex manufacturing systems. Break downs of single stations and other system imbalances cause idle periods for both machinery and manpower which leads to high financial loss (opportunity cost).

Providing inventory banks between successive stages of the production line in order to decouple the stages and partially isolate overall system output from machine failures in any stage is one way of improving the line efficiency.

This study is concerned in modeling and analyzing multistage automatic transfer lines with finite interstage buffers. In the first part of the study an analytical model making use of the Markov chain approach is introduced. In the second part a simulation model of the multistage transfer line is proposed. The aim is to establish relationships between system parameters like failure and repair rates of the stages and storage capacities and the important system performance measures like production rate, average in-process inventories and forced-down times. At the end of the work the applications of the theoretical results to the real world problems will be discussed.

## ÖZET

Üretimde otomasyonun artan önemi, sistem tasarımcılarını üretimi yapan birden fazla makina/makina dizisini direkt olarak birbirine bağlayan büyük üretim sistemleri oluşturmaya zorlamıştır.

Bu tür büyük üretim sistemlerini gerçekleştirmek yüksek kapital yatırımı gerektirdiğinden, bu yatırımların tam kapasitede üretim yapmaları zorunludur. Ancak esnekliğin az olması bu tip kompleks imalat sistemlerinin bir özelliğidir. Herhangi bir makinanın arızalanarak durması ve diğer bazı sistem dengesizlikleri hem makina hem de insangücü için atıl zaman periyotları yaratmaktadır ki, bu da yüksek finansal kayıplara (fırsat maliyeti) yol açar. (Bu şartlarda tam olarak kullanılamayan yatırımın alternatif kullanım imkanlarından dolayı).

Bu durumda üretim hattının birbirini takip eden üretim istasyonları arasına bağımlılığı azaltacak ve genel sistem çıktısını herhangi bir üretim noktasındaki makina arızasından kısmen soyutlayacak şekilde stok kümeleri yerleştirmek sistemin verimliliğini arttırmak için bir çıkış yoludur.

Bu çalışma yukarıda sözü edilen, ara stok kullanan imalat hatlarının modellerinin kurulması ve çözümlenmesiyle ilgilidir. Çalışmanın ilk kısmında Markov zinciri yaklaşımına dayanan bir analitik model sunulmuştur. İkinci kısımda ise çok istasyonlu bir transfer hattının benzetim modeli açıklanmaktadır. Amaç bozulma ve onarma hızı ve stoklama kapasiteleri gibi sistem parametreleri ile üretim hızı, ortalama imalat içi stokları ve zorunlu duruş zamanları gibi önemli sistem performans ölçütleri arasındaki ilişkileri ortaya koymaktır. Çalışmanın sonunda ise teorik sonuçların gerçek hayat problemlerine uygulanması tartışılmaktadır.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# I. INTRODUCTION

## 1.1. CONCEPT OF "TRANSFER LINE" AND EFFICIENY PROBLEM

A transfer line can be defined as a number of automatic machining and inspection stations, in series, integrated into one system by a common transfer mechanism and a common control system (BUZACOTT, 2) (as exhibited in Figure 1.1).

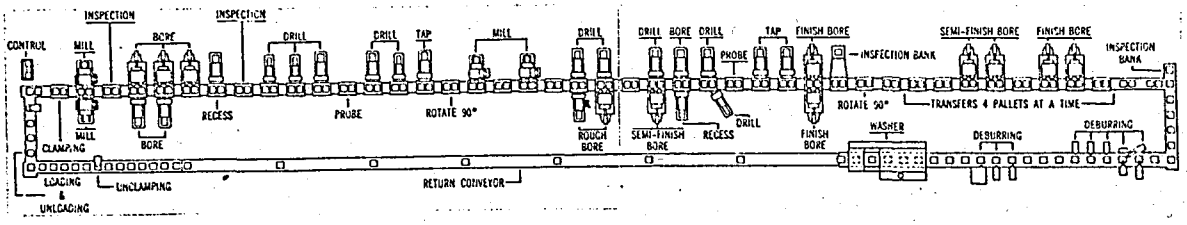


Figure 1.1. Overall layout of a transfer line

Materials and/or semi-finished parts enter the line from one end, flow through work-stations downstream the line to get processed and leave the line from other end as finished products. If one of the stations along the line fails, then all other stations in the line are forced to shut down and production is terminated. Valuable production time is lost, while the broken down station is repaired and as a result high financial cost is suffered.

To improve the efficiency, the line can be divided into a number of stages and inventory banks of semi-finished parts can be provided between these stages. With buffer stocks, other stages can continue working even if a station in one stage stops. The last stage in the live can continue working till all buffers between it and the stopped stage are empty and the first stage can continue working till all buffers between it and the stopped stage are full.

The improvement in the production rate, or efficiency of the line, will vary according to the allocation and the size of the interstage buffers. In the following study our concern will be the answer to the question, what the effect of given buffer capacities on system performance measures such as efficiency, average inventories and forceddown times is, if the system parameters are known.

## 1.2. Efficiency of the System

The purpose of providing buffers between the stages of a transfer line is to increase efficiency by means of storing and replenishing function of buffers. Therefore, it will be useful to define efficiency before going further. Efficiency of a transfer line is defined as the probability that the line produces a part in a cycle. This probability is equal to the expected value of the ratio of the number of cycles where the line produces items to the total number of cycles. Production rate of a transfer line is the expected number of completed parts produced by the system per unit time. In a balanced line

$$\text{Production rate} = \frac{\text{Efficiency}}{\text{Cycle time}}$$

Since in this study cycle time is taken as one unit time, terms "production rate" and "efficiency" are equivalents of each other.

$E(\infty)$ defines the efficiency in isolation of a line with buffer capacities equal to $\infty$. It is assumed that the buffer stocks will be such that they will never be exhausted during a stage breakdown, if the line has been operating long enough.

Accordingly, the isolated efficiency of any stage i is defined as

$$E_i(\infty) = \frac{\text{Mean uptime}_i}{\text{Mean uptime}_i + \text{Mean down time}_i}$$

$$= \frac{1/b(i)}{1/b(i) + 1/r(i)}$$

$$= \frac{1}{1 + b(i)/r(i)}$$

Where $b(i)$ and $r(i)$ are failure and repair rates of stage i respectively.

Efficiency in isolation of the line will be determined by the most inefficient stage in the system. Thus, $E(\infty)$ is defined as

$$E(\infty) = \min_{\text{all } i} \{E_i(\infty)\}$$

$$= \frac{}{1 + \max_{\text{all } i} \{\frac{b(i)}{r(i)}\}}$$

On the other hand, E(0) defines the efficiency of a line without buffers where for a k-machine line

$$E(0) = P\{(M_1 \text{ up}) \text{ and } (M_2 \text{ up}) \text{ and } \ldots \text{ and } (M_k \text{ up})\}$$

$$= P\{M_1 \text{ up}\} \cdot P\{M_2 \text{ up}\} \cdot \ldots P\{M_k \text{ up}\}$$

$$= \prod_{i=1}^{k} \frac{1}{1 + \frac{b(i)}{r(i)}}$$

In this case entire line will stop as soon as any stage breaks down.

Defining E(B) as line efficiency with buffers of any size it follows

$$E(0) \leqq E(B) \leqq E(\infty)$$

However the rate of convergence of E(B) to E(∞) by increasing storage capacities is strongly dependent on system parameters.

## 1.3. Economic Analysis of Improving Line Efficiency

As previously stated, improvement of line efficiency is very important for high volume machining systems. The opportunity cost of lost production due to idle periods caused by single machine failures is very high. Fortunately, there are some ways of reducing idle periods and increasing the utilization of machinery and manpower. However, every single alternative has its own disadvantages and extra expenses with it.

Improving reliability of the machinery through effective maintenance planning may be a solution. Substantial contribution is realized by introducing redundancy, i.e. stand by machines that enter the network in case of failures. However, this is often prohibitively expensive, especially in the case of systems involving very costly components.

In the case of providing buffer stocks at certain points along the transfer line, there also exists associated costs. Buffer storage may occupy valuable space and the workpiece kept in the storage may have high inventory holding cost. Handling the unit into and out of these inprocess inventory banks adds to the storage facility cost.

It is clear that production rate increases with increasing storage size, but this leads to increases in average in-process inventory levels too. Consequently increasing production rate brings extra expenses due to the storage space getting larger and due to the capital investment tied up to the higher in-process inventories.

Some of the cost types associated with the transfer line problem are shortly listed (Gershwin, Schick, 8):

(i) Cost of increasing the reliability of machines
(ii) Cost of providing materials handling equipment for each stage
(iii) Cost of providing storage capacity
(iv) Cost of repair of failed machines
(v) Cost of maintaining in-process inventory
(vi) Cost due to delay or processing time

There may also be some constraints affecting the economic analysis of interstage buffer storages that need to be taken into account. A limit on the total storage space or on

some or all of the interstage storages and a limit on the expected total number of units in the system or on the expected number of parts in certain storages are only some of the constraints affecting the analysis.

A careful economic analysis for changing the system configuration and installing interstage buffer is possible but if some assumptions are clearly stated then a simple break-even analysis will be very helpful to evaluate the conditions without to much computational effort.

The two most important assumptions are

(i) There is always a demand for the extra production gained by the line efficiency improvement
(ii) The unit variable cost of production is independent of production volume. Cost of capital ratio or the internal rate of return can be used as the interest rate for the computation of the holding cost.

The objective building an economic model is to compare profits made by improving the line efficiency and the cost of installing in-process inventory banks. The parameters of the model are

p: Unit price
v: Variable cost per unit
$v_j$: Variable cost of one unit semi-finished product in the j-th buffer storage
FC: Fixed cost per period
i: Interest rate
M: Number of stages

and the variables are

$\Delta P$: Positive increment in the production rate
$\bar{I}_j$: Average inventory level in the j-th buffer storage per cycle

In that case, the cost, the cost and profit equalities can be formulated as

$$[\text{Profit}] = \underbrace{(p-v)}_{\substack{\text{Marginal} \\ \text{contribution} \\ \text{per unit}}} \cdot \underbrace{[\Delta P * \text{period length in cycles}]}_{\substack{\text{Extra production} \\ \text{per period}}}$$

$$[\text{Cost}] = \underbrace{FC}_{\substack{\text{Fixed cost} \\ \text{per period}}} \underbrace{\left[\left(\sum_{j=1}^{M-1} \bar{I}_j * v_j\right) * \text{period length in cycles} * i \text{ per period}\right]}_{\substack{\text{Average holding} \\ \text{Cost per cycle} \\ \text{Total variable cost per period}}}$$

So given the $\bar{I}_j$'s, average in-process inventory levels per period, the expected P at the break-even point is formulated as

$$P \text{ at BEP} = \frac{FC + \left[\left(\sum_{j=1}^{M-1} \bar{I}_j * v_j\right) * \text{period length} * i \text{ per period}\right]}{(p-v) * \text{period lenght}}$$

As can be seen, even a very simple model requires knowledge about the system performance measures for decision making. Predictions of the performance measures can be obtained based on an analytical and/or a simulation model after having an identifying study on the system parameters. The management can decide to use inventory banks or not depending on the determined profitability.

## 1.4. Review of Past Research

There has been increasing interest in modelling the effect of inventory banks in automated flow lines such as

transfer lines, assembly machines and canning and packaging lines since fifties. The first analytical studies were the works of Vladzievski (1952, 1953) and Erpsher (1952) published in USSR in early fifties.

In a paper of Stanley B.Gershwin and Oded Berman, the related literature is divided into four classes.

The first group includes papers on systems without failures. The motion of discrete parts is modeled and the processing times are taken as random variables. The works of Hunt (1956), Hildebrand (1967 and 1968), Knott (1970), Kramer and Love (1970), Rao (1975) are included in this group.

The authors of the second group of papers assume deterministic processing times but random failures. Buzacott (1967a, 1967b, 1969), Gerschwin (1973), Sheskin (1974, 1976), Soyster and Toof (1976), Okamuro and Tamashino (1977), Ignall and Silver (1977), Buzacott and Hanifin (1978a), Schick and Gershwin (1978) fall into this class.

The third group of papers analyze models in which parts are not treated as discrete items. Instead the material to be processed is treated as continuous fluid. This includes the work of Finch, Vladzievski, and Sevastyarov (1962), described inKoenigsberg (1959) and Buzacott and Hanifin (1978a).

The fourth and last class is one in which individual parts are represented, machineoperation times are random, and failure and repair times are random. Only Buzacott (1972) and Gershwin and Berman (1979) belong to the fourth category.

## 1.5. Outline of Research and Contributions

The present thesis aims at devising analytic and simulation models for solving the problem of obtaining the production rate and other important performance measures of multistage transfer lines with finite interstage buffers. The results of the theoretical analysis are applied to real world systems.

In the second chapter of the study a Markov Chain model is introduced. The motion of discrete parts is modeled. The up and down times are taken as random variables but deterministic processing times are assumed. A computer program is prepared to compute the steady state values of the system states. Exact solutions for the steady state probabilities are obtained by solving a set of transition equations using Gauss Seidel iteration method.

Because of computer memory limitation the analytic model cannot analyze systems with large storage capacities. So in the third chapter a simulation model of the multistage transfer line is proposed. A second computer program is developed to simulate systems involving up to twenty stages and nineteen buffer stores with sufficiently large capacity limits. Approximate solutions of the system performance measures are discussed at the end of the chapter. The results are reviewed in three groups according to the structure of the system parameters and some conclusions are made.

The theory is applied to a bottling line, in beverage industry. The aim is to point out its short comings and weaknesses and to discuss model extensions to make it more applicable to actual situations.

In this study it is desired to obtain exact and approximate values for important line performance measures by means of analytical and numerical techniques given the characteristics of the machines and storages. The main target is to analyze the interactions between the elements of the system and the relations between various system parameters, so as to be able to determine the advantages of using buffer storages and their effect on system production rate.

# II. ANALYTICAL MODEL OF THE MULTISTAGE TRANSFER LINE

## 2.1. Problem Statement

### 2.1.1. Modeling the Transfer Line

#### 2.1.1.1. Description of the System

The system under study is a multistage transfer line. It consists of three unreliable stages which are separated by buffer storages of finite capacities. A stage is a section of the transfer line containing at least one, or more work stations. A workstation is defined as a stopping point where an operation(s) is (are) performed on the workpiece. If any one of the work stations fails, depending on the interdependence of the stations the related stage breaks down (see Figure 2.1).

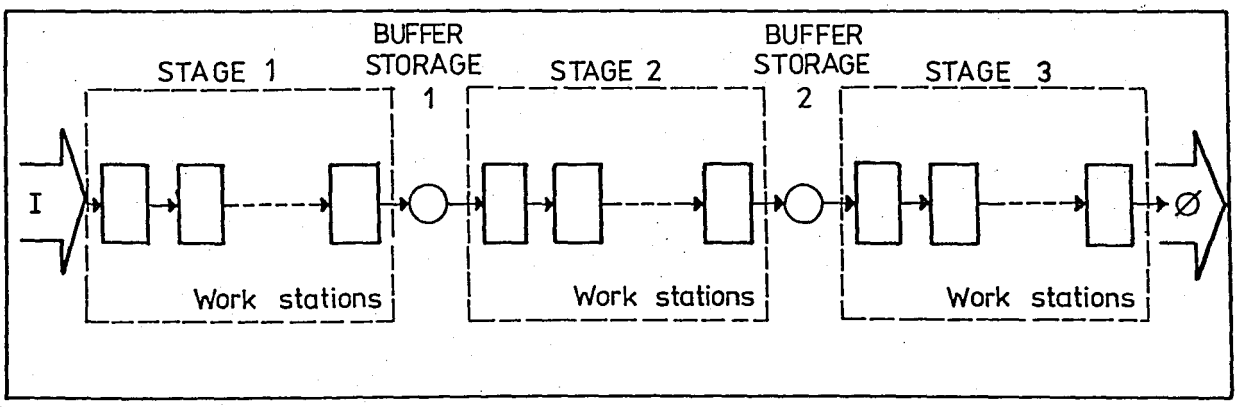


Figure 2.1. A three-stage transfer line with two buffers

Every workpiece enters the line from the first stage, goes through every stage along the line in the downstream direction and leaves the line as a finished item. Each straight uninterrupted part of the line is served by one transfer bar and the semifinished items are transfered from one station to the next by these bars. The movement of the semi-machined parts is performed simultaneously on all parts of the transfer line. The time interval between two subsequent transfers is called a cycle. In the following analysis unit production time including the transport time is taken as a time unit.

The function of the buffer storages is to absorb the negative effect of the single machine failures on the overall output of the line. The buffer storages are of finite capacity so during operations the number of units in any buffer store changes between 0 and the maximum capacity of the store. Each buffer can accumulate the output of the adjacent upstream stage until it is full and it can provide the adjacent downstream stage with semi-finished parts until it is empty.

2.1.1.2. State Space Formulation

Since it is desirable to develop a probabilistic model of unreliable transfer lines, it is necessary to formulate a state space for the model.

Production rate is dependent on the performance of every individual machine and the level of every buffer storage in the line. Before defining the system state it is necessary to define the operational status of the stages and the number of pieces in each storage.

The performance of a stage is defined by a set of four states:

1. Operating:

The stage is in working order and carries out its function. It is said to be "up".

2. Broken down or/and under repair:

The stage is broken down in that cycle or/and the repairmen are still working on it. The stage is said to be "down".

3. Forced down (starved):

The stage is in working order but it is not operational because there is no workpiece to operate. It is said to be "starved" or "idle".

4. Forced down (Blocked):

The stage is again in working order but since it cannot transfer its completed workpiece to the next stage or buffer storage it is forced down. This situation is called "blocked".

For a k-machine (stage) line the states are shown by variable $a_i$ defined as

$$a_i = \begin{cases} 1 & \text{if stage } i \text{ is up} \\ 0 & \text{" " " " down} \\ I & \text{" " " " idle} \\ B & \text{" " " " blocked} \end{cases} \qquad i=1,\ldots,k$$

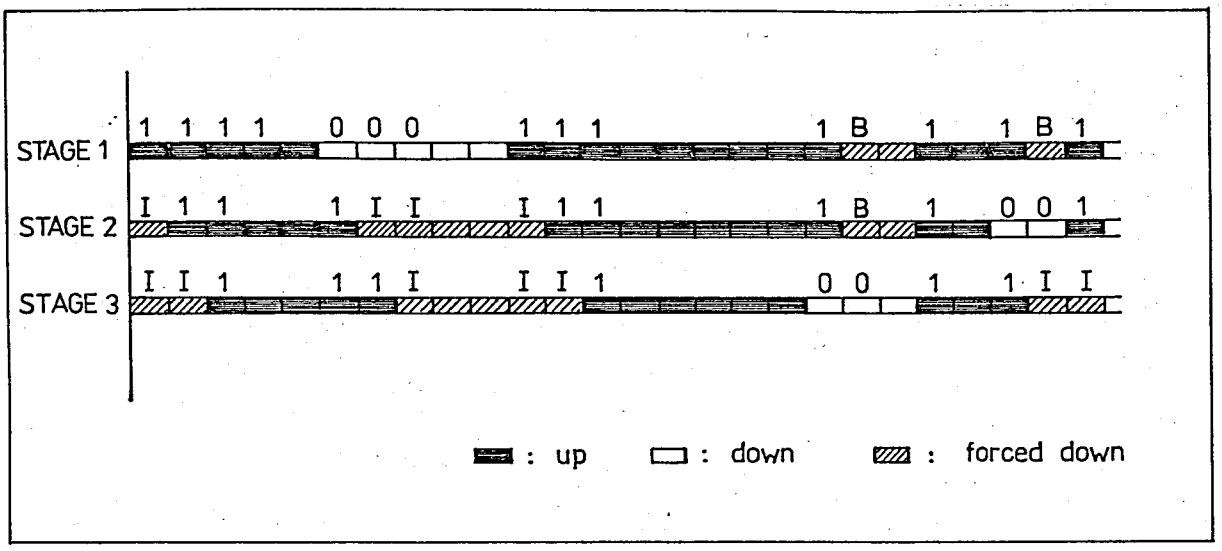


Figure 2.2. Diagram of state transitions for a three-stage line

The state of a buffer store at a certain time represents the number of units in the storage space. Nj is the maximum capacity for the j-th buffer storage, where j=1,...,k-1. The variable $n_j$ shows the level of the in-process inventory in the j-th storage so that.

$$0 \leqq n_j \leqq N_j \quad ; \quad j=1,\ldots,k-1$$

The state of the system is defined as the combination of the stage performance states and the inventory levels in the buffers. A system state at time t is formulated as

$$S(t) = \left[a_1(t)a_2(t)\ldots a_k(t);\ n_1(t)\ldots n_{k-1}(t)\right]$$

The system will be observed in constant time intervals, machining cycles, so that time is a discrete variable.

States describing the line behaviour are defined as follows

$U_p$ : If the last stage is operating the line is said to be up.

Down : If the last stage is stopped the line is said to be down.

Consequently the efficiency of the line is defined to be the probability of producing a finished piece within any given cycle, i.e. that the last stage is up in any given cycle and transfers its completed work piece to the finished item inventory at the end of the cycle. This probability can be taken as the ratio of time in which the line is up to total time.

As previously stated, cycle time is equal to a time unit. So efficiency and production rate are identical terms. Efficiency is denoted by E and always taken values in the range (0,1).

### 2.1.2. Assumptions of the Model

#### 2.1.2.1. Nature of Stoppages

In order to develop a predictive model of transfer lines it is necessary to examine failures closely. Failures can be categorized in different ways. The most reasonable classification is by extent or cause. Examining the extent of a failure, it can either be a single station failure, such as a tool failure, or a total line failure, such as a failure of the control system. If the cause of a failure is observed, failures can either be due to phenomenon which are operation dependent, such as tool wear, or time dependent, such as shift change.

An extensive study of the failures in two transfer lines at Chrysler Corporation showed that (Buzacott, Hanifin, 3).

5 % was due to operation dependent, total line failures
10 % was due to time dependent, total line failures
79 % was due to operation dependent, single station failures
6 % was due to time dependent, single station failures

Also supported by other studies, the conclusion can be made, that time dependent, single station failures in transfer lines are relatively insignificant. Moreover, total line failures prevent all stations in the line form working even if inventory banks are provided. Because the aim was to analyze the effect of the buffers on the efficiency, to add total line failures to the model would be meaningless. As a result, the following study focused only on operation dependent, single station failures. That means when a station is forced down by the failure of another station, the probability that it also breaks down is zero since the failure mechanisms are not operative.

Some other important point is that the failure of any station will result in stoppage of a group of adjacent station within that stage. A stage can be considered as a sub transferline without buffers. the failure of any station forces all other stations within the stage to shut down.

If the workstations in any stage A are $A_i$, $i=1,\ldots,m$ then

$$P(\text{stage A breaksdown}) = P(A_1 \text{ or } A_2 \text{ or } \ldots \text{ or } A_m \text{ breaks down})$$

Assuming that the up times of the stations are independent of each other and defining

$$P(A_i \text{ breaks down}) \equiv P(A_i) \; ; \; i=1,\ldots,m$$

for the sequence $A_1, A_2, \ldots, A_m$ of disjoint events (Çınlar, 5)

$$P\left(\bigcup_{i=1}^{m} A_i\right) = \sum_{i=1}^{m} P(A_i)$$

Hence, it comes out that

$$P(\text{Stage A breaks down}) = P(A_1 \text{ break down}) + \ldots + P(A_m \text{ breaks down})$$

2.1.2.2. Failure and Repair Distributions

The up (operating) and down (repair) time of a station or stage will have probability distributions. Because operation dependent failures are assumed the uptime of a stage (station) is defined as the difference between the clock time at which a failure occured and the clock time at which the previous repair ended minus the forced down times. Downtime of a stage (station) is defined as the duration of the repair action caused by any breakdown at that stage.

In the following study the up-and downtimes have a geometric distribution. Because the line is only observed at discrete intervals the operating and repair times are measured in number of cycles.

The probability that stage i breaks down in a cycle, given that it was operative at the end of the previous cycle, is called breakdown or failure rate of stage i (i=1,2,3). Denoted by b(i), failure rate of the i-th stage is a constant which is independent of the time since the last breakdown of the same stage. The mean time between failures, expected uptime, is equal to the reciprocal of the breakdown rate.

The probability that the repair is completed in a cycle, given that stage i(i=1,2,3) was broken down or under

repair during the previous cycle, is called repair rate of the i-th stage. Denoted by r(i), repair rate of the i-th stage is a constant again, regardless of how long the repairmen are working on the stage. The mean time to repair is equal to the reciprocal of the repair rate.

The memoryless property of the geometric distribution makes it possible to model the system as a Markov chain. Also most of the literature assumes that these distributions have the "memoryless" property for ease of modeling.

The validity of the geometric up-and downtime distribution is dependent on the reason for station break down. Most of the failures are due to stochastic events such as tool breakage. However tool wear would imply an increasing probability of stoppage with time, which is contradicting with the stationarity assumption. If there is a programme of tool replacement which is carried out between shifts or there are scheduled downtimes for the replacements then tool wear would not be a major cause of station breakdowns. Yet, when there is a very large number of possible causes of failure, in spite of scheduled replacements, to assume geometric uptime distribution is reasonable.

For a large set of possible causes of stoppages the duration of repairs takes different values. So also the geometric downtime assumption may not be far from the truth.

Transfer lines of two different industrial manufacturers are studied and one of them is represented in chapter 4. Also the records of the significant failures of one of the lines is examined. Data collection is realized for 8 hours each in 4 different days and by different shifts.

The results showed that there are a lot of different causes for the station stoppages. During the study no scheduled downtimes were met, but the records implied scheduled downtimes for revision of the whole line. Most of the failures were due to workpiece jams. Shut downs due to defective performance caused by inaccurate processing characteristics took the second place. Majority of the repairs were realized by the operators in short times. Only for important failures and scheduled replacements repairmen were needed and very few repairs lasted more than half an hour.

Examining the frequency distributions for up-and downtimes it can be clearly seen that these data sets are well represented by exponential distribution. As an example Figures 2.3 and 2.4 show the up and downtime frequency distributions of the fast operating end stage of one of the transfer lines.

2.1.2.3. Properties of Input/Output

The transfer line under study consists of three unreliable stages integrated in series and seperated by two buffers. It produces one kind of commodity. There is always an endless supply of parts avaliable for the head stage which is never starved. Parts are fed to the system one by one at a predetermined rate. An unlimited reservoir for finished items is assumed which implies that the end stage is never blocked.

In general, the assumption of infinite workpiece supply for the head stage will be justified for most industrial applications. Except the reasons of different nature, such as strikes, entire production lines seldom have to stop because of lack of raw material. There may be cases in which delays in reordering raw materials may cause a shortage of workpieces

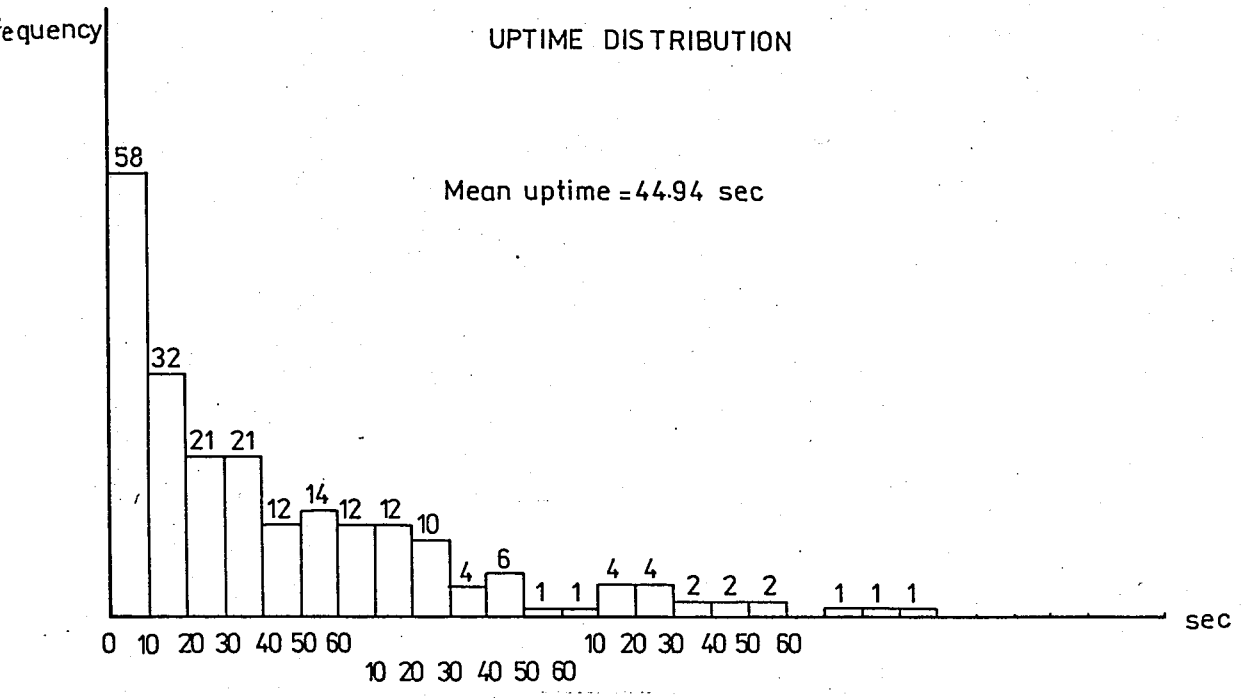


Figure 2.3. Uptime distribution of a real workstation

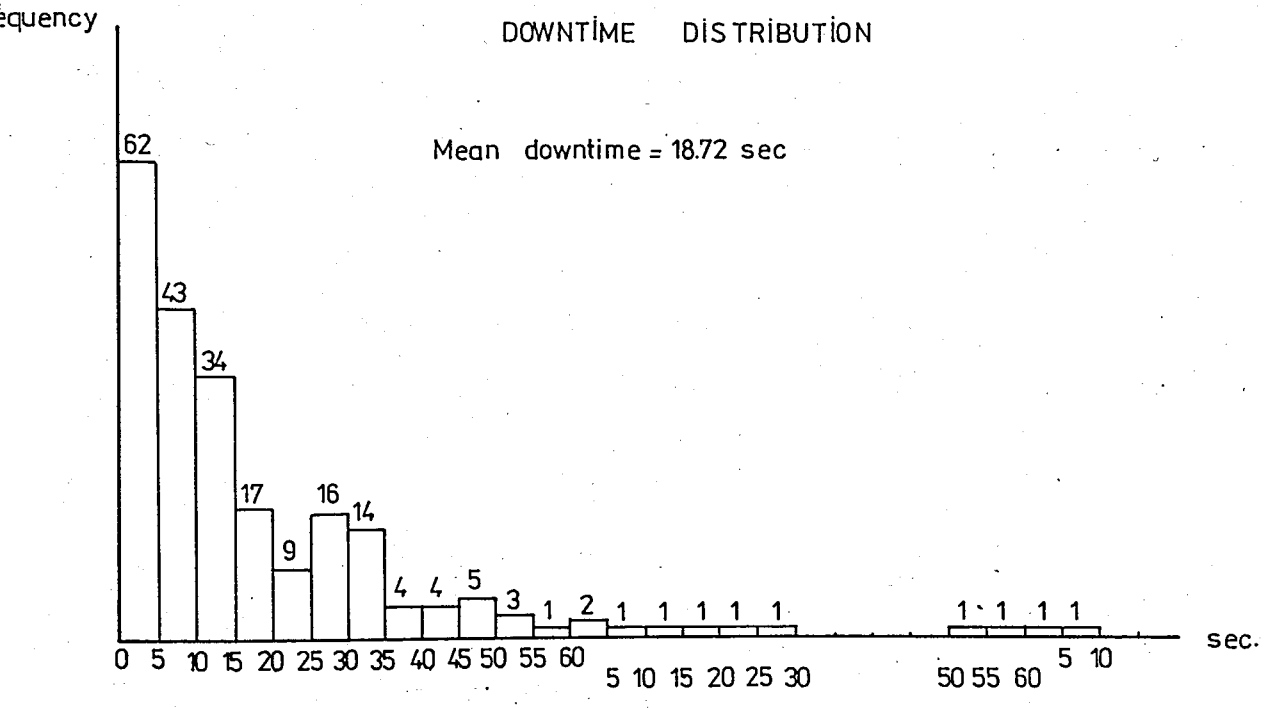


Figure 2.4. Downtime distribution of a real workstation

at the head stage. Moreover, at systems where the parts arrive and leave in batches, loading and unloading of batches may take some time. If the period of time in which the line is starved/blocked is negligible compared to the other times involved in the system, it may be ignored. An alternative solution is to model the temporary shortages of parts or bolckings due to unloading as failures of the related stages.

During operations semifinished parts are not destroyed or rejected due to inspection actions. However, when a stage is broken down its unit is scrapped. At systems where workpiece jam is a major cause of failures, this assumption may not be far from the truth.

2.1.2.4. Processing times of the Stages

Duration of time for processing one unit, is constant and equal for all stages i.e. the transfer line is balanced, which is not always true for real production lines.

At systems, where during manufacuring operations a certain percent of semimachined workpieces is removed or destroyed the production rate of workstations may decrease in downstream direction. There are also cases where the situation is reversed in order to avoid the blocking of upstream machines.

Some researchers claim that the assumption of deterministic service times is justifiable if service times do not deviate appreciably from the mean service time. The reason is that systems with interstage inventory banks run most often near boundaries. This implies that, in general, the number of units in any buffer i is either 1 or $N_i$. Accordingly, large deviations from the mean may starve certain machines and block others decreasing the line efficiency.

2.1.2.5. Rules for State Transitions

The system as a whole and and every individual part of it is observed at the end of each cycle. State transitions of stages and buffers occur at those certain points of time. Stages change their state, conditional an the level of the adjacent storages and after that the level of the in-process inventories is adjusted, conditional on the operational status of the adjacent stages. It can be seen that the stage and storage transitions depend only on the adjacent stage and storage states, and do not depend on the states of stages and storages further away.

Any stage, except the first one, is starved if the upstream stage was down or idle at the end of the previous cycle and the upstream storage is empty. Accordingly any stage, except the last one, is blocked if the downstream stage is down or blocked and the downstream storage is full. The semifinished part waits within the stage until it can be transfered to the next stage or storage. No stage can fail in its forced down status.

In contrast with some past models, more than one stage can fail within the same cycle. It is assumed that there are enough repairmen and equipment so that the repair probabilities are not affected when more than one machine are down simultaneously.

The level of a buffer store changes only by one unit or it stays at the same value at the end of each new cycle. If both stages, stage i and i+1 are up in a cycle the storage level of the i-th buffer does not change, because the part processed at the i-th stage is transfered to the next stage. However, if stage i has been up and stage i+1 has been inoperational within a cycle the increment at the i-th storage level is +1 and -1 at the reversed case.

2.1.2.6. Steady State Assumption

As the system is represented by a probabilistic model it is possible to determine a stationary probability distribution for the system states in steady state. Although a stochastic system is never at rest, after running long enough all effects of start-up transients will vanish. The claim is that knowledge of the initial condition of the system does not give any information of the present state of the system. While the system does not give any information of the present state of the system. While the system is still fluctuating the average performance of the system approaches the steady state values, assuming that the probabilistic model of the system is stationery.

2.2. Formulation of the Markov Chain Model

2.2.1. The Markovian Assumption and Some Basic Properties

The stochastic process $X=\{X_n;\ n\ N\}$ is called a Markov chain provided that

$$P\{X_{n+1} = j \mid X_o,\ldots,X_n\} = P\{X_{n+1} = j \mid X_n\}$$

for all $j\epsilon E$ and $n\epsilon N$, where E is a countable set and called the state space of the process X (Çınlar, 5). Thus defining the state of the system at time t as s(t)

$$P\{s(t+1)\mid s(t-z),\ldots,s(t-1),s(t)\} = P\{s(t+1)\mid s(t)\}$$

The formulation above implies that the transition probability at a certain time t depends only on the state occupied at that time and it is independent of past history of transitions. This means that the transition from state s(t) to state s(t+1) is independent of how the system has come to

state s(t) which shows the memoryless property of the Markov process.

Defining $t_{ij}$ as the state transition probability from state i at time t to state j at time t+1

$$t_{ij} \equiv P\{s(t+1) = j \mid s(t) = i\} \; ; \text{ for all } i,j \in E \text{ and } t \in N$$

$T^T$ is called then a Markov matrix over E provided that

(i) for any $i,j \in E$, $t_{ij} \geq 0$, and

(ii) for each $i \in E$, $\sum_{j \in E} t_{ij} = 1$

If there are K states then

$$T^T = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{1k} \\ t_{21} & t_{22} & t_{23} & \cdots & t_{2k} \\ \cdots & & & & \\ t_{k1} & t_{k2} & t_{k3} & \cdots & t_{kk} \end{bmatrix}$$

At any time t, the probabilities that the system is in state i=1,...,k may be represented as a state probability vector defined as

$$\underset{\sim}{P}(t) = \begin{bmatrix} P_1(t) \\ P_2(t) \\ \vdots \\ P_k(t) \end{bmatrix} \equiv \begin{bmatrix} P\{S(t) = 1\} \\ P\{S(t) = 2\} \\ \\ P\{S(t) = K\} \end{bmatrix}$$

Where

$$\sum_{i=1}^{K} P_i(t) = 1$$

Then the state probability vector at time t+1 is given by the difference equation system

$$\underset{\sim}{P}(t+1) = T\,\underset{\sim}{P}(t)$$

Setting t equal to 0 and applying the above formula recursively it comes out that

$$\underset{\sim}{P}(t) = T^{t}\underset{\sim}{P}(0)$$

where $\underset{\sim}{P}(0)$ is a given initial probability vector. Defining $T^{t}$ as L(t), the chain is termed ergodic if the limit

$$\lim_{t\to\infty} L(t) = L$$

exists and as a result, if the steady state probability vector is defined as

$$\underset{\sim}{P} = L\underset{\sim}{P}(0)$$

then it is independent of the value of the initial state probability vector $\underset{\sim}{P}(0)$.

As t goes to infinity both vectors $\underset{\sim}{P}(t)$ and $\underset{\sim}{P}(t+1)$ converge to $\underset{\sim}{P}$ so that in steady state

$$\underset{\sim}{P} = T\,\underset{\sim}{P}$$

Solving that system of linear equations the steady state values of the system states can be obtained.

## 2.2.2. Transfer Line as a Markov Chain

Memorylessness of failure and repair distribution of the workstations makes it possible to model the system under study as a Markov Chain. Given the state space for the system states and failure and repair rates for each stage, state transition probabilities can be easily obtained.

As stated before, if the Markov process is normal and the number of system states is equal to K, there exists stationary probabilities $P = [P_1 P_2 \ldots P_K]$ of system states $S_1$, $S_2, \ldots, S_K$ given by

$$P_j = \sum_{i=1}^{K} P_i t_{ij} \quad ; \quad j=1,\ldots,K$$

$$\sum_{j=1}^{K} P_j = 1$$

Showing that the chain is ergodic the steady state probabilities for the system states can be computed by solving a system of linear equations. The solution of the system will be used in evaluating the line's performance.

## 2.2.3. Ergodicity of the Process

Property of ergodicity is the necessary condition for obtaining a unique, exact solution of the system of transition equations. For the existance of ergodicity some conditions are defined.

If the system contains only one final (recurrent) aperiodic closed communicating class it is sufficient for ergodicity, where a closed class is defined as a set of states C such that no state outside C can be reached from any state inside C.

Two states are said to communicate if each can be reached from the other.

A closed communicating class is defined as a closed class in which all pairs of states communicate.

A final (recurrent) class is one that includes no transient states where a transient state is defined as a state which cannot be reached once the system leaves it.

A process is periodic if a state can be reached from itself in d, 2d, 3d,...,nd,... trials. If d=1 only, the process is termed aperiodic. If there is at least one state in a final class which is has a self-loop, it is sufficient for its aperiodicity.

2.2.4. Computation of Steady State Probabilities

Given that the process is ergodic the problem reduces to solving the system of linear transition equations given as

$$\underset{\sim}{P} = T\,\underset{\sim}{P}$$

or

$$(T-I)\,\underset{\sim}{P} = 0$$

$$\sum_{i=1}^{K} P_i = 1$$

where $\underset{\sim}{P} \neq 0$ and (T-I) is a singular matrix of rank (K-1)

Defining a vector

$$e^T = [111 \ldots 1]$$

the normality equation can be rewritten as

$$\underset{\sim}{e}^T \underset{\sim}{P} = 1$$

At this step the system has m unknowns and m+1 equations. Deleting one of the rows of (T-I) and substituting for this row the vector $\underset{\sim}{e}^T$ and defining the right-hand side vector as $\underset{\sim}{b}^T = |0..010..0|$ where the 1 entry corresponds to the location of $\underset{\sim}{e}^T$ in T, it follows that

$$T^* \underset{\sim}{P} = \underset{\sim}{b}$$

Here T* represents the adjusted version of (T-I) and has full rank.

For the system under study the size of the problem gets larger if the capacity of the buffer storages is increased. Table 2.1 shows the number of system states, which is equal to the number of unknowns, for various sizes of inventory banks. The possible system states for a k-stage line are generated by a computer program given in the Appendix. The formula for finding the number of system states given storage capacities $N_i$ , i=1,2, is given as

$$\left\lceil \begin{matrix}\text{number of} \\ \text{system states}\end{matrix} \right\rceil = 8\, N_1 N_2 + 12\, N_1 + 12\, N_2 + 18$$

or for symmetric storage capacity case

$$\left\lceil \begin{matrix}\text{number of} \\ \text{system states}\end{matrix} \right\rceil = 8\, N^2 + 24\, N + 18$$

TABLE 2.1. Number of system states for various storage capacities

| $N_1 = N_2 = N$ | of states |
|---|---|
| 0 | 18 |
| 1 | 50 |
| 2 | 98 |
| 3 | 162 |
| 4 | 242 |
| 5 | 338 |

Solving the system for large storage capacities would involve an extremely large amount of computation and computer memory. It is therefore necessary to fully exploit the sparsity and structure of T*.

A matrix is termed sparse if the ratio of the number of nonzero entries to the total number of entries is small. In our case sparsity follows from the fact that many transition probabilities are equal to 0. The special structure of the transition matrix follows from the constrained transition conditions of the buffer storages. During a single transition, storage levels can each change by a maximum of 1 and except very few cases adjacent storages cannot change in the same direction, i.e. they cannot both gain or lose a piece within a single cycle.

The advantage of the sparsity of the transition matrix is that only storing the non zero elements and their coordinates extremely large matrices can be represented with relatively small arrays.

To solve the system of linear equations Gauss Seidel Iteration Method is utilized. The algorithm is as follows:

Step 0 : Initialization

Step 1 : $$P_i(t+1) = \frac{\left(-\sum_{i=1}^{i-1} t_{ij} P_j(t+1) - \sum_{j=i+1}^{M} t_{ij} P_j(t)\right) + b_i}{t_i} \quad \text{for all } i$$

Step 2 : if $$\left|\frac{P_i(t+1) - P_i(t)}{P_i(t+1)}\right| < \varepsilon \quad \text{for all } i \quad \text{STOP}$$

otherwise GO TO 1

Major limitation of the algorithm is the computation necessary for the convergence of $\underset{\sim}{P}(t)$ to $\underset{\sim}{P}$. Rate of convergence is determined by two factors. One is the accuracy of the initial guess and the other is the second largest eigenvalue of the transition matrix. The eigenvalues of T are dependent on the system parameters. Since the computation of the eigenvalues of a matrix as large as T is far from trivial the estimation of the rate of convergence of the algorithm is not possible. However, the initial guess can be improved significantly, by making certain observations and giving appropriate initial values to certain system states.

## 2.3. Computation of System Performance Measures

### 2.3.1. Transfer Line Without Buffers

The purpose of this study is to analyse the effect of buffer stocks on system performance, but before doing that it will be useful to see the behaviour of a system without buffers.

For the system without buffers, or equivalently with buffers of D capacity, there exists eighteen system states as portrayed in Figure 2.5. Transition matrix for the system is presented in Figure 2.6.

$\bar{b}(i)$ is defined as the probability that any stage i does not fail within a cycle, given that it has been up at the end of the previous cycle, which is equal to 1-b(i). Similarly, $\bar{r}(i)$ is the probability that any stage i is still under repair within a cycle, given that it has been down at the end of the previous cycle, which is equal to 1-r(i).

Production rate is defined as

P {System discharges a part in acycle} $\equiv$ P {System is up within a cycle and does not fail at the end of the cycle}

= P {System is up within a cycle}. P {System does not fail at the end of the cycle}

= P {Last stage is up within a cycle}. P {Last stage does not fail at the end of the cycle}

$$= \left(\sum_i P_i\right).(1-b(3))$$

Where $i \in S$ and S is the set of system states where last stage is up.

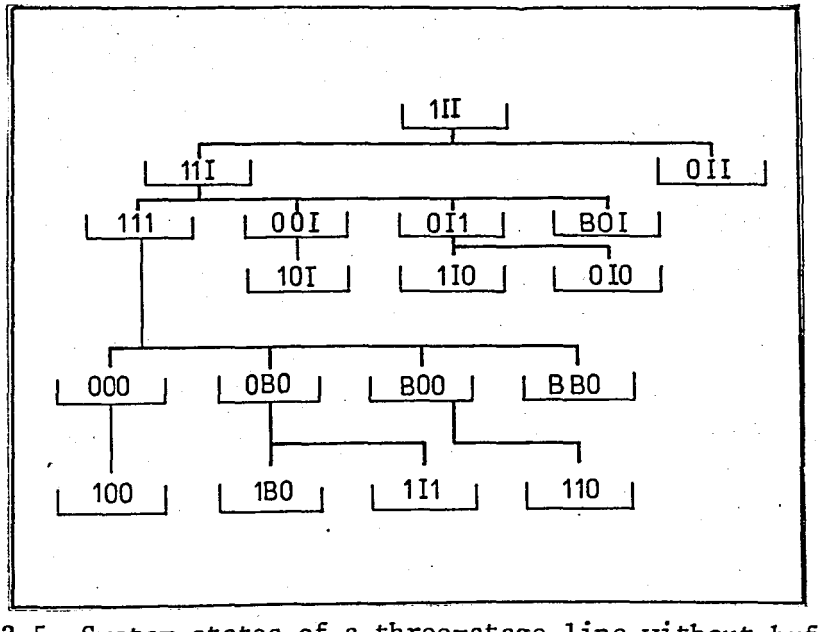


Figure 2.5. System states of a three-stage line without buffers

| S(t) \ S(t+1) | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $S_5$ | $S_6$ | $S_7$ | $S_8$ | $S_9$ | $S_{10}$ | $S_{11}$ | $S_{12}$ | $S_{13}$ | $S_{14}$ | $S_{15}$ | $S_{16}$ | $S_{17}$ | $S_{18}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (111) $S_1$ | $\bar{b}_1\bar{b}_2\bar{b}_3$ | | | | | | | | | $b_1b_2b_3$ | $b_1b_2\bar{b}_3$ | $b_1\bar{b}_2b_3$ | $b_1\bar{b}_2\bar{b}_3$ | | | $\bar{b}_1b_2b_3$ | $\bar{b}_1b_2\bar{b}_3$ | $\bar{b}_1\bar{b}_2b_3$ |
| (110) $S_2$ | $\bar{b}_1\bar{b}_2r_3$ | | | | | | | | | $b_1b_2\bar{r}_3$ | $b_1b_2r_3$ | $b_1\bar{b}_2\bar{r}_3$ | $b_1\bar{b}_2r_3$ | | | $\bar{b}_1b_2\bar{r}_3$ | $\bar{b}_1b_2r_3$ | $\bar{b}_1\bar{b}_2\bar{r}_3$ |
| (11I) $S_3$ | $\bar{b}_1\bar{b}_2$ | | | | | | | | | | $b_1b_2$ | | $b_1\bar{b}_2$ | | | | $\bar{b}_1b_2$ | |
| (100) $S_4$ | | $\bar{b}_1r_2\bar{r}_3$ | $\bar{b}_1r_2r_3$ | | | | | | | $b_1\bar{r}_2\bar{r}_3$ | $b_1\bar{r}_2r_3$ | | | $b_1r_2\bar{r}_3$ | $b_1r_2r_3$ | $\bar{b}_1\bar{r}_2\bar{r}_3$ | $\bar{b}_1\bar{r}_2r_3$ | |
| (10I) $S_5$ | | | $\bar{b}_1r_2$ | | | | | | | | $b_1\bar{r}_2$ | | | | $b_1r_2$ | | $\bar{b}_1\bar{r}_2$ | |
| (1B0) $S_6$ | $\bar{b}_1r_3$ | | | | | | | | | | | $b_1\bar{r}_3$ | $b_1r_3$ | | | | | $\bar{b}_1\bar{r}_3$ |
| (1I1) $S_7$ | | $\bar{b}_1b_3$ | $\bar{b}_1\bar{b}_3$ | | | | | | | | | | | $b_1b_3$ | $b_1\bar{b}_3$ | | | |
| (1I0) $S_8$ | | $\bar{b}_1\bar{r}_3$ | $\bar{b}_1r_3$ | | | | | | | | | | | $b_1\bar{r}_3$ | $b_1r_3$ | | | |
| (1II) $S_9$ | | | $\bar{b}_1$ | | | | | | | | | | | | $b_1$ | | | |
| (000) $S_{10}$ | | | | $r_1\bar{r}_2\bar{r}_3$ | $r_1\bar{r}_2r_3$ | | | $r_1r_2\bar{r}_3$ | $r_1r_2r_3$ | $\bar{r}_1\bar{r}_2\bar{r}_3$ | $\bar{r}_1\bar{r}_2r_3$ | | | $\bar{r}_1r_2\bar{r}_3$ | $\bar{r}_1r_2r_3$ | | | |
| (00I) $S_{11}$ | | | | | $r_1\bar{r}_2$ | | | | $r_1r_2$ | | $\bar{r}_1\bar{r}_2$ | | | | $\bar{r}_1r_2$ | | | |
| (0B0) $S_{12}$ | | | | | | $r_1\bar{r}_3$ | $r_1r_3$ | | | | | $\bar{r}_1\bar{r}_3$ | $\bar{r}_1r_3$ | | | | | |
| (0I1) $S_{13}$ | | | | | | | | $r_1b_3$ | $r_1\bar{b}_3$ | | | | | $\bar{r}_1b_3$ | $\bar{r}_1\bar{b}_3$ | | | |
| (0I0) $S_{14}$ | | | | | | | | $r_1\bar{r}_3$ | $r_1r_3$ | | | | | $\bar{r}_1\bar{r}_3$ | $\bar{r}_1r_3$ | | | |
| (0II) $S_{15}$ | | | | | | | | | $r_1$ | | | | | | $\bar{r}_1$ | | | |
| (B00) $S_{16}$ | | $r_2\bar{r}_3$ | $r_2r_3$ | | | | | | | | | | | | | $\bar{r}_2\bar{r}_3$ | $\bar{r}_2r_3$ | |
| (B0I) $S_{17}$ | | | $r_2$ | | | | | | | | | | | | | | $\bar{r}_2$ | |
| (BB0) $S_{18}$ | $r_3$ | | | | | | | | | | | | | | | | | $\bar{r}_3$ |

Figure 2.6. Transition matrix of a three-stage line without buffers

Consequently the production rate of the system is defined as

$$P = (1-b(3)) \cdot [p_1 + p_7 + p_{13}]$$

where $p_1$, $p_7$, $p_{13}$ are the steady state probabilities corresponding to system states $S_1 = [111]$, $S_7 = [1I1]$ and $S_{13} = [0I1]$.

Solving a set of transition equations of size 18 for $=10^{-10}$ and various combinations of system parameters some predictions are made. Results for the 3-stage line without buffers are presented in Table 2.2.

TABLE 2.2. E(∞) and E(0) values for different combinations of system

| b(1) | b(2) | b(3) | r(1) | r(2) | r(3) | Isolated efficiency | Efficiency (computed) |
|---|---|---|---|---|---|---|---|
| .005 | .005 | .005 | .05 | .05 | .05 | .90909 | .75980 |
| .001 | .001 | .001 | .05 | .05 | .05 | .98039 | .94066 |
| .001 | .001 | .001 | .02 | .02 | .02 | .95238 | .86715 |
| .001 | .001 | .001 | .04 | .04 | .04 | .97561 | .92755 |
| .001 | .001 | .003 | .04 | .04 | .04 | .93023 | .88469 |
| .001 | .001 | .005 | .04 | .04 | .04 | .88889 | .84547 |
| .003 | .001 | .001 | .04 | .04 | .04 | .93063 | .88469 |
| .005 | .001 | .001 | .04 | .04 | .04 | .88889 | .84547 |
| .001 | .003 | .001 | .04 | .04 | .04 | .93023 | .88467 |
| .001 | .005 | .001 | .04 | .04 | .04 | .88569 | .84513 |
| .003 | .003 | .003 | .02 | .02 | .02 | .86957 | .68454 |
| .005 | .005 | .003 | .02 | .02 | .02 | .80000 | .59986 |
| .005 | .003 | .005 | .02 | .02 | .02 | .80000 | .59996 |
| .003 | .005 | .005 | .02 | .02 | .02 | .80000 | .59986 |
| .005 | .005 | .005 | .02 | .02 | .02 | .80000 | .56493 |
| .05 | .03 | .01 | .1 | .1 | .1 | .66667 | .48996 |
| .01 | .03 | .05 | .1 | .1 | .1 | .66667 | .49000 |
| .0015 | .0015 | .0015 | .05 | .03 | .01 | .86957 | .80981 |
| .0015 | .0015 | .0015 | .01 | .03 | .05 | .96957 | .80957 |

Examining the results for lines with identical failure and repair rates. It can be seen that E(0) differs significantly from from the theoretical value of $E(\infty)$. The ratio of computed efficiency to isolated efficiency increases when the ratio of failure rate to repair rate dicreases. This shows that if mean repair time is relatively small compered with mean uptime, the effect of buffer stocks on the improvement of the overall output of the system is not significant.

Holding repair rates constant, dicreasing (increasing) failure rates results in higher (lower) line efficiency as expected. Line performance changes in the same way for cases of constant failure rates and variying repair rates.

For systems with unidentical failure and repair rates line performance takes different values with changing combination of system parameters. Assuming identical repair rates, if one (two) of the system components has a higher failure rate $b_1$ and the other two (remaining one) components have lower and identical failure rates $b_2$ then production rate of the system falls into the range

$$P_{b(1)=b(2)=b(3)=b_1} < P_{b_1,b_2} < P_{b(1)=b(2)=b(3)=b_2}$$

If failure rates (repair rates) change in the down-stream or upstream direction following results are obtained.

For identical repair rates production rate does not differ much if the first or last stage is most efficienct, because the middle stage has a dominating effect on the line efficiency. If failure rates are identical, a higher repair probability of the first stage yields a higher production rate than the case where repair rate of the first stage is lower leaving the following stages idle for longer times during a breakdown.

## 2.3.2. Transfer Line With Interstage Buffers

For the 3-stage transfer line provided with interstage buffers state space of the system and accordingly the size of the problem gets larger with increasing storage capacities. For ease of modeling it is assumed that size of the inventory banks are identical and equal to N. For identical storage size case there are $8N^2+24N+18$ possible system states.

| SYSTEM STATES | | NUMBER OF STATES |
|---|---|---|
| [ 111;O O ] --> S 1 | | |
| . | | |
| . | | |
| [ 111;N N ] --> S N^2+2N+1 | ) | N^2+2N+1 |
| [ 110;O O ] --> S N^2+2N+2 | | |
| . | | |
| . | | |
| [ 110;N N ] --> S 2N^2+4N+2 | ) | N^2+2N+1 |
| [ 11I;O O ] --> S 2N^2+4N+3 | | |
| . | | |
| . | | |
| [ 11I;N O ] --> S 2N^2+5N+3 | ) | N+1 |
| [ 101;O O ] --> S 2N^2+5N+4 | | |
| . | | |
| . | | |
| [ 101;N N-1] --> S 3N^2+6N+3 | ) | N^2+N |
| [ 100;O O ] --> S 3N^2+6N+4 | | |
| . | | |
| . | | |
| [ 100;N N ] --> S 4N^2+8N+4 | ) | N^2+2N+1 |
| [ 10I;O O ] --> S 4N^2+8N+5 | | |
| . | | |
| . | | |
| [ 10I;N O ] --> S 4N^2+9N+5 | ) | N+1 |
| [ 1B0;O N ] --> S 4N^2+9N+6 | | |
| . | | |
| . | | |
| [ 1B0;N N ] --> S 4N^2+10N+6 | ) | N+1 |
| [ 1I1;O O ] --> S 4N^2+10N+7 | | |
| . | | |
| . | | |
| [ 1I1;O N ] --> S 4N^2+11N+7 | ) | N+1 |
| [ 1I0;O O ] --> S 4N^2+11N+8 | | |
| . | | |
| . | | |
| [ 1I0;O N ] --> S 4N^2+12N+8 | ) | N+1 |

```
[ 1II;O O  ] --> S 4N^2+12N+9    } 1
[ O11;O O  ] --> S 4N^2+12N+10
      .
      .
[ O11;N-1 N] --> S 5N^2+13N+9    } N^2+N
[ O10;O O  ] --> S 5N^2+13N+10
      .
      .
[ O10;N-1 N] --> S 6N^2+14N+9    } N^2+N
[ O1I;O O  ] --> S 6N^2+14N+10
      .
      .
[ O1I;N-1 O] --> S 6N^2+15N+9    } N
[ OO1;O O  ] --> S 6N^2+15N+10
      .
      .
[ OO1;N N-1] --> S 7N^2+16N+9    } N^2+N
[ OOO;O O  ] --> S 7N^2+16N+10
      .
      .
[ OOO;N N  ] --> S 8N^2+18N+10   } N^2+2N+1
[ OOI;O O  ] --> S 8N^2+18N+11
      .
      .
[ OOI;N O  ] --> S 8N^2+19N+11   } N+1
[ OBO;O N  ] --> S 8N^2+19N+12
      .
      .
[ OBO;N N  ] --> S 8N^2+20N+12   } N+1
[ OI1;O O  ] --> S 8N^2+20N+13
      .
      .
[ OI1;O N  ] --> S 8N^2+21N+13   } N+1
[ OIO;O O  ] --> S 8N^2+21N+14
      .
      .
[ OIO;O N  ] --> S 8N^2+22N+14   } N+1
[ OII;O O  ] --> S 8N^2+22N+15   } 1
[ BO1;N O  ] --> S 8N^2+22N+16
      .
      .
[ BO1;N N-1] --> S 8N^2+23N+15   } N
[ BOO;N O  ] --> S 8N^2+23N+16
      .
      .
[ BOO;N N  ] --> S 8N^2+24N+16   } N+1
[ BOI;N O  ] --> S 8N^2+24N+17   } 1
[ BBO;N N  ] --> S 8N^2+24N+18   } 1
```

Examining the system states it can be seen that there are only 24 different combinations of stage states. System state space is generated by combining these operational status combinations with corresponding storage states. Yet, general transition matrix for storage capacity N can be divided into 24x24 sub-transition matrices, most of which are zero matrices. Making use of this special structure of T a subroutine is designed for storing T into the computer memory. After assigning the transition probabilities the problem reduces again to solve a set of linear transition equations.

The computed steady-state probabilities of the system states are used to predict the expected production rate and the mean number of units in the storage areas. Production rate of the system is defined as

$$P=(1-b(3))\left[\sum_{i=1}^{N^2+2N+1} P_i + \sum_{i=1}^{N^2+N} P2N^2+5N+3+i + \sum_{i=1}^{N+1} P4N^2+10N+6+i \right.$$

$$+ \sum_{i=1}^{N^2+N} P4N^2+12N+9+i + \sum_{i=1}^{N^2+N} P6N^2+15N+9+i$$

$$\left. + \sum_{i=1}^{N+1} P8N^2+20N+12+i + \sum_{i=1}^{N} P8N^2+22N+15+i \right]$$

where the given $P_i$ values correspond to certain system states where the last stage is operative. The average in-process inventories are calculated by

$$\bar{I}_i = \sum_{j=1}^{8N^2+24N+18} n_{i,j} \cdot P_j \quad ; \quad i=1,2$$

where $n_{i,j}$ is defined as the inventory level in the i-th storage corresponding to the j-th system state.

Running a computer program designed to compute system performance measures several times, for $\varepsilon=10^{-5}$, N=0,1,2 and different combinations of system parameters steady-state probabilities are obtained. A close look at the results gives insight into the structure of the system.

Steady-state probabilities can be subdivided into three classes according to their orders of magnitude. In general, states statisfying the condition that all three stages are up within a cycle have the highest probabilities. Total probability for this set of states increases as the ratio of failure rate to repair rate of system components decreases.

The second cathegory includes the states where only one of the stages is down and the other two are forced to shut down. This results from the fact that the analysis is made only for N=0,1,2. For realistic repair rate values the ratio of time in which the other stages keep operating processing parts in the buffer stores or adding parts to the stores, to the mean repair time of the broken down stage is very small. But increasing storage capacity N, holding the parameters constant, decreases such steady-state probabilities significantly (see Table 2.3).

TABLE 2.3. Changing of steady-state probabilities with increasing storage capacity

| | b(1)=b(2).01 , b(3) = .02<br>r(1)=r(2)=r(3)=.1 | |
|---|---|---|
| N=0 | P(111) | = .6975758048 |
| | P(OII) | = .0631347697 |
| | P(BOI) | = .0700222978 |
| | P(BBO) | = .1384955702 |
| N=1 | P(111; 00) + ... + P(111;11) | = .6961008733 |
| | P(OII; 00) | = .0553245195 |
| | P(BOI; 10) | = .0648895856 |
| | P(BBO; 11) | = .1280239271 |
| N=2 | P(111; 00) + ... + P(111; 22) | = .6958713888 |
| | P(OII; 00) | = .0491781290 |
| | P(BOI; 20) | = .0603982191 |
| | P(BBO; 22) | = .1185138427 |

The third group involves all remaining states which have the lowest steady-state probabilities.

It can be shown that also storage states influence the magnitude of steady-state values. The results point out that the system runs most often near boundaries, which means that the number of part in storage areas are mostly equal to lower or upper bounds. This results from the fact that N is small and if any stage breaksdown both of the storage levels converge to their lover or upper limits in a few cycles and stay there until a new failure occurs changing their storage state (see Table 2.4).

TABLE 2.4. Magnitudes of steady-state probabilities

| b(1)=b(2)=b(3)= .0002<br>r(1)=r(2)=r(3)= .01 | | N=2 |
|---|---|---|
| p(111; 00) | = | .3121764964 |
| 01 | = | .0020638867 |
| 02 | = | .0020672333 |
| 10 | = | .0020889663 |
| 11 | = | .0000069652 |
| 12 | = | .0000072257 |
| 20 | = | .3142951349 |
| 21 | = | .0010479556 |
| 22 | = | .3090831749 |

Although line efficiency is known to vary between E(0) and E(∞), it is important to know the rate at which this increase occurs with respect to buffer capacities. Depending on system parameters a significant improvement of efficiency can be achieved by providing extremely small buffers or only by installing large inventory banks. Figure 2.7 shows how the production rate of the system changes by increasing buffer

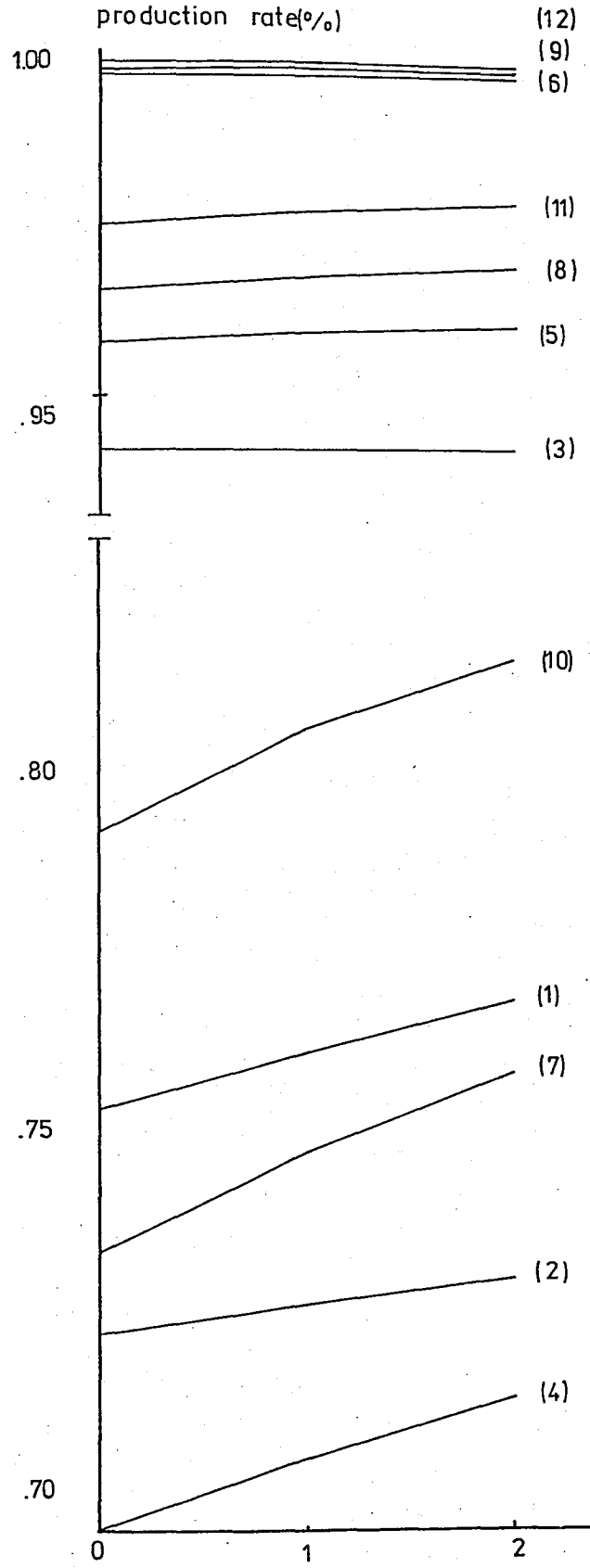


Figure 2.7. Change of Production Rate by increasing buffer size

TABLE 2.5. Failure and repair rates of 12 different cases

| Case | b(1) | b(2) | b(3) | r(1) | r(2) | r(3) |
|---|---|---|---|---|---|---|
| 1 | .01 | .01 | .01 | .1 | .1 | .1 |
| 2 | .005 | .005 | .005 | .04 | .04 | .04 |
| 3 | .0002 | .0002 | .0002 | .01 | .01 | .01 |
| 4 | .01 | .01 | .02 | .1 | .1 | .1 |
| 5 | .001 | .001 | .002 | .1 | .1 | .1 |
| 6 | .0001 | .0001 | .0002 | .1 | .1 | .1 |
| 7 | .01 | .02 | .03 | .2 | .2 | .2 |
| 8 | .001 | .002 | .003 | .2 | .2 | .2 |
| 9 | .0001 | .0002 | .0003 | .2 | .2 | .2 |
| 10 | .03 | .02 | .01 | .3 | .3 | .3 |
| 11 | .003 | .002 | .001 | .3 | .3 | .3 |
| 12 | .0003 | .0002 | .0001 | .3 | .3 | .3 |

capacity from 0 to 2 for twelve different cases. In addition, average storage fills of buffers 1 and 2 are presented in Figure 2.6 for the same cases.

Generally speaking, when the stages fail extremely rarely, and when they fail, it takes very long to be repaired, the influence of small buffers on production rate is negligible. only large buffers can improve efficiency in such cases, since they take longer to empty or fill up. On the other hand, if the failures occur frequently and the repair times are short, small buffers can improve production rate significantly. Accordingly, average storage fills are high in the former case and low in the latter case because the storages empty and fill up frequently.

In cases, where all stages have equal efficiencies in isolation the effect of increasing buffer capacity is most clearly visiable. Graphs showing efficiency increase are almost linear. The average inventory in the first storage is greater than the second although capacity limits are equal. The reason is the blocking effect of stage 2 and 3 on the first stage.

TABLE 2.6. Change of average storage fills by increasing buffer size

| | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 | Case 6 |
|---|---|---|---|---|---|---|
| $\bar{I}_{1_{N=1}}$ | .648033 | .658687 | .665411 | .730581 | .736709 | .737417 |
| $\bar{I}_{1_{N=2}}$ | 1.260164 | 1.302797 | 1.326536 | 1.428722 | 1.433972 | 1.434376 |
| $\bar{I}_{2_{N=1}}$ | .322122 | .328869 | .332249 | .490204 | .488444 | .488156 |
| $\bar{I}_{2_{N=2}}$ | .636185 | .655862 | .663531 | .976024 | .964274 | .962593 |

| | Case 7 | Case 8 | Case 9 | Case 10 | Case 11 | Case 12 |
|---|---|---|---|---|---|---|
| $\bar{I}_{1_{N=1}}$ | .809805 | .815890 | .816583 | .451831 | .472548 | .474753 |
| $\bar{I}_{1_{N=2}}$ | 1.576754 | 1.578655 | 1.578702 | .805751 | .837220 | .840328 |
| $\bar{I}_{2_{N=1}}$ | .481624 | .484696 | .484965 | .142279 | .147404 | .147866 |
| $\bar{I}_{2_{N=2}}$ | .939305 | .935546 | .934792 | .260057 | .271315 | .272356 |

If the first stage is most efficient, the first storage is often nearly full. Thus, the downstream stages get rarely starved so that increasing second storage capacity their individual efficiencies can converge to their values in isolation.

Finally, if teh last stage is most efficient, the parts emerging from the second stage are most often instantly processed by the third (machine) stage. The expected value of in-process inventory in storage 2 is small because it is often nearly empty. Again, it will be profitable to increase first storage capacity in order to increase line efficiency.

## 2.4. Limitations of the Model

Predictions of the Markov Chain Model show how the most important performance measures, line efficiency and average inventories, tend to change by increasing storage capacities for given system parameters. But unfortunately, the assumptions on the distributions of this model (and other available models) are too elementary for the real world situation so that the presented model do not enable precise predictions of the effect of the buffers. However, it is extremely difficult to develop an analytic model which incorporates both operation and time dependent failures and appropriate downtime distributions.

Also the symmetric storage capacity case may not be common and constrains the analysis.

But although some assumptions of the introduced model may not reflect the real system's behaviour, it gives insight into the mechanism of the real system and inaldes to evaluate real production lines by extending it to more closely conform to actual situations.

# III. SIMULATION MODEL OF MULTISTAGE TRANSFER LINE

Although it is possible to extend the Markov Chain model to deal with larger multistage systems it is very difficult to derive a solution from it. In this case it is appropriate to develop a dynamic simulation model of multi-stage transfer line in order to obtain approximate solutions. The basic idea will be to imitate the behaviour of the real system by generating random values for key model variables, such as up-and downtime of each stage, that obey certain probability laws as the corresponding system variables. By conducting numerical experiments on the model, it is possible to observe the system's behaviour over time and evaluate certain of its design characteristics or different decision rules for its operation, like the allocation and capacity of the buffers.

## 3.1. Problem Statement

The assumptions of the dynamic simulation model do not differ much from the analytic model's.

The stages are arranged serially so that each workpiece enters the line at the first stage, and begins to transfer from one stage to the next at equal time intervals, which are called cycle time. Cycle time includes processing and transfer time and it is taken as a unit time.

Both up-and downtime are assumed to have exponential probability distributions, which has memoryless property. Failure and repair rates are defined as in the Markov Chain Model. So, mean uptime(downtime) of any stage is defined as the reciprocal of the failure (repair) rate of this stage.

Stage states in the simulation model are defined the same as in the analytical model. With one difference states "starved" and "blocked" are combined into one state, namely "forceddown" or "idle" state.

It is assumed that there will be no lack of raw material for the first stage and the last stage will never be blocked because there always exists storage space for finished items. Parts are not created or destroyed by the system, so that every workpiece entering the line leaves it as a finished item.

Machines breakdown only due to operation dependent failures, so that during idle periods no stage can breakdown.

Finally, since all stages work synchronously, there is no feed forward information flow, so that the knowledge that a place will be vacant in the downstream storage or that a piece will emerge from the upstream stage in the time cycle to follow does not influence the decision on whether or not to attempt to process a piece.

## 3.2. System Description

System parameter, variables and performance measures are defined as follows:

System Parameters

M: Number of stages (M-1: number of buffers)
Td(i): Mean downtime of stage i in cycles
F(i): Failure rate of stage i per cycle

Input variables

B(j): Capacity of j-th storage
finish: length of simulation run in cycles

Output variables

S(i): State of stage i (up, down or idle)
newup: Uptime generated, in cycles
dur: Downtime generated, in cycles
time(i): time of next state change of stage i
uptime(i): time of next up-or downtime generation of stage i
clock: global time pointer
C(i): total uptime in cycles of stage i
N(i): number of failures of stage i
DT(i): total downtime in cycles of stage i
FC(i): number of shut downs of stage i due to failures elsewhere
1(j): number of units in j-th storage area
ML(j): maximum number of units in j-th storage area
CIL(j): cumulative number of units in j-th storage area

System performance measures

C(i)/finish: Efficiency of stage i
CIL(j)/finish: Mean number of units in j-th storage area
finish-(C(i)+DT(i)): Total idle time of stage i

## 3.3. Description of the Computer Program

A computer program is designed to simulate the system (see Appendix).

Each stage is represented by a record containing data corresponding to the parameters and variables of the stage, such as failure rate, mean downtime, state of the stage etc. Capacity of each storage and length of simulation is specified before each run. Level of a buffer stock is bounded by 0 and capacity of the storage.

The program is so structured that it consists of 3 functions, 8 procedures and a short main part. Function UNİFORM generates pseudo random numbers between 0 and 1 given an initial seed value, using the linear congruential method.

Function EXPON generates up-and downtimes which are exponentially distributed with a given mean up-or downtime and a random number between 0 and 1. Real downtimes are rounded to next greatest integer and real uptimes are truncated because time is measured in discrete cycles.

Function OK determines whether an idle stage will be operating or forceddown in the next cycle conditional on the level of the adjacent storages.

Procedure FORCEDDOWN determines how long a stage will be idle according to the state and remaining repair or idle time of the stage which has forced it to shut down.

Procedure REARRANGE recomputes the remaining idle time(s) of the forceddown stage(s) if a zero uptime for the brokendown stage is generated, ie. if it breaks down again within teh first cycle following the last repair.

Procedure INITIALIZE assignes initial values of system parameters and variables depending on whether initial system state is equal to the origin state (all states idle, all storages empty) or not.

Procedure BOUNDEVENTS takes are of the stages whose time is equal to clock time and makes the necessary state transitions and storage adjustments. If stage state is up it will be changed into idle and a part will be removed from the upstream storage and one added to the downstream storage. For a breaken down stage whose repair has just finished a new uptime will be generated. In case of a zero-uptime (stage fails within the first cycle following the last repair) state is changed into idle, otherwise time and uptime variables are revalued and state will be changed into up. If the observed stage is forced down a new uptime will be generated and added to uptime value of the stage.

Procedure INVENT adds the number of units in the j-th storage to CIL(j) for all j after each change of clock time. If level of any buffer j has reached the highest value since the beginning of simulation run this value will be replaced with the present value of ML(j).

Procedure CONDITIONALEVENTS decides for a stage to stop or continue working within the present cycle given that it is idle and its time is equal to clock time. State changes from idle to down if uptime of any stage is equal to clock time. Otherwise state changes from idle to up or idle depending on the level of the adjacent storages.

Procedure SETCLOCK determines the increment in the clock time according to the lowest time value among the stages which are not idle.

Procedure REPORT computes the performance measures of the system and prints values of important system parameters and variables.

Main part of the program initializes system parameters and variables and simulates the system running through a repetitative statement which calls boundevents, invent, conditionalevents and setclock respectively and stops when clock time reaches a given limit.

## 3.4. Computation of Performance Measures

As mentioned before, in stochastic system steady state does not imply that the system itself is at rest, but that the probabilistic model of the system has become stationary. Although the system continues to fluctuate, it has been running long enough so that the effect of the initial conditions have vanished. Consequently, it can be said that long time average is the practical equivalent of the concept of steady state.

In our case, the problem is how long the system has to be simulated in order to obtain sufficiently precise solutions. Several runs made for different lengths of time and system parameters showed that systems in which stages fail frequently and repairs are made in short times converge to steady state extremely fast. But in systems with small probabilities of failure and repair it takes longer that the fluctuations on the system performance decrease to a sufficiently low level. This shows how strongly the convergence rate depends on system parameters. Figure 3.1 and 3.2 show changing of production rate and average storage fills of a 3-stage system, with identical failure and repair rates, which are equal to .005 and .067 respectively. Continuous lines show sample averages at the end of each fixed interval length of 500 cycles where

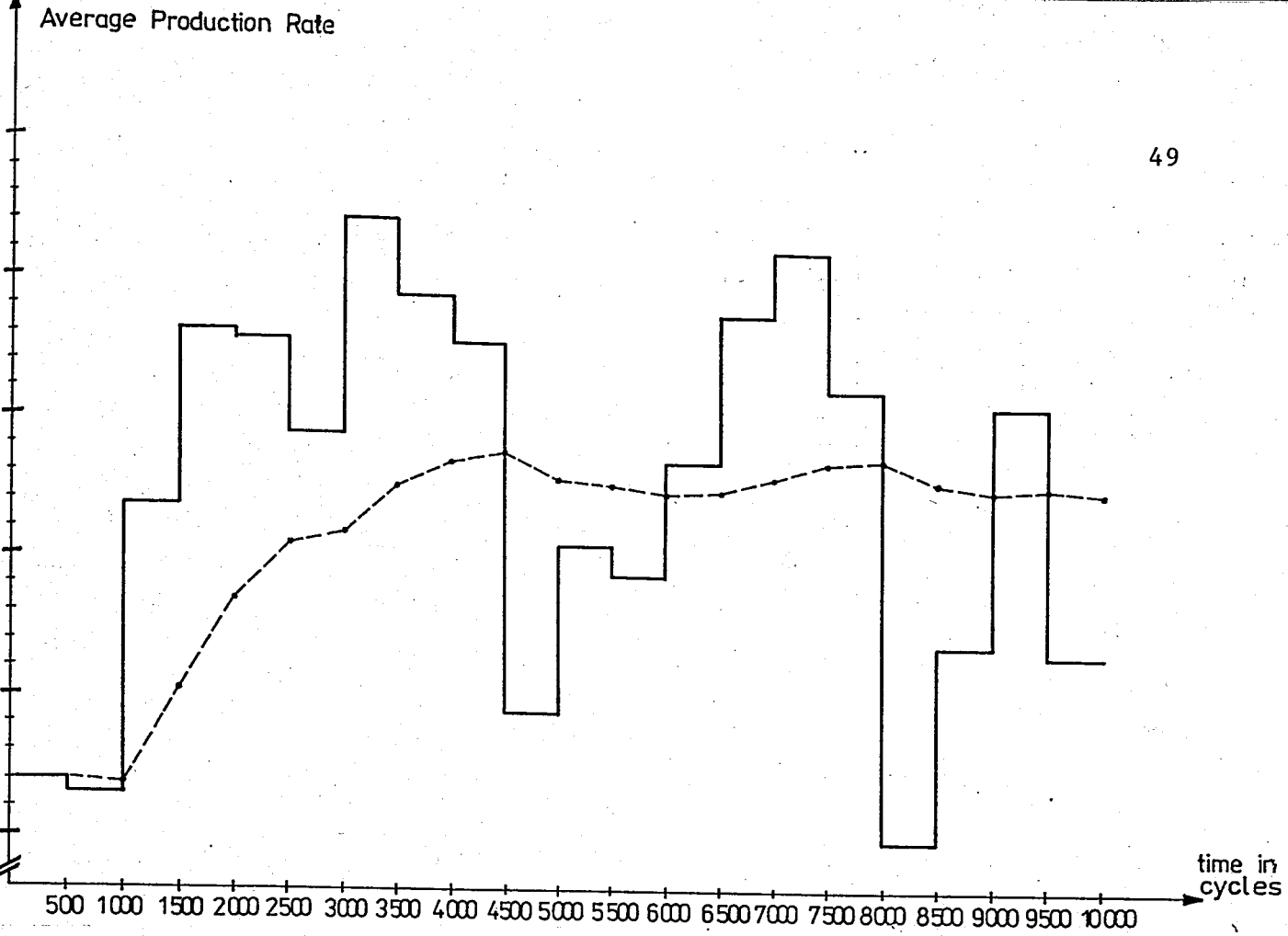


Fig.3.1. Sample and Cumulative Averages of Production rate in a three-stage line

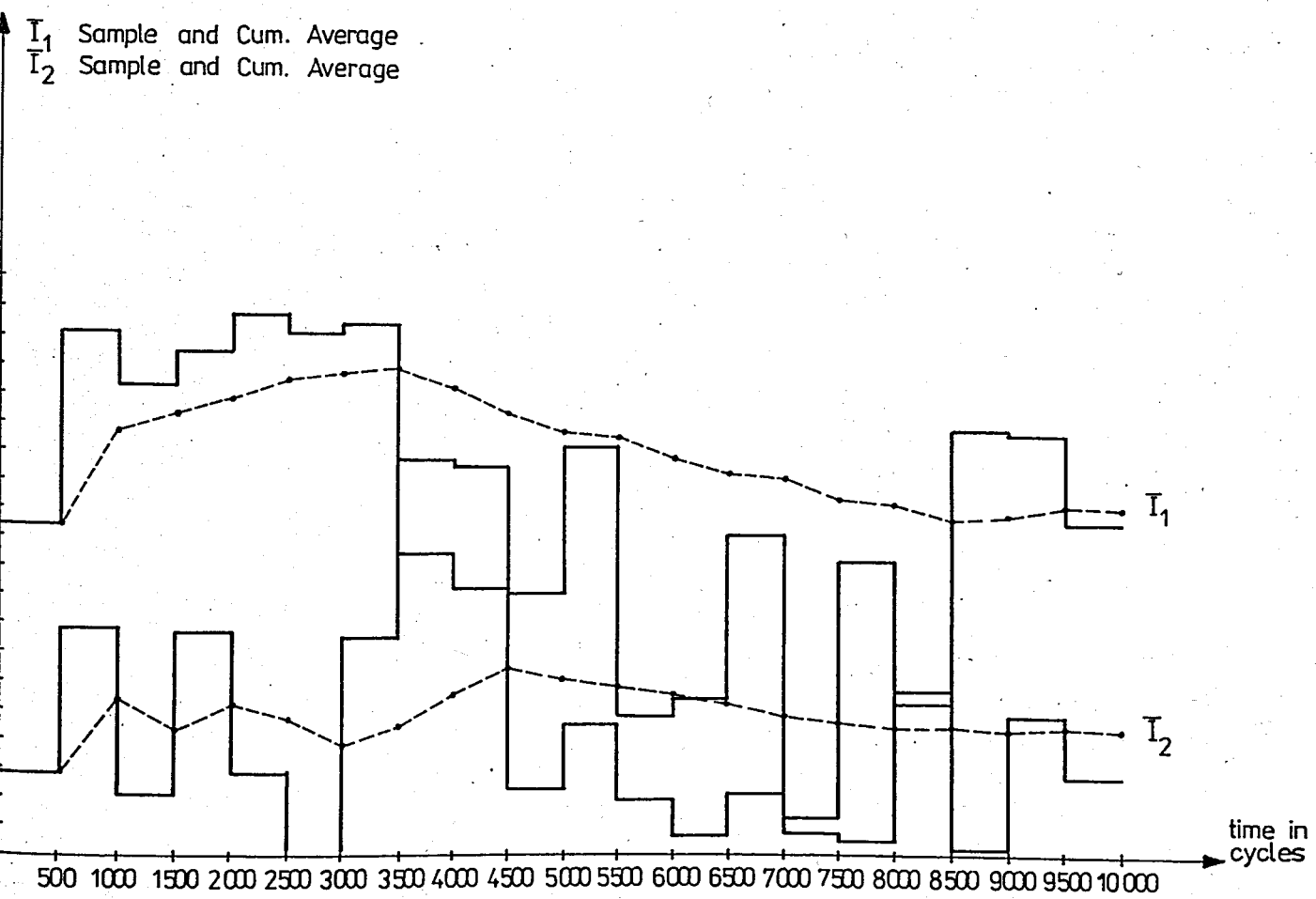


Fig.3.2. Sample and Cumulative Averages of Mean Storage fills in a three-stage line

the discrete curves show cumulative averages. It is clearly visiable that after a total of 10.000 cycles fluctuations of cumulative averages die away.

Although the sample averages in this way are not uncorrelated, given that intervals do not overlap, because they follow each other immediately, both graphs give a good picture of the finite-time, or non-steady-state behaviour of the system.

## 3.5. Discussion of Results

Using dynamic simulation, behaviour of a 3-stage system is observed for three cases with different characteristics. To obtain sufficiently precise solutions the system is simulated long enough depending on the magnitude of its parameters. Presented results are the average values of the solutions obtained from several runs after the system has reached steady-state.

For all cases first storage capacity is held constant at an appropriate level and second storage capacity is varied in positive increments. The change in production rate, average storage fills and forceddown times is plotted against the second storage capacity.

In the first case failure and repair rates are high and $E_1(\infty) > E_3(\infty) > E_2(\infty)$. In second case failure probabilities are small but repair probabilities are large respectively, where $E_1(\infty) > E_2(\infty) > E_3(\infty)$. Third and last case represents a system with small failure and repair probabilities, where $E_3(\infty) > E_2(\infty) > E_1(\infty)$ (See Table 3.1).

TABLE 3.1. Failure and repair probability combinations for 3 different cases

| | b(1) | b(2) | b(3) | r(1) | r(2) | r(3) |
|---|---|---|---|---|---|---|
| Case 1 | 1/100 | 3/100 | 5/100 | 1/15 | 1/10 | 1/5 |
| Case 2 | 3/1000 | 5/1000 | 7/1000 | 1/20 | 1/25 | 1/25 |
| Case 3 | 5/1000 | 2/1000 | 1/1000 | 1/150 | 1/100 | 1/100 |

(i) Case 1 (see Figure 3.3)

In the first case production rate curve shows clear concavity and approaches asymptotically the efficiency in isolation of the least efficient stage, which is the second one.

Accordingly curves for forceddown times of each stage have the form of rectangular hyperbola, decaying fast and approaching x-axis. At some initial increase of buffer capacity the improvement in line efficiency and the decrease in forceddown times is fast but there after increasing storage capacity brings progressively smaller improvement.

Moreover, increasing storage capacity of the second buffer increases the mean number of units in the second storage area almost linearly with a slope less than 1. At the same time the average number of units in the first storage decreases in constant and small increments.

Because first stage is the most efficient one, first storage is often nearly full. Larger failure probability of the third stage with respect to the second one causes the increase of average inventory in the second storage with increasing capacity.

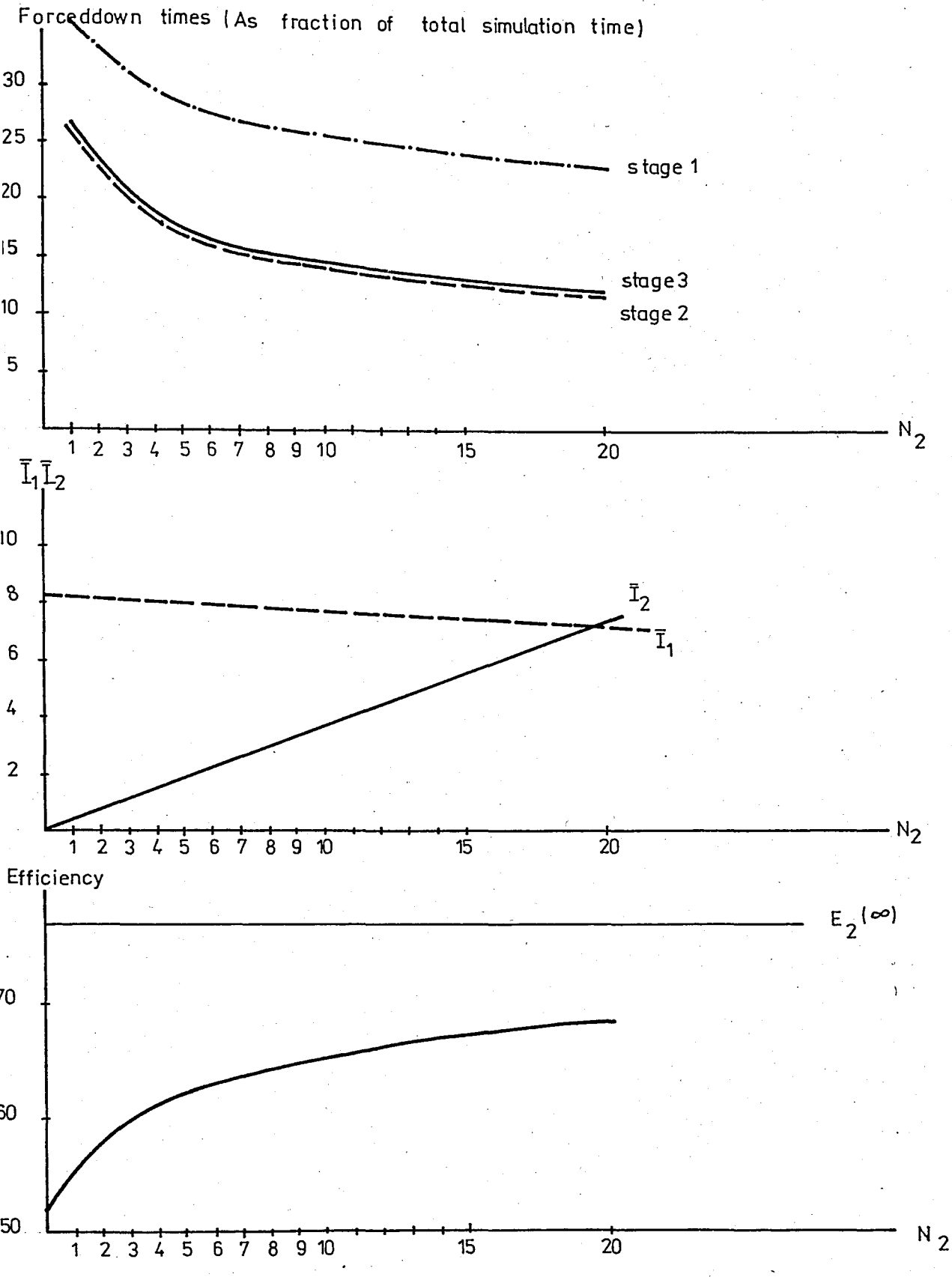


Figure 3.3. Performance measures for increasing $N_2$ in case 1

Since the upstream storage is mostly at the upper limit if the capacity of the downstream storage is sufficiently enlarged there will be probable always vacant place for workpieces emerging from the second stage when last stage is brokendown. This means that the real efficiency of the second stage will converge to the theoretical value in isolation. In this case line efficiency will approach the minimum $E_i(\infty)$ value in the system because the stages are nearly decoupled by the buffers. So, in systems with the same characteristics defined in case 1, it will be profitable to increase storage capacity between the second and third stages if the corresponding holding cost is not very high.

(ii) Case 2 (see Figure 3.1):

Like in the first case production rate of thesystem increases and approaches asymptotically the efficiency in isolation of the third stage with increasing capacity of storage 2. But the necessary storage enlargement for the same efficiency improvement is much larger than the first case.

Since second stage is more efficient than the third one average storage fill of the second buffer grows very fast with increasing capacity, which leads to a very steep dicrease in forceddown time of the third stage initially. For a buffer capacity of 200 units the ratio of forceddown time to total time is below 2%. This shows once again that the third stage operates like isolated for sufficiently large upstream storage capacity. Since efficiency in isolation dicreases in the downstream direction there is no significant effect of the first buffer size on line efficiency.

Average buffer stock in storage 2 increases approximately linear at first, and then turns to be smooth concave as capacity increases. At the same time mean inventory level in

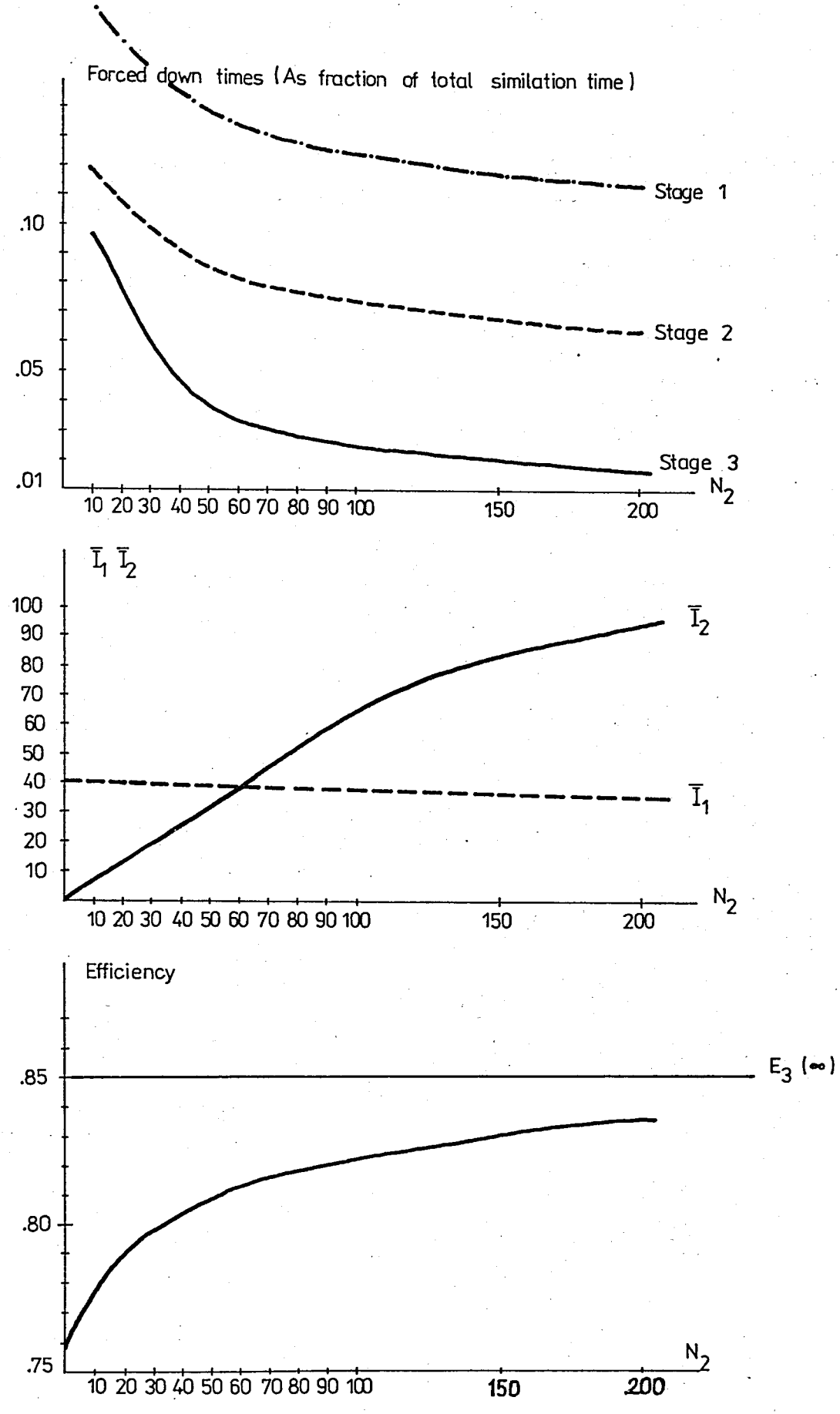


Figure 3.4. Performance measures for increasing $N_2$ in case 2

storage 1 dicreases linearly, because blocking effect of the third stage on the first one is diminished by larger capacity of storage 2.

Again, good buffer effect can be expected by increasing second storage capacity in any line which has a similar structure to the given system.

Case 3 (See Figure 3.5)

In case 3, increasing second storage capacity yields a very small improvement rate for line efficiency. Although production rate seems to converge to the isolated efficiency of the first stage, there must be an extremely large buffer provided between the last two stages to reach this value.

Forceddown times of the first and least stages dicrease significantly with increasing buffer capacity where the second stages does not differ much.

Average inventory curve of the second buffer shows concavity with growing buffer size, where average inventory in storage 1 decreases linearly with a very small slope.

Since last stage is the most efficient one in the line, parts emerging from the second stage are most often processed by the third stage without having to wait in storage 2. This implies that the second storage is often nearly empty, so that providing it with a large capacity is not reasonable. Accordingly, little is gained in case 3 by increasing second storage capacity. however, enlargement of the first storage will cause rapid increase in production rate.

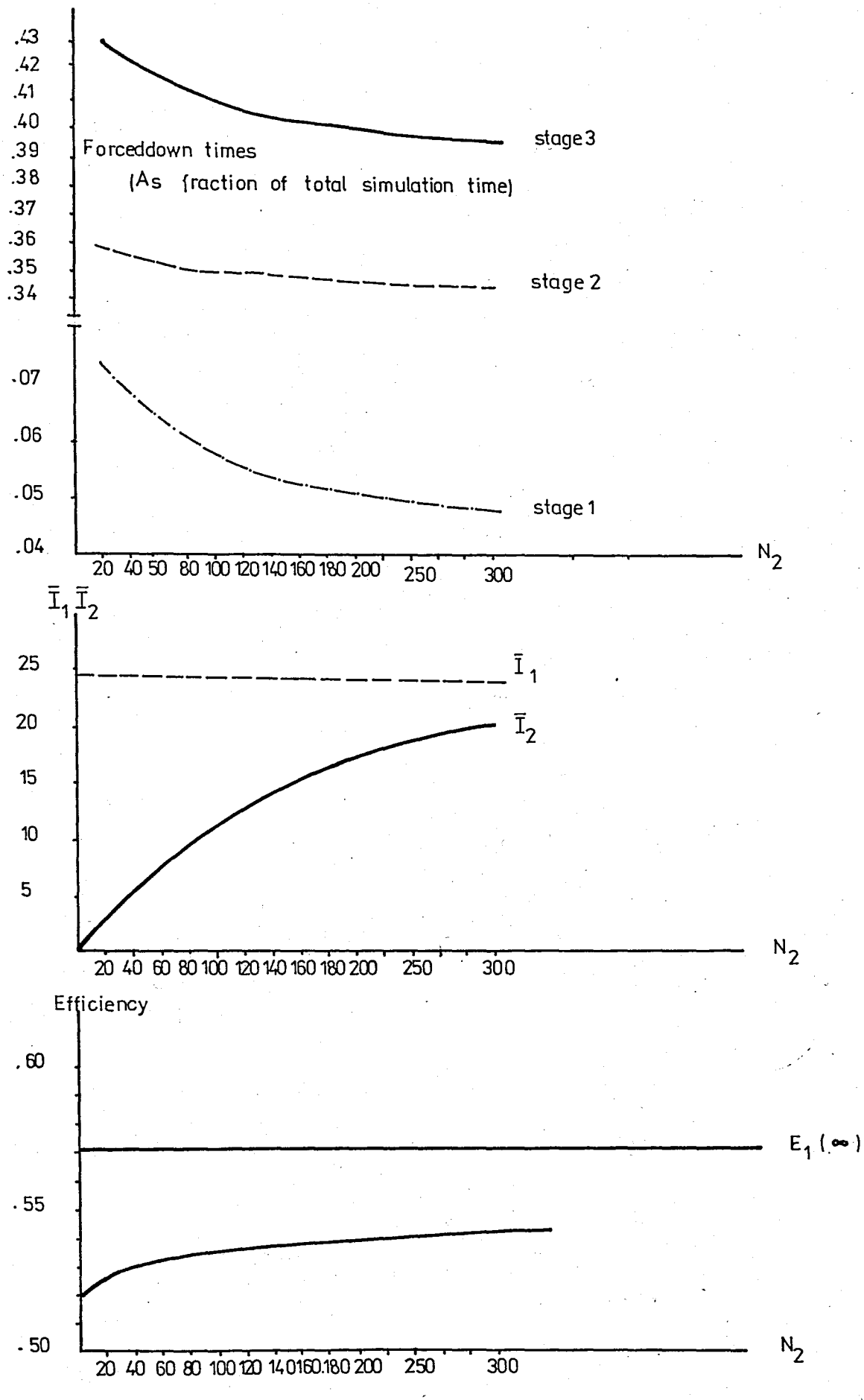


Figure 3.5. Performance measures for increasing $N_2$ in case 3

# IV. APPLICATION OF THEORY

Both analytical and simlation model are developed to gain insight into the effect of buffer storages in multistage transfer lines. But real systems are too complex to be represented by mathematical model even if modeling assumptions are chosen realistically.

In the following chapter the assumptions and behaviour of the models will be compared with a real system, pointing out their shortcomings and weaknesses.

## 4.1. Description of the Actual System

The system under study is a bottling line of an industrial manufacturer. It produces two kinds of commodity of beverage type alternatigly. The line consists of 5 machines (two of them are integrated into one stage), 3 inspection units and transfer bars (see figure 4.1).

The activity centers are the following:

(i) Loading machine
(ii) Washing machine
(iii) Filling + Capping machine
(iv) Unloading machine

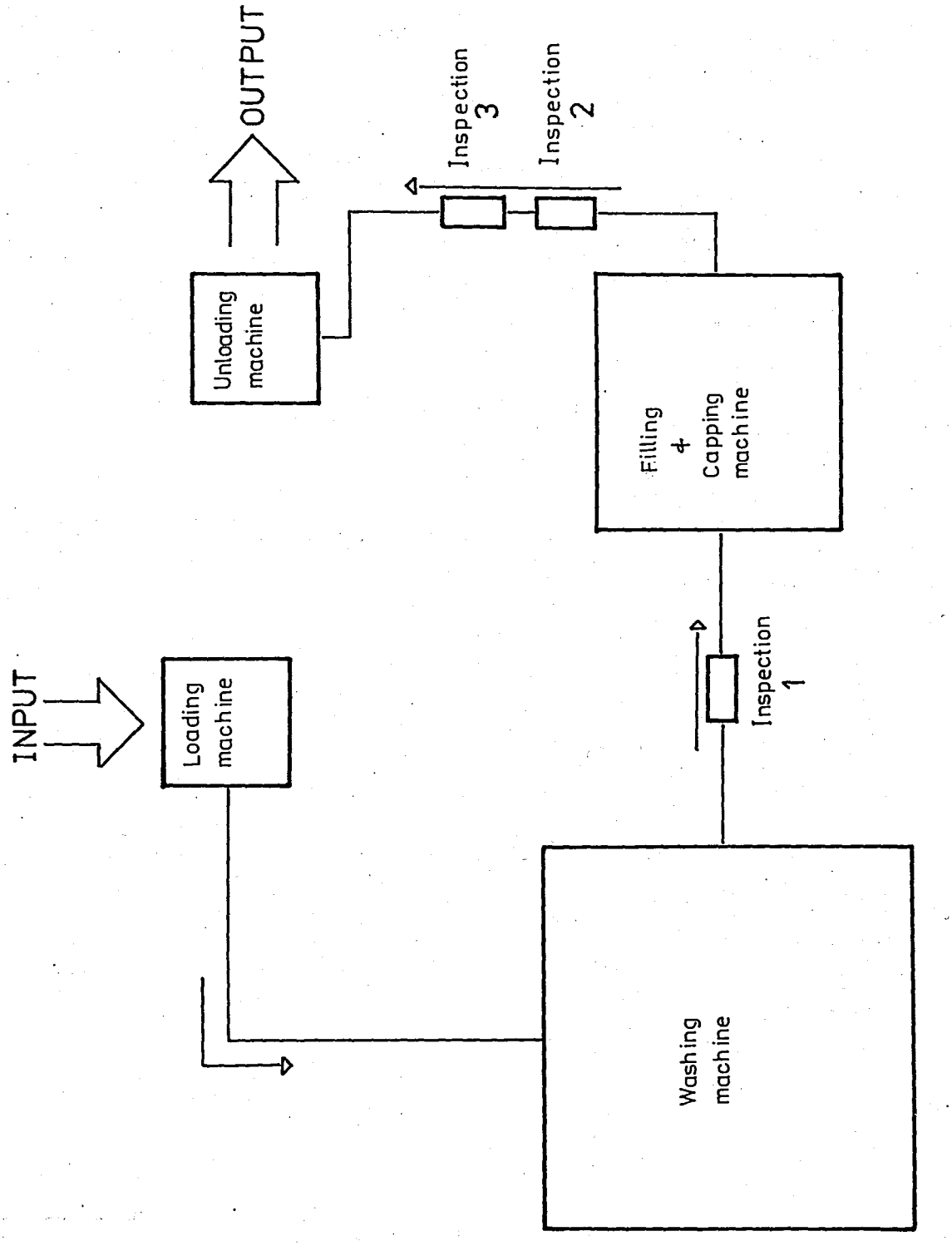


Figure 4.1. Four-machines bottling line

Dirty bottles loaded on the line by the loading machine are transfered to the washing station. After being washed in 4 different phases bottles are transfered to the filling machine. On the way they go through an inspection unit where the damaged and uncleaned articles are removed. In the filling station bottles are filled with syrup and $CO_2$ gas under pressure coming from holding tanks. Workpieces emerging from the filling machine go directly to the capping unit which is integrated to that one. After copping operation items pass by two inspection units which are serially located along the transfer bar before being unloaded. If the content of any item is below the standard amount then it is emptied and loaded to the filling machine once more.

## 4.2. Service Times

Concept of cycle time cannot be applied to the actual system. Workpieces do not enter the loading, washing and unloading stations one by one. The entrance to those machines and the emergence from these machines occurs in batches of size 48 and 24 units for loading and unloading stations, and washing machine respectively. Moreover, parts are not operated singly. There are always more than one workpiece on the machines being at any phase of the continuous operations.

Time between each loading and unloading is adjusted according to the processing rate of F+C (filling + capping) machine. To avoid starving of the downstream machines due to removed parts, the upstream machines operate at faster rates.

However, a common cycle time can be defined for the system, where cycle time ise taken as the expected time to produce one unit of item.

For an adjusted production rate of 213.2 units/min cycle timeof F+C machine is nearly 0.28 1/sec. In this case washing machine releases 24 units at every 6 seconds which corresponds to a cycle time of 0.25 1/sec.

As in the Markov Chain model stated, if deviations from the mean service time are small then constant machining cycle assumption will be justified.

## 4.3. Failure and Repair of Machines

Machines, inspection units and conveyor belts of the bottling line fail at random times. Failure probability of inspection units and conveyor belts are very small so that they can be ignored or modeled as failures of adjacent machines. If they were significantly large then it would be advisable to model these components as unreliable, individual parts of the system.

There are a great number of causes of machine stoppages. Some failures are easy to diagnose and quick to repair (mostly by the operators without help of repairmen) such as workpiece jams, but others involve more serious and time-consuming breakdowns, such as material shortage and tool breakage.

There exist also scheduled downtimes for the entire line but these are not of great interest for an analysis of buffer effect. But fortunately the existance of a program of tool replacement supports the assumption of constant failure probability in any cycle. Otherwise tool wear as a cause of machine breakdowns would imply an increasing probability of stoppage with time.

Examining the actual failure and repair probability distributions of the machines, it can be said that they are exponential with memoryless property. Loading and unloading machines fail rarely but repair times are longer. However, washing and F+C machines breakdown more frequently where repairs do not take much time.

Finally, great majority of the observed breakdowns were due to operation dependent failures. But there are also a few recorded cases where time dependent entire line failures, such as energy and material shortage, occured. Consequently it can be claimed that an operation dependent model is more appropriate than any time dependent model for the bottling line.

## 4.4. Conservation of Workpiece

Contradicting with both models parts are removed due to inspection operations or damaged or destroyed by the system.

Bottles are removed from the line if they are not cleaned or damaged. Also at certain times samples are taken for quality control tests. Finished items rejected at the inspection units 2 and 3 are emptied first and then loaded on the F+C machine to be filled again.

The assumption that parts are scrapped at station breakdowns is not reasonable for the whole line but in a certain number of failures of F+C machine bottles are really destroyed by the system.

## 4.5. Conveyor Belt

Each machine is connected to the next one with a conveyor belt. Parts are transfered from one machine to the next with the help of those transfer bars.

It is common in the literature to encounter conveyor belts as in-process storage facility. Also in the actual system there are large amounts of workpieces moving on the transfer bars. But they cannot be considered as units in storage areas. The reason is that the conveyors have always to be full to make transfer easier and to avoid the bottles to fall down. Otherwise falling down bottles hinder the transfer or they are damaged because glass is very sensitive to crashes at high temperature changes. Therefore operators shut down machines if any other machine breaksdown although there are enough workpieces on the conveyors to keep on operating.

## 4.6. Efficiency of the Line

Demand for the commodities which are produced by the bottling line shows seasonal fluctuations. In cold times demand decreases, so the production also decreases because the items cannot be stored over a long time. But in warm seasons demand increases so that the line has to operate at full capacity to meet the increasing demand. In such times efficiency of the line becomes very imporant. Idle times caused by single machine failures result in shortage cost due to lost sales. Increasing processing rates can be a solution but this increases failure probability of machines and the number of defective items. In this case one way of improving efficiency is to provide inventory banks between the workstations.

Data obtained from the observations show that b(2) = .000347, r(2) = .01136 and b(3) = .000553, r(3) = .01695. Data obtained from the records show that the isolated efficiencies of the loading and unloading machines are over 99%. Although records contain only important stoppages, it is known that short repairs are not common for the first and last workstations, so that computed values may not for from the truth.

Analysing the results it can be decided to divide the line into two stages and provide a real buffer storage between them, where stage 1 contains loading and washing machines and stage 2 contains F+C and unloading machines. Since washing and F+C units are the most inefficient stages in the line it would improve efficiency which has a mean value of 93.8%. Allocation of the storage can reduce the blocking effect of the F+C machine and the starving effect of the washing machine and line efficiency can be increased up to the efficiency in isolation of the F+C machine.

But the situation is not so simple. Waiting of cleaned up bottles in the storage may cause that they become dirty again with dust particules in the air. So, there must be a limit on the average time a part has to wait in the storage. As a result, this fact which was also discussed in the economic analysis of the problem, where mean number of units in certain storages was to be limited for high holding cost of items, must also be taken into account in the modeling attempt of the line.

# V. CONCLUSIONS

The aim of the work presented here has been to obtain analytical and numerical methods in order to quantify the relationships between design parameters and performance measures in unreliable transfer lines with interstage buffer storages.

The justification for providing buffer stocks in a particular transfer line requires a complete economic and technical study which takes into account the cost of providing buffers and the benefit of increased line efficiency. However, the results of the theoretical analysis in this study provide some general answers to certain questions that must be asked when the decision of installing buffer stocks is being considered.

1- It is obvious that when the line efficiency is high the gain in efficiency through using a given set of storage capacities becomes small. For example, the same buffer capacity could increase the efficiency of a line from 44 per cent to 64 per cent as from 68 per cent to 77 per cent. Thus, when line efficiency is high buffer are difficult to justify.

In cases where line efficiency is low and single station failures are the main reason for line stoppages substantial improvement can be expected by introducing buffer storages.

Also, if there is one stage with extremely low isolated efficiency compared to other stages, buffers will not greatly contribute to the line efficiency.

2- To provide a buffer storage of large capacity between two stages where the successive stage is more efficient than the preceeding one is not profitable. Since the parts discharged from the upstream stage are instantly processed by the downstream stage the buffer is almost empty. On the other hand, if upstream stage is more efficient than the downstream stage it is advisable to enlarge the capacity of the storage between them, but this will also increase the mean number of units in the storage are.

3- Provided the capacity of each storage point is at least as big as the mean repair time of a stage, the marginal benefit of extra storage capacity decreases.

Simulation results show that it is unlikely that a capacity of each storage point greater than five times the mean repair time could be justified unless the marginal cost of extra storage capacity is negligible.

Results obtained from available models also support these conclusions. But, as previously stated, these models do not enable precise predictions of the effect of the inventory banks. So future research will be concerned in developing models incorporating both operation and time dependent failures and an appropriate downtime distribution for each category of repair and/or remedial action taken.

# REFERENCES

1- Bryant,J.L., Murphy,R.A., "Availability Characteristics of an Unbalanced Buffered Series Production System with Repair Priority", AIIE Transactions, Vol.13, No.3, April 1981.

2- Buzacott,J.A., "Automatic Transfer Lines with Buffer Stocks", The International Journal of Production Research, Vol.5, No.3, 1967.

3- Buzacott,J.A., Hanifin,L.E., "Models of Automatic Transfer Lines with Inventory Banks. A Review and Comparison", AIIE Transactions, Vol.10, No.2, June 1978.

4- Cohen,A.M., Cutts,J.F., Fielder,R., Jones,D.E., Ribbans,J., Stuart,E., Numerical Analysis, John Wiley and Sons, Inc. N.Y. (1973).

5- Çınlar,E., Introduction to Stochastic Processes, Prentice-Hall, Inc., Englewood Cliffs, N.Y. (1974).

6- Dervitsiotis,K.N., Operations Management, McGraw-Hill, N.Y. (1981).

7- Gershwin,S.B., Berman,O., "Analysis of Transfer Lines Consisting of Two Unreliable Machines with Random Processing Times and Finite Storage Buffers", AIIE Transactions, Vol.13, No.1, March 1981.

8- Gershwin,S.B., Schick,I.C., "Modeling and Analysis of Unreliable Transfer Lines with Finite Interstage Buffers", MIT Laboratory for Information and Decision Systems, Report ESL-FR-834-6. (1978).

9- Grogono,P., Programming in Pascal, Addison-Wesley Publishing, (1980).

10- Gujarati,D., Basic Econometrics, McGraw-Hill, N.Y. (1978).

11- Ignal,E., Silver,P.E.A., "The Output of a Two-Stage System with Unreliable Machines and Limited Storage", AIIE Transactions, Vol.9, No.2, June 1977.

12- Okamura,K., Yamashina,H., "Analysis of the Effect of Buffer Storage Capacity in Transfer Line Systems", AIIE Transactions, Vol.9, No.2, June 1977.

13- Sheskin,T.J., "Allocation of Interstage Storage Along an Automatic Production Line", AIIE Transactions, Vol.8, No.1, March 1976.

14- Young,M.D., Iterative Solution of Large Linear Systems, Academic Press, N.Y. (1971).

# APPENDIX I

```
PROGRAM  FINDSTATES;

CONST  L=10;

TYPE   line=string[L] ;cases=string[4] ;
       indice=1..L ;index=1..4 ;subind=1..1000 ;
       states=array [subind] of line ;

VAR    f:text;
       state:line ;
       t:subind ;x,N:indice ;position:index ;
       performance:cases ;
       posstate:states ;
       stage:char ;stop:boolean ;

PROCEDURE initialize;
 var    a:indice;
 begin
  assign(f,'ozercik.pas');
  rewrite(f);
  state:='          ';
  for a:=1 to N do
    insert('1',state,a);
  t:=1;posstate[t]:=state;
  performance[1]:='1';performance[2]:='0';
  performance[3]:='B';performance[4]:='I';
  stop:=false
 end;

PROCEDURE search;
 begin
  x:=N+1;
  repeat
    x:=x-1
  until (state[x]<>'I')
 end;

PROCEDURE change;
 begin
  delete(state,x,1);
  insert(performance[position],state,x)
 end;

PROCEDURE findpos;
```

```
begin
 stage:=copy(state,x,1);
 position:=pos(stage,performance)
end;

PROCEDURE control;
type  short=1..3;
var   a,b,c,z:short;
begin
 a:=pos('BI',state);
 b:=pos('B1',state);
 z:=a+b;
 if  z=0  then
  begin
   t:=t+1;
   posstate[t]:=state;
   if pos('01',state)<>0
    then insert('*',posstate[t],N+2);
  end
end;

PROCEDURE step;
var   i:indice;
begin
  for i:=(x+1) to N do
  begin
   delete(state,i,1);
   insert(performance[1],state,i)
  end
end;

PROCEDURE report;
var   j:subind ;
begin
      writeln(f);writeln(f);writeln(f);
      writeln(f,'*** SYSTEM STATES OF A ',N,'_STAGE TRANSFER LINE ***');
      writeln(f);writeln(f);writeln(f);
      writeln(f,'Characters defining operational status of each stage : ')
      writeln(f,'-------------------------------------------------------------');
      writeln(f,'1: operating ,0: down ,B: blocked ,I: idle  ');
      writeln(f);write(f,'*:only for lines with interstage buffers ');
                ;writeln(f);writeln(f);
      for j:=1 to t do
         writeln(f,'          ',j:4,' . .',posstate[j])
end;

PROCEDURE first;
begin
      repeat
       if position=1 then  position:=position+1
                     else  position:=position+2;
       change;control
      until (position=4)
end;

PROCEDURE second;
begin
      position:=position+1;
      change;step;control
end;

PROCEDURE third;
begin
      position:=position+1;
      if position<>4
        then begin change;step;control end
        else begin report;stop:=True end
```

```
                else begin report;stop:=true end
 end;

BEGIN
  read(N);
  initialize;
  repeat
    search;findpos;
    if x=N then first
           else if x=1 then third
                       else second
  until stop;
  close(f)
END.
```

*** SYSTEM STATES OF A 3_STAGE TRANSFER LINE ***

Characters defining operational status of each stage :

1: operating ,O: down ,B: blocked ,I: idle

*:only for lines with interstage buffers

```
 1 . 111
 2 . 110
 3 . 11I
 4 . 101 *
 5 . 100
 6 . 10I
 7 . 1BO
 8 . 1I1
 9 . 1IO
10 . 1II
11 . 011 *
12 . 010 *
13 . 01I *
14 . 001 *
15 . 000
16 . 00I
17 . 0BO
18 . 0I1
19 . 0IO
20 . 0II
21 . B01 *
22 . B00
23 . B0I
24 . BBO
```

```
C 3-STAGE TRANSFER LINE PROVIDED WITH BUFFER STORAGES
C OF SIZE N .A MARKOV CHAIN MODEL IS REPRESENTED .
C SYSTEM PERFORMANCE MEASURES ARE OBTAINED BY SOLVING
C A SET OF LINEAR TRANSITION EQUATIONS .GAUSS SEIDEL
C ITERATION METHOD IS USED TO COMPUTE THE STEADY STATE
C PROBABILITIES FOR THE SYSTEM STATES .
      PROGRAM MARKOV (INPUT,INIT2,INVENT2,OUTPUT,TAPE1=INIT2,
     #                TAPE2=INVENT2,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON N,IS,ICOOR(500,2),T(500)
      REAL LB1,LB2,LB3,M1,M2,M3,LB1B,LB2B,LB3B,M1B,M2B,M3B
      INTEGER AA,AB,BB,BC,CC,CD,DD,DE,EE,EF,FF,FG,GG,GH,HH,HI,II,
     #        OP,PP,PQ,QQ,QR,RR,RS,SS,ST,TT,TL,LL,VV,VW,WW,WX,XX,YY
      DIMENSION PX(500),X(500),RHS(500),IX(500)
      N=2
      READ(5,*) LB1,LB2,LB3
      READ(5,*) M1,M2,M3
      LB1B=1.-LB1
      LB2B=1.-LB2
      LB3B=1.-LB3
      M1B=1.-M1
      M2B=1.-M2
      M3B=1.-M3
C INITIALIZATION
      IS=0
      MSIZE=8*(N*N)+24*N+18
      DO 100 I=1,MSIZE
      PX(I)=.0
      ICOOR(I,1)=1
      ICOOR(I,2)=I
      T(I)=1.
      RHS(I)=.0
      IX(I)=0
  100 T(I)=1.
      READ(1,*)(X(I),I=1,MSIZE)
      REWIND 1
      IS=MSIZE
      RHS(1)=1.
C CODING
      AA=1
      AB=N*(N+2)+1
      BB=AB+1
      BC=2*AB
      CC=BC+1
      CD=CC+N
      DD=CD+1
      DE=CD+N*(N+1)
      EE=DE+1
      EF=DE+AB
      FF=EF+1
      FG=FF+N
      GG=FG+1
      GH=GG+N
      HH=GH+1
      HI=HH+N
      II=HI+1
      IJ=II+N
      KK=IJ+1
      LL=KK+1
      LM=KK+N*(N+1)
      MM=LM+1
      MN=LM+N*(N+1)
      NN=MN+1
      NO=MN+N
      OO=NO+1
      OP=NO+N*(N+1)
      PP=OP+1
```

```
      PC=CP+AB
      QC=PQ+1
      QR=CQ+N
      RR=QR+1
      RS=RR+N
      SS=RS+1
      ST=SS+N
      TT=ST+1
      TU=TT+N
      UU=TU+1
      VV=UU+1
      VW=UU+N
      WW=VW+1
      WX=WW+N
      XX=WX+1
      YY=XX+1
C     ASSIGNMENT OF TRANSITION PROBABILITIES
      CALL ASG(AA,AB,AA,AB,1,1,LB1B*LB2B*LB3B,0)
      CALL ASG(AA,AB,BB,BC,1,1,LB1B*LB2B*LB3,2)
      CALL ASG(AA,AB,DD+N,DE,1,1,LB1B*LB2*LB3E,1)
      CALL ASG(AA,AB,EE+N+1,EF,1,1,LB1B*LB2*LB3,0)
      CALL ASG(AA,AB,FF+1,FG,N+1,1,LB1B*LB2*LB3B,0)
      CALL ASG(AA+N,AB,GG+1,GH,N+1,1,LB1B*LB2E*LB3,0)
      CALL ASG(AA+N+1,AB,LL,LM,1,1,LB1*LB2B*LB3B,0)
      CALL ASG(AA+N+1,AB,MN,MN,1,1,LB1*LB2B*LB3,2)
      CALL ASG(AA,AB,OO,OP,1,1,LB1*LB2*LB3E,1)
      CALL ASG(AA,AB,PP,PQ,1,1,LB1*LB2*LB3,0)
      CALL ASG(AA,AB,QQ,QR,N+1,1,LB1*LB2*LB3B,0)
      CALL ASG(AA+N,AB,RR,RS,N+1,1,LB1*LB2B*LB3,0)
      CALL ASG(AA,AA+N,SS,ST,1,1,LB1*LB2B*LB3E,0)
      CALL ASG(AA,AA+N,TT+1,TU,1,1,LB1*LB2B*LB3,0)
      CALL ASG(AB-N+1,AB,VV,VW,1,1,LB1B*LB2*LB3B,0)
      CALL ASG(AB-N,AB,WW,WX,1,1,LB1B*LB2*LB3,0)
      CALL ASG(AB-N,AB-N,XX,XX,1,1,LB1B*LB2*LB3B,0)
      CALL ASG(AB,AB,YY,YY,1,1,LB1B*LB2B*LB3,0)
      CALL ASG(BB,BC,AA,AB,1,1,LB1B*LB2B*MB3,0)
      CALL ASG(BB,BC,BB,BC,1,1,-1,0)
      CALL ASG(BB,BC,BB,BC,1,1,LB1B*LB2B*MB3B,3)
      CALL ASG(BB,BC,DD+N,DE,1,1,LB1B*LB2*MB3,1)
      CALL ASG(BB,BC,EE+N+1,EF,1,1,LB1B*LB2*MB3B,0)
      CALL ASG(BB,BC,FF+1,FG,N+1,1,LB1B*LB2*MB3,0)
      CALL ASG(BB+N,BC,GG+1,GH,N+1,1,LB1B*LB2E*MB3B,0)
      CALL ASG(BB+N+1,BC,LL,LM,1,1,LB1*LB2B*MB3,0)
      CALL ASG(BB+N+1,BC,MN,MN,1,1,LB1*LB2B*MB3B,3)
      CALL ASG(BB,BC,OO,OP,1,1,LB1*LB2*MB3,1)
      CALL ASG(BB,BC,PP,PQ,1,1,LB1*LB2*MB3B,0)
      CALL ASG(BB,BC,QQ,QR,N+1,1,LB1*LB2*MB3,0)
      CALL ASG(BB+N,BC,RR,RS,N+1,1,LB1*LB2B*MB3B,0)
      CALL ASG(BB,BB+N,SS,ST,1,1,LB1*LB2B*MB3,0)
      CALL ASG(BB,BB+N,TT+1,TU,1,1,LB1*LB2B*MB3B,0)
      CALL ASG(BC-N+1,BC,VV,VW,1,1,LB1B*LB2*MB3,0)
      CALL ASG(BC-N,BC,WW,WX,1,1,LB1B*LB2*MB3B,0)
      CALL ASG(BC-N,BC-N,XX,XX,1,1,LB1B*LB2*MB3,0)
      CALL ASG(BC,BC,YY,YY,1,1,LB1B*LB2B*MB3B,0)
      CALL ASG(CC,CD,AA,AB,1,N+1,LB1B*LB2B,0)
      CALL ASG(CC,CD,CC,CD,1,1,-1,0)
      CALL ASG(CC,CD,FF+1,FG,1,1,LB1B*LB2,0)
      CALL ASG(CC+1,CD,LL,LM,1,N+1,LB1*LB2B,0)
      CALL ASG(CC,CD,QQ,QR,1,1,LB1*LB2,0)
      CALL ASG(CC,CC,SS,SS,1,1,LB1*LB2B,0)
      CALL ASG(CD,CD,XX,XX,1,1,LB1B*LB2,0)
      IF(N.EQ.1) GO TO 1
      CALL ASG(DD,DE,AA,AB,1,1,LB1B*M2*LB3B,4)
    1 CALL ASG(DD,DE,BB,BC,1,1,LB1B*M2*LB3,2)
      CALL ASG(DD,DE,CC,CD,N,1,LB1B*M2*LB3B,0)
      CALL ASG(DD,DE,DD,DE,1,1,-1,0)
```

```
      IF(N.EQ.1) GO TO 2
      CALL ASG(DD,DE-N,DD+N,DE,1,1,LB1B*M2B*LB3B,7)
    2 CALL ASG(DD,DE-N,SE,EF,1,1,LB1B*M2B*LB3,5)
      CALL ASG(DD,DE,FF+1,FG,N,1,LB1B*M2B*LB3B,0)
      IF(N.EQ.1) GO TO 3
      CALL ASG(DD+N,DE,LL,LM,1,1,LB1*M2*LB3B,4)
    3 CALL ASG(DD+N,DE,MM,MN,1,1,LB1*M2*LB3,2)
      CALL ASG(DD+N,DE,NN,NC,N,1,LB1*M2*LB3B,0)
      IF(N.EQ.1) GO TO 4
      CALL ASG(DD,DE,QQ,QP,1,1,LB1*M2B*LB3B,7)
    4 CALL ASG(DD,DE,PP,PQ,1,1,LB1*M2B*LB3,2)
      CALL ASG(DD,DE,QQ,QR,N,1,LB1*M2B*LB3B,0)
      IF(N.EQ.1) GO TO 5
      CALL ASG(DD,DD+N-1,SS,ST,1,1,LB1*M2*LB3B,4)
    5 CALL ASG(DD,DD+N-1,TT,TU,1,1,LB1*M2*LB3,2)
      CALL ASG(DD,DD,UU,UU,1,1,LB1*M2*LB3B,0)
      IF(N.EQ.1) GO TO 6
      CALL ASG(DE-N+1,DE,VV,VW,1,1,LB1B*M2B*LB3B,7)
    6 CALL ASG(DE-N+1,DE,WW,WX,1,1,LB1B*M2B*LB3,2)
      CALL ASG(DE-N+1,DE-N+1,XX,XX,1,1,LB1B*M2B*LB3B,0)
      CALL ASG(EE,EF,AA,AB,1,1,LB1B*M2*M3,6)
      CALL ASG(EE,EF,BB,BC,1,1,LB1B*M2*M3B,0)
      CALL ASG(EE,EF,CC,CD,N+1,1,LB1B*M2*M3,0)
      CALL ASG(EE,EF,DD+N,DE,1,1,LB1B*M2B*M3,1)
      CALL ASG(EE,EF,EE,EF,1,1,-1,0)
      CALL ASG(EE,EF,EE+N+1,EF,1,1,LB1B*M2B*M3B,0)
      CALL ASG(EE,EF,FF+1,FG,N+1,1,LB1B*M2B*M3,0)
      CALL ASG(EE+N+1,EF,LL,LM,1,1,LB1*M2*M3,6)
      CALL ASG(EE+N+1,EF,MM,MN,1,1,LB1*M2*M3B,0)
      CALL ASG(EE+N+1,EF,NN,NC,N+1,1,LB1*M2*M3,0)
      CALL ASG(EE,EF,QQ,QP,1,1,LB1*M2B*M3,1)
      CALL ASG(EE,EF,PP,PQ,1,1,LB1*M2B*M3B,0)
      CALL ASG(EE,EF,QQ,QR,N+1,1,LB1*M2B*M3,0)
      CALL ASG(EE+1,EE+N,SS,ST,1,1,LB1*M2*M3,0)
      CALL ASG(EE,EE+N,TT,TU,1,1,LB1*M2*M3B,0)
      CALL ASG(EE,EE,UU,UU,1,1,LB1*M2*M3,0)
      CALL ASG(EF-N+1,EF,VV,VW,1,1,LB1B*M2B*M3,0)
      CALL ASG(EF-N,EF,WW,WX,1,1,LB1B*M2B*M3B,0)
      CALL ASG(EF-N,EF-N,XX,XX,1,1,LB1B*M2B*M3,0)
      CALL ASG(FF,FG,CC,CD,1,1,LB1B*M2,0)
      CALL ASG(FF,FG,FF,FG,1,1,-1,0)
      CALL ASG(FF,FG,FF+1,FG,1,1,LB1B*M2B,0)
      CALL ASG(FF+1,FG,NN,NC,1,1,LB1*M2,0)
      CALL ASG(FF,FG,QQ,QR,1,1,LB1*M2B,0)
      CALL ASG(FF,FF,UU,UU,1,1,LB1*M2,0)
      CALL ASG(FG,FG,XX,XX,1,1,LB1B*M2B,0)
      CALL ASG(GG,GH,AA,AB,1,N+1,LB1B*M3,0)
      CALL ASG(GG,GH,GG,GH,1,1,-1,0)
      CALL ASG(GG,GH,GG+1,GH,1,1,LB1B*M3B,0)
      CALL ASG(GG+1,GH,LL+N,LM,1,N+1,LB1*M3,0)
      CALL ASG(GG,GH,RR,RS,1,1,LB1*M3B,0)
      CALL ASG(GG,GG,ST,ST,1,1,LB1*M3,0)
      CALL ASG(GH,GH,YY,YY,1,1,LB1B*M3B,0)
      CALL ASG(HH+1,HI,AA,AA+N,1,1,LB1B*LB3B,0)
      CALL ASG(HH,HI,BB,BB+N,1,1,LB1B*LB3,0)
      CALL ASG(HH,HH,CC,CC,1,1,LB1B*LB3B,0)
      CALL ASG(HH,HI,HH,HI,1,1,-1,0)
      CALL ASG(HH+1,HI,SS,ST,1,1,LB1*LB3B,0)
      CALL ASG(HH,HI,TT,TU,1,1,LB1*LB3,0)
      CALL ASG(HH,HH,UU,UU,1,1,LB1*LB3B,0)
      CALL ASG(II+1,IJ,AA,AA+N,1,1,LB1B*M3,0)
      CALL ASG(II,IJ,BB,BB+N,1,1,LB1B*M3B,0)
      CALL ASG(II,II,CC,CC,1,1,LB1B*M3,0)
      CALL ASG(II,IJ,II,IJ,1,1,-1,0)
      CALL ASG(II+1,IJ,SS,ST,1,1,LB1*M3,0)
      CALL ASG(II,IJ,TT,TU,1,1,LB1*M3B,0)
```

```
      CALL ASG(II,II,UU,UU,1,1,LB1*M3,0)
      CALL ASG(KK,KK,CC,CC,1,1,LB1B,0)
      CALL ASG(KK,KK,KK,KK,1,1,-1,0)
      CALL ASG(KK,KK,UU,UU,1,1,LB1,0)
      CALL ASG(LL+N+1,LM,AA,AB,1,1,M1*LB2B*LB3B,0)
      CALL ASG(LL+N+1,LM-1,BB,BC,1,1,M1*LB2B*LB3,2)
      CALL ASG(LL,LM,DD,DE,1,1,M1*LB2*LB3B,1)
      CALL ASG(LL,LM,EE,EF,1,1,M1*LB2*LB3,0)
      CALL ASG(LL,LM,FF,FG-1,N+1,1,M1*LB2*LB3B,0)
      CALL ASG(LL+N,LM,GG,GH,N+1,1,M1*LB2B*LB3,0)
      CALL ASG(LL,LL+N,HH,HI,1,1,M1*LB2B*LB3B,0)
      CALL ASG(LL,LL+N,II+1,IJ,1,1,M1*LB2B*LB3,0)
      CALL ASG(LL+N+1,LM,LL,LM,1,1,M1B*LB2B*LB3B,0)
      CALL ASG(LL,LM,LL,LM,1,1,-1,0)
      CALL ASG(LL+N+1,LM-1,MM,MN,1,1,M1B*LB2B*LB3,2)
      CALL ASG(LL,LM,OO,OP,1,1,M1B*LB2*LB3B,1)
      CALL ASG(LL,LM,PP,PQ,1,1,M1B*LB2*LB3,0)
      CALL ASG(LL,LM,QQ,QR-1,N+1,1,M1B*LB2*LB3B,0)
      CALL ASG(LL+N,LM,RR,RS,N+1,1,M1B*LB2B*LB3,0)
      CALL ASG(LL,LL+N,SS,ST,1,1,M1B*LB2B*LB3B,0)
      CALL ASG(LL,LL+N,TT+1,TU,1,1,M1B*LB2B*LB3,0)
      CALL ASG(MM+N+1,MN,AA,AB,1,1,M1*LB2B*M3,0)
      CALL ASG(MM+N+1,MN-1,BB,BC,1,1,M1*LB2B*M3B,2)
      CALL ASG(MM,MN,DD,DE,1,1,M1*LB2*M3,1)
      CALL ASG(MM,MN,EE,EF,1,1,M1*LB2*M3B,0)
      CALL ASG(MM,MN,FF,FG-1,N+1,1,M1*LB2*M3,0)
      CALL ASG(MM+N,MN,GG,GH,N+1,1,M1*LB2B*M3B,0)
      CALL ASG(MM,MM+N,HH,HI,1,1,M1*LB2B*M3,0)
      CALL ASG(MM,MM+N,II+1,IJ,1,1,M1*LB2B*M3B,0)
      CALL ASG(MM+N+1,MN,LL,LM,1,1,M1B*LB2B*M3,0)
      CALL ASG(MM+N+1,MN-1,MM,MN,1,1,M1B*LB2B*M3B,2)
      CALL ASG(MM,MN,MM,MN,1,1,-1,0)
      CALL ASG(MM,MN,OO,OP,1,1,M1B*LB2*M3,1)
      CALL ASG(MM,MN,PP,PQ,1,1,M1B*LB2*M3B,0)
      CALL ASG(MM,MN,QQ,QR-1,N+1,1,M1B*LB2*M3,0)
      CALL ASG(MM+N,MN,RR,RS,N+1,1,M1B*LB2B*M3B,0)
      CALL ASG(MM,MM+N,SS,ST,1,1,M1B*LB2B*M3,0)
      CALL ASG(MM,MM+N,TT+1,TU,1,1,M1B*LB2B*M3B,0)
      IF(N.EQ.1) GO TO 7
      CALL ASG(NN+1,NO,AA,AB,1,N+1,M1*LB2B,0)
    7 CALL ASG(NN,NO,FF,FG,1,1,M1*LB2,0)
      CALL ASG(NN,NN,HH,HI,1,1,M1*LB2B,0)
      IF(N.EQ.1) GO TO 8
      CALL ASG(NN+1,NO,LL,LM,1,N+1,M1B*LB2B,0)
    8 CALL ASG(NN,NO,NN,NO,1,1,-1,0)
      CALL ASG(NN,NO,QQ,QR,1,1,M1B*LB2,0)
      CALL ASG(NN,NN,SS,SS,1,1,M1B*LB2B,0)
      IF(N.EQ.1) GO TO 9
      CALL ASG(OO+N,OP,AA,AB,1,1,M1*M2*LB3B,4)
    9 CALL ASG(OO+N,OP,BB,BC,1,1,M1*M2*LB3,2)
      CALL ASG(OO+N,OP,CC,CC-1,N,1,M1*M2*LB3B,0)
      IF(N.EQ.1) GO TO 10
      CALL ASG(OO,OP,DD,DE,1,1,M1*M2B*LB3B,7)
   10 CALL ASG(OO,OP,EE,EF,1,1,M1*M2B*LB3,2)
      CALL ASG(OO,OP,FF,FG,N,1,M1*M2B*LB3B,0)
      IF(N.EQ.1) GO TO 11
      CALL ASG(OO,OO+N-1,HH,HI,1,1,M1*M2*LB3B,4)
   11 CALL ASG(OO,OO+N-1,II,IJ,1,1,M1*M2*LB3,0)
      CALL ASG(OO,OO,KK,KK,1,1,M1*M2*LB3B,0)
      IF(N.EQ.1) GO TO 12
      CALL ASG(OO+N,OP,LL,LM,1,1,M1B*M2*LB3B,4)
   12 CALL ASG(OO+N,OP,MM,MN,1,1,M1B*M2*LB3,2)
      CALL ASG(OO+N,OP,NN,NO,N,1,M1B*M2*LB3B,0)
      IF(N.EQ.1) GO TO 13
      CALL ASG(OO,OP,OO,OP,1,1,M1B*M2B*LB3B,7)
   13 CALL ASG(OO,OP,OO,OP,1,1,-1,0)
```

```
      CALL ASG(CO,OP,PP,PC,1,1,M1B*M2B*LB3,2)
      CALL ASG(CC,CP,CC,CR,N,1,M1B*M2B*LB3B,0)
      IF(N.EQ.1) GO TO 14
      CALL ASG(CO,CO+N-1,SS,ST,1,1,M1B*M2*LB3B,4)
   14 CALL ASG(CO,CC+N-1,TT,TU,1,1,M1B*M2*LB3,0)
      CALL ASG(CO,CO,UU,UU,1,1,M1B*M2*LB3B,0)
      CALL ASG(PP+N+1,PQ,AA,AB,1,1,M1*M2*M3,6)
      CALL ASG(PP+N+1,PQ,BB,BC,1,1,M1*M2*M3B,0)
      CALL ASG(PP+N+1,PQ,CC,CD-1,N+1,1,M1*M2*M3,0)
      CALL ASG(PP,PQ,DD,DE,1,1,M1*M2B*M3,1)
      CALL ASG(PP,PQ,EE,EF,1,1,M1*M2B*M3B,0)
      CALL ASG(PP,PQ,FF,FG,N+1,1,M1*M2B*M3,0)
      CALL ASG(PP+1,PP+N,HH,HI,1,1,M1*M2*M3,0)
      CALL ASG(PP,PP+N,II,IJ,1,1,M1*M2*M3B,0)
      CALL ASG(PP,PP,KK,KK,1,1,M1*M2*M3,0)
      CALL ASG(PP+N+1,PQ,LL,LM,1,1,M1B*M2*M3,6)
      CALL ASG(PP+N+1,PQ,MM,MN,1,1,M1B*M2*M3B,0)
      CALL ASG(PP+N+1,PQ,NN,NC,N+1,1,M1B*M2*M3,0)
      CALL ASG(PP,PQ,CO,CP,1,1,M1B*M2B*M3,1)
      CALL ASG(PP,PC,PP,PC,1,1,M1B*M2B*M3B,0)
      CALL ASG(PP,PC,CC,CR,N+1,1,M1B*M2B*M3,0)
      CALL ASG(PP+1,PP+N,SS,ST,1,1,M1B*M2*M3,0)
      CALL ASG(PP,PP+N,TT,TU,1,1,M1B*M2*M3B,0)
      CALL ASG(PP,PP,UU,UU,1,1,M1B*M2*M3,0)
      CALL ASG(QQ+1,QR,CC,CD,1,1,M1*M2,0)
      CALL ASG(QC,QR,FF,FG,1,1,M1*M2B,0)
      CALL ASG(QQ,QQ,KK,KK,1,1,M1*M2,0)
      CALL ASG(QQ+1,QR,NN,NC,1,1,M1B*M2,0)
      CALL ASG(QC,QR,CC,QR,1,1,M1B*M2B,0)
      CALL ASG(QC,QC,UU,UU,1,1,M1B*M2,0)
      CALL ASG(RR+1,RS,AA+N,AB,1,N+1,M1*M3,0)
      CALL ASG(RR,RS,GG,GH,1,1,M1*M3B,0)
      CALL ASG(RR,RR,HI,HJ,1,1,M1*M3,0)
      CALL ASG(RR+1,RS,LL+N,LM,1,N+1,M1B*M3,0)
      CALL ASG(RR,RS,RR,RS,1,1,M1B*M3B,0)
      CALL ASG(RR,RR,ST,ST,1,1,M1B*M3,0)
      CALL ASG(SS+1,ST,HH,HI,1,1,M1*LB3B,0)
      CALL ASG(SS,ST,II,IJ,1,1,M1*LB3,0)
      CALL ASG(SS,SS,KK,KK,1,1,M1*LB3B,0)
      CALL ASG(SS+1,ST,SS,ST,1,1,M1B*LB3B,0)
      CALL ASG(SS,ST,SS,ST,1,1,-1,0)
      CALL ASG(SS,ST,TT,TU,1,1,M1B*LB3,0)
      CALL ASG(SS,SS,UU,UU,1,1,M1B*LB3B,0)
      CALL ASG(TT+1,TU,HH,HI,1,1,M1*M3,0)
      CALL ASG(TT,TU,II,IJ,1,1,M1*M3B,0)
      CALL ASG(TT,TT,KK,KK,1,1,M1*M3,0)
      CALL ASG(TT+1,TU,SS,ST,1,1,M1B*M3,0)
      CALL ASG(TT,TU,TT,TU,1,1,M1B*M3B,0)
      CALL ASG(TT,TT,UU,UU,1,1,M1B*M3,0)
      CALL ASG(UU,UU,KK,KK,1,1,M1,0)
      CALL ASG(UU,UU,UU,UU,1,1,M1B,0)
      IF(N.EQ.1) GO TO 15
      CALL ASG(VV,VW,AB-N,AB,1,1,M2*LB3B,4)
   15 CALL ASG(VV,VW,BC-N,BC,1,1,M2*LB3,0)
      CALL ASG(VV,VV,CD,CD,1,1,M2*LB3B,0)
      IF(N.EQ.1) GO TO 16
      CALL ASG(VV,VW,VV,VW,1,1,M2B*LB3B,7)
   16 CALL ASG(VV,VW,VV,VW,1,1,-1,0)
      CALL ASG(VV,VW,WW,WX,1,1,M2B*LB3,0)
      CALL ASG(VV,VV,XX,XX,1,1,M2B*LB3B,0)
      CALL ASG(WW+1,WX,AB-N,AB,1,1,M2*M3,0)
      CALL ASG(WW,WX,BC-N,BC,1,1,M2*M3B,0)
      CALL ASG(WW,WW,CD,CD,1,1,M2*M3,0)
      CALL ASG(WW+1,WX,VV,VW,1,1,M2B*M3,0)
      CALL ASG(WW,WX,WW,WX,1,1,M2B*M3B,0)
      CALL ASG(WW,WW,XX,XX,1,1,M2B*M3,0)
```

```
      CALL ASG(XX,XX,CD,CD,1,1,M2,0)
      CALL ASG(XX,XX,XX,XX,1,1,M2B,0)
      CALL ASG(YY,YY,AB,AB,1,1,M3,0)
      CALL ASG(YY,YY,YY,YY,1,1,M3B,0)
C SORTING OF ARRAYS ICOOR AND T
      LAST=IS
      IPT=MSIZE+1
      DO 150 K=2,MSIZE-1
      IS=IPT-1
  160 IS=IS+1
      IF(IS.GT.LAST) GO TO 150
      IF(ICOOR(IS,1).NE.K) GO TO 160
      IDUM1=ICOOR(IPT,1)
      IDUM2=ICOOR(IPT,2)
      DUM3=T(IPT)
      ICOOR(IPT,1)=ICOOR(IS,1)
      ICOOR(IPT,2)=ICOOR(IS,2)
      T(IPT)=T(IS)
      ICOOR(IS,1)=IDUM1
      ICOOR(IS,2)=IDUM2
      T(IS)=DUM3
      IPT=IPT+1
      GO TO 160
  150 CONTINUE
C COMPUTATION OF STEADY STATE PROBABILITIES
      NI=0
  800 IS=1
      DO 200 K=1,MSIZE
      SUM=.0
      ICNT=0
  400 IE=ICOOR(IS,1)
      JE=ICOOR(IS,2)
      IF(ICNT.NE.0) GO TO 250
      IF(IE.EQ.JE) GO TO 300
  250 SUM=SUM+T(IS)*X(JE)
      GO TO 350
  300 DEN=T(IS)
      ICNT=1
      IF(IS.EQ.LAST) GO TO 450
  350 IS=IS+1
      IF(ICOOR(IS,1).EQ.K) GO TO 400
  450 X(K)=(-SUM+PHS(K))/DEN
      DELTA=X(K)-PX(K)
      CPERC=ABS(DELTA/X(K))
      IF(CPERC.GT.1.E-05) GO TO 200
      IX(K)=1
  200 CONTINUE
      NI=NI+1
      IE=0
      DO 650 I=1,MSIZE
      IF(IX(I).EQ.1) GO TO 650
      IE=1
  650 CONTINUE
      IF(IE.EQ.0) GO TO 700
      IF(NI.GT.500) GO TO 999
      DO 750 I=1,MSIZE
  750 PX(I)=X(I)
      GO TO 800
  700 SSSUM=.0
      WRITE(6,98)
      WRITE(6,97)
      WRITE(6,96)
   98 FORMAT(" BEHAVIOUR OF A 3-STAGE TRANSFER LINE WITH")
   97 FORMAT(" UNRELIABLE MACHINES AND INTERSTAGE BUFFERS")
   96 FORMAT(80("*"))
      WRITE(6,95) LB1
```

```
      WRITE(6,94) LB2
      WRITE(6,93) LB3
      WRITE(6,92) M1
      WRITE(6,91) M2
      WRITE(6,90) M3
   95 FORMAT(4X,"FAILURE  RATE  OF  STAGE  1 = ",F5.4)
   94 FORMAT(4X,"FAILURE  RATE  OF  STAGE  2 = ",F5.4)
   93 FORMAT(4X,"FAILURE  RATE  OF  STAGE  3 = ",F5.4)
   92 FORMAT(13X,"REPAIR  RATE  OF  STAGE  1 = ",F5.4)
   91 FORMAT(13X,"REPAIR  RATE  OF  STAGE  2 = ",F5.4)
   90 FORMAT(13X,"REPAIR  RATE  OF  STAGE  3 = ",F5.4)
      WRITE(6,89)
   89 FORMAT(50("_"))
      DO 888 I=1,MSIZE
      SSSUM=SSSUM+X(I)
  888 WRITE(6,88) I,X(I)
   88 FORMAT(" X(",I2,") =",F16.14)
      WRITE(6,99)
   99 FORMAT(50("_"))
      WRITE(6,87)SSSUM
   87 FORMAT(" SUM  = ",F16.14)
C COMPUTATION OF SYSTEM PERFORMANCE MEASURES
      PSUM=.0
      DO 801 I=1,AB
  801 PSUM=PSUM+X(I)
      DO 802 I=1,N*(N+1)
  802 PSUM=PSUM+X(CD+I)
      DO 803 I=1,N+1
  803 PSUM=PSUM+X(GH+I)
      DO 804 I=1,N*(N+1)
  804 PSUM=PSUM+X(KK+I)
      DO 805 I=1,N*(N+1)
  805 PSUM=PSUM+X(NO+I)
      DO 806 I=1,N+1
  806 PSUM=PSUM+X(RS+I)
      DO 807 I=1,N
  807 PSUM=PSUM+X(UU+I)
      PRATE=LB3B*PSUM
      L=0
      AVINV1=.0
      AVINV2=.0
      REWIND 2
  900 READ(2,86,END=850) INV1,INV2
   86 FORMAT(2I1)
      L=L+1
      AVINV1=AVINV1+INV1*X(L)
      AVINV2=AVINV2+INV2*X(L)
      GO TO 900
  850 WRITE(6,85)
   85 FORMAT(50("_"))
      WRITE(6,84) PRATE
   84 FORMAT("+    PRODUCTION RATE  = ",F16.14)
      EFF=1./(1.+LB3/M3)
      WRITE(6,83)EFF
   83 FORMAT("+ ISOLATED EFFICIENCY = ",F16.14)
      WRITE(6,82)
   82 FORMAT(50("_"))
      WRITE(6,81) AVINV1
   81 FORMAT("+ AVERAGE INVENTORY LEVEL 1  = ",F16.14)
      WRITE(6,80) AVINV2
   80 FORMAT("+ AVERAGE INVENTORY LEVEL 2  = ",F16.14)
      WRITE(6,79)
   79 FORMAT(50("_"))
      FRINV1=AVINV1/N
      FRINV2=AVINV2/N
      WRITE(6,78)FRINV1
```

```
   78 FORMAT("+ AVERAGE INVENTORY LEVEL 1 (AS FRACTION OF N) =",
     #        F16.14)
      WRITE(6,77)FRINV2
   77 FORMAT("+ AVERAGE INVENTORY LEVEL 2 (AS FRACTION OF N) =",
     #        F16.14)
      WRITE(6,76)
   76 FORMAT(50("_"))
  999 STOP
      END
      SUBROUTINE ASG(MB,ME,NB,NE,MDEL,NDEL,ELM,FLG)
      COMMON N,IS,ICOOR(5000,2),T(5000)
      INTEGER FLG
      IF((MB.GT.ME).OR.(NB.GT.NE)) GO TO 120
      INCM=-MDEL
      INCN=-NDEL
   80 INCM=INCM+MDEL
      INCN=INCN+NDEL
      I=MB+INCM
      J=NB+INCN
      IF(FLG.EQ.0) GO TO 10
      IF(FLG.EQ.1) GO TO 20
      IF(FLG.EQ.2) GO TO 30
      IF(FLG.EQ.3) GO TO 40
      IF(FLG.EQ.4) GO TO 50
      IF(FLG.EQ.5) GO TO 60
      IF(INCM.NE.0) GO TO 70
  100 INCN=INCN-1
      GO TO 80
   70 IF(FLG.EQ.6) GO TO 90
      IF(MOD(INCM,N).EQ.0) GO TO 80
      GO TO 10
   90 IF(MOD(INCM,N+1).EQ.0) GO TO 80
      GO TO 10
   60 IF(MOD(INCM,N+1).NE.0) GO TO 10
  110 INCM=INCM-1
      GO TO 80
   50 IF(INCM.EQ.0) GO TO 100
      IF(MOD(INCM,N).NE.0) GO TO 10
      INCM=INCM+1
      GO TO 80
   40 IF(INCM.EQ.0) GO TO 110
      IF(MOD(INCM,N+1).EQ.0) GO TO 80
      GO TO 10
   30 IF(MOD(INCM+1,N).NE.0) GO TO 10
      INCM=INCM+1
      GO TO 10
   20 IF(MOD(INCM,N+1).NE.0) GO TO 10
      GO TO 100
   10 IF(J.EQ.1) GO TO 130
      IS=IS+1
      ICOOR(IS,1)=J
      ICOOR(IS,2)=I
      IF(I.EQ.J) GO TO 140
      T(IS)=ELM
C     WRITE(6,99) IS,J,I,T(IS)
      GO TO 130
  140 T(IS)=ELM-1.
C     WRITE(6,99) IS,J,I,T(IS)
C  99 FORMAT(I5,2X,I3,2X,I3,2X,F12.8)
  130 IF((I.EQ.ME).OR.(J.EQ.NE)) GO TO 120
      GO TO 80
  120 RETURN
      END
16.15.34.UCLP, BU, P03     ,       0.562KLNS.
```

BEHAVIOUR OF A 3-STAGE TRANSFER LINE WITH
UNRELIABLE MACHINES AND INTERSTAGE BUFFERS

**********

FAILURE RATE OF STAGE 1 = .0200
FAILURE RATE OF STAGE 2 = .0200
FAILURE RATE OF STAGE 3 = .0300
REPAIR RATE OF STAGE 1 = .2000
REPAIR RATE OF STAGE 2 = .2000
REPAIR RATE OF STAGE 3 = .2000

---

X( 1) = .176330086844 91
X( 2) = .026453373481 57
X( 3) = .026913847188 33
X( 4) = .028102801216 11
X( 5) = .002445490819 33
X( 6) = .002545499416 95
X( 7) = .202625571551 91
X( 8) = .017682268812 43
X( 9) = .221735080302 29
X(10) = .001090564503 34
X(11) = .093921293660 7
X(12) = .050727245136 7
X(13) = .000208524574 9
X(14) = .008476699996 4
X(15) = .000739664129 64
X(16) = .001659394239 6
X(17) = .006125827025 11
X(18) = .005348203731 37
X(19) = .010876496211 16
X(20) = .000870913496 85
X(21) = .012520720113 72
X(22) = .000112220897 78
X(23) = .000141386887 7
X(24) = .005480684322 8
X(25) = .000549581156 45
X(26) = .004999037538 65
X(27) = .000692173526 2
X(28) = .000021102134 5
X(29) = .000002547543 12
X(30) = .000023937976 6
X(31) = .001098129126 6
X(32) = .001166057920 4
X(33) = .001120217846 9
X(34) = .000087371100 26
X(35) = .000896111470 3
X(36) = .000084786227 34
X(37) = .001018953256 9
X(38) = .003704502280 23
X(39) = .004035113844 46
X(40) = .000145893272 47
X(41) = .004946999221 54
X(42) = .004648359728 57
X(43) = .001461357843 49
X(44) = .000749182261 12
X(45) = .000704018771 16
X(46) = .000159790212 12
X(47) = .000171455714 35
X(48) = .000159087277 73
X(49) = .009602583841 46
X(50) = .003784279074 40
X(51) = .000343897778 70
X(52) = .003538547387 71
X(53) = .004112728118 45
X(54) = .000360614334 77
X(55) = .004383458409 84
X(56) = .000002073786 34

```
 X(57) =  .0011696687102
 X(58) =  .0010256976676
 X(59) =  .0006019846042?
 X(60) =  .0012378748812
 X(61) =  .0016815295451
 X(62) =  .0010961025157
 X(63) =  .0011643820129
 X(64) =  .0009598344945
 X(65) =  .0006741092836
 X(66) =  .0006143401437
 X(67) =  .0007114384839
 X(68) =  .0006476740766
 X(69) =  .0008816386715
 X(70) =  .0001064377089
 X(71) =  .0001032027399
 X(72) =  .0001022187355
 X(73) =  .0001063856614
 X(74) =  .0001061555928
 X(75) =  .0001063429294
 X(76) =  .0001042504796
 X(77) =  .0001033902664
 X(78) =  .0001118542186
 X(79) =  .0004821813651
 X(80) =  .0005153951677?6
 X(81) =  .0005493944885?
 X(82) =  .0006812731867?
 X(83) =  .0007310531585?
 X(84) =  .0008017050432?
 X(85) =  .0942692254856
 X(86) =  .0353509564652
 X(87) =  .0334823588756
 X(88) =  .0064267354889
 X(89) =  .0079535747622
 X(90) =  .0073942999492
 X(91) =  .0386318926?625
 X(92) =  .0393663941517
 X(93) =  .0436873912089
 X(94) =  .0078550586999
 X(95) =  .0074758884528
 X(96) =  .0074282773198
 X(97) =  .5418142963213
 X(98) =  .7071038525873

 SUM  = 1.0000096443577

+     PRODUCTION RATE  =  .728403591299?9
+ ISOLATED EFFICIENCY =  .869506217391?0

+ AVERAGE INVENTORY LEVEL 1  = 1.2817925009471?3
+ AVERAGE INVENTORY LEVEL 2  =  .7895152554496?

+ AVERAGE INVENTORY LEVEL 1 (AS FRACTION OF N) = .6408962502359?
+ AVERAGE INVENTORY LEVEL 2 (AS FRACTION OF N) = .3947576277248?

11.35.52.UCLP, BU, P04     ,       0.157KLNS.
```

# APPENDIX III

```
{ THIS COMPUTER PROGRAM SIMULATES MULTISTAGE TRANSFER LINES.
  NUMBER OF STAGES IS LIMITED BY 20 AND NUMBER OF BUFFERS IS
  LIMITED BY 19.
  UP- AND DOWNTIME DISTRIBUTIONS ARE ASSUMED EXPONENTIAL.
  CYCLE TIME IS EQUAL TO UNIT TIME.}
PROGRAM TRANSSIM (INPUT,f1);

TYPE  machine=1..20 ;buffer=1..19;
      state=(idle,up,down);
      data=record     Tc,Td,F,DT,time,uptime: real;
                      S:state ;
                      N,C,FC:integer;
                      case0:0..1
           end;

VAR   f1:text;
      sinf,org:char;
      sd,dur:integer;
      M,k:machine;
      clock,exclock,finish,delta,newup,dum:real;
      B,l,ML:array [buffer] of integer ;CIL:array [buffer] of real ;
      T:array [machine] of data ;

FUNCTION uniform(var seed:integer):real;
  begin
       uniform:=seed/32765;
       seed:=(25173*seed+13849) mod 32765;
  end;

FUNCTION expon(down:real):integer;
  begin
       expon:=trunc(-down*Ln(1.0-uniform(sd)))+1;
  end;

FUNCTION OK(a:machine):boolean;
  begin
   if a=1
    then   OK:=l[1]<=B[1]
    else
        if a=M
           then OK:=l[M-1]>0
           else OK:=(l[a]<=B[a]) and (l[a-1]>0)
   end;

PROCEDURE forceddown(b:machine);
 begin
      T[b].FC:=T[b].FC+1;
      if b=1
         then case T[b+1].S of
                up:T[b].time:=clock+1
              else T[b].time:=T[b+1].time+1
              end
         else if b=M then case T[b-1].S of
                            up:T[b].time:=clock+1
                          else T[b].time:=T[b-1].time+1
                          end
                     else if l[b-1]=0
                        then case T[b-1].S of
                               up:T[b].time:=clock+1
                             else T[b].time:=T[b-1].time+1
                             end
                        else case T[b+1].S of
```

```
                                        else T[b].time:=T[b+1].time+1
                                        end;
 end;

PROCEDURE rearrange;
type cofac=0..1;
var x,y:machine;
      cf:cofac;
 begin
  cf:=0;
  x:=k-1;
  while (x<>0) and (cf=0) do
   begin
         if l[x]=B[x]+1
           then forceddown(x)
           else cf:=1;
         x:=x-1;
   end;
  cf:=0;
  y:=k;
  while (y<>M) and (cf=0) do
   begin if l[y]=0
           then forceddown(y+1)
           else cf:=1;
          y:=y+1;
   end;
 end;

PROCEDURE initialize;
 begin
  write('number of stages        : ');readln(M);
  write('simulation time limit   : ');readln(finish);
  write('seed                    : ');readln(sd);
  write('origin state?Y(es)/N(o): ');readln(org);
  for k:=1 to M do with T[k] do
    begin writeln;writeln (' STAGE ',k);writeln;
         Tc:=1;
         write(' mean down time   : ');readln(Td);
         write(' failure rate     : ');readln(F);
         if(org='N') then begin write(' initial state    : ');readln(sin
                                write(' remaining time   : ');
                           end
                     else sinf:='I';
         case sinf of
              'I':begin if (org='Y') then time:=k-1
                                     else readln(time);
                        uptime:=time;DT:=0;
                        S:=idle
                   end;
              '1':begin readln(uptime);
                        time:=0;DT:=0;
                        S:=up
                   end;
              '0':begin readln(time);
                        uptime:=time;DT:=time;
                        S:=down
                   end;
         end;
         C:=0;N:=0;FC:=0;case0:=0;
         if k<M then
           begin writeln;write(' maximum buffer capacity  : ');
                 readln(B[k]);writeln;
                 write(' initial inventory level  : ');
                 readln(l[k]);if (org='N') then l[k]:=l[k]+1;writeln;
                 CIL[k]:=0;ML[k]:=0
```

```
    end;
  clock:=0;exclock:=0;delta:=1;dum:=uniform(sd)
 end;

PROCEDURE boundevents;
 begin
  for k:=1 to M do with T[k] do
   begin if time=clock then
        case S of
            up : begin S:=idle; if k<>1 then l[k-1]:=l[k-1]-1;
                                if k<>M then l[k]:=l[k]+1
                 end;
          down : begin newup:=expon(1/F)-1;writeln(k,'dwn>up',newup:5:0);
                       if newup=0 then begin S:=idle;case0:=1 end
                                  else begin S:=up;C:=C+1; {no scraped item
                                             time:=clock+Tc;
                                             uptime:=uptime+newup
                                       end;
                 end;
          idle : begin newup:=expon(1/F)-1;writeln(k,'fcd>up',newup:5:0);
                       uptime:=time+newup;if (newup=0) then case0:=1;
                 end;
        end;
    end;
 end;

PROCEDURE invent;
begin
  for k:=1 to M-1 do
   if l[k]>1 then begin
                   CIL[k]:=CIL[k]+delta*(l[k]-1);
                   if l[k]>ML[k] then ML[k]:=l[k]-1;
                  end
end;

PROCEDURE conditionalevents;
 begin
  for k:=1 to M do with T[k] do
      if (S=idle) and (time=clock)
        then
        if OK(k) then begin if uptime=clock
                            then begin S:=down;
                                       dur:=expon(Td);writeln(k,'durat
ur);
                                       time:=clock+dur;
                                       DT:=DT+dur;N:=N+1;
                                       uptime:=uptime+dur;
                                       if case0=1
                                         then begin rearrange;case0:=0
                                 end
                            else begin S:=up;C:=C+1;
                                       time:=clock+Tc
                                 end;
                      end
                 else  begin writeln(time:5:0);
                             forceddown(k);writeln(k,'. ',FC,'!',time:5:0
                       end;
 end;

PROCEDURE setclock;
 begin
      exclock:=clock;
      clock:=1E10;
      for k:=1 to M do with T[k] do
         if (S<>idle) and (time<clock)
```

```
                    case S of
                              down: delta:=time-exclock;
                              up: delta:=1;
                    end;
          end;
nd;

OCEDURE report;
egin
writeln(fl,'CLOCK=',clock:5:0);
writeln(fl,'************');
for k:=1 to M do with T[k] do
 begin writeln(fl,'STAGE ',k,':');
  writeln(fl,'************');
  writeln(fl,' mean  up -time : ',(1/F):5:2,' ,  failure rate : ',F:1:5);
  writeln(fl,' mean down-time : ',Td:6:2,' ,  repair rate  : ',1/Td:1:5);
  writeln(fl,' ':11,'**',' ':18,'**');
  if k<M then begin
              writeln(fl);
              writeln(fl,'   storage capacity              :  ',B[k]:5);
              writeln(fl,'   average storage fill          :  ',CIL[k]/fi
:2);
              writeln(fl,'   max num of items in buffer    :  ',ML[k]:5);
              l[k]:=l[k]-1;
              writeln(fl,'   final inventory level         :  ',l[k]:5);
             end
       else begin writeln(fl,' no limitation on the buffer ');writeln(f
             end;
  writeln(fl,' ':11,'**',' ':18,'**');
  writeln(fl,' uptime           : ',C:10,'      {E(N)=',F*finish:5:0,'}');
  writeln(fl,' downtime         : ',DT:10:0,'         {N= ',N:5,' }');
  writeln(fl,' idletime         : ',finish-(C+DT):10:0);
  writeln(fl,FC,'-times forceddown');
  writeln(fl,' ___________________________________________________ ');
  writeln(fl,'!                        efficiency : ',C/finish*100:8:6,' %!');
  writeln(fl,'!theoretical(isolated)efficiency : ',1/(1+F*Td)*100:8:6,' %
  writeln(fl,'!_____________________________________________________!');
  if S=up
    then writeln(fl,'final state : *up*,  remtime :',uptime-finish:4:0)
    else if S=down
          then writeln(fl,'final state :*down*,  remtime :',time-finish:4
          else writeln(fl,'final state :*idle*,  remtime :',time-finish:4
  writeln(fl,'++++++++++++++++++++++++++++++++++++++++');
 end;
end;

EGIN
    initialize;
    repeat
          boundevents;
          invent;
          conditionalevents;
          setclock
    until clock>=finish;
    if delta>1 then begin delta:=finish-exclock;invent end;
    assign(fl,'sonuc.pas');rewrite(fl);
    report;
    close (fl)
ND.
```

```
CLOCK=10000
**********
STAGE 1:
**********
 mean  up -time : 500.00 ,  failure rate : 0.00200
 mean down-time :  25.00 ,  repair rate  : 0.04000
           **                    **

   storage capacity                :    100
   average storage fill            :     82.95
   max num of items in buffer      :    100
   final inventory level           :      0
           **                    **
 uptime            :       7739       (E(N)=    20)
 downtime          :        478         (N=     13 )
 idletime          :       1783
47-times forceddown

 _____________________________________________
!                      efficiency : 77.390000 %!
!theoretical(isolated)efficiency : 95.238095 %!
!_____________________________________________!
final state : *up*,  remtime : 121
++++++++++++++++++++++++++++++++++++++++
STAGE 2:
**********
 mean  up -time : 333.33 ,  failure rate : 0.00300
 mean down-time :  30.00 ,  repair rate  : 0.03333
           **                    **

   storage capacity                :    100
   average storage fill            :     65.16
   max num of items in buffer      :    100
   final inventory level           :      3
           **                    **
 uptime            :       7738       (E(N)=    30)
 downtime          :       1107         (N=     31 )
 idletime          :       1155
28-times forceddown

 _____________________________________________
!                      efficiency : 77.380000 %!
!theoretical(isolated)efficiency : 91.743119 %!
!_____________________________________________!
final state : *up*,  remtime : 573
++++++++++++++++++++++++++++++++++++++++
STAGE 3:
**********
 mean  up -time : 200.00 ,  failure rate : 0.00500
 mean down-time :  40.00 ,  repair rate  : 0.02500
           **                    **
 no limitation on the buffer

           **                    **
 uptime            :       7734       (E(N)=    50)
 downtime          :       1892         (N=     46 )
 idletime          :        374
10-times forceddown

 _____________________________________________
!                      efficiency : 77.340000 %!
!theoretical(isolated)efficiency : 83.333333 %!
!_____________________________________________!
final state : *up*,  remtime : 255
++++++++++++++++++++++++++++++++++++++++
```