Analysis of locking policies in database management systems

D. Potier, P. Leblanc

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105 78150 Le Chesnay
France
Tél.: 954 90 20

Rapports de Recherche

N° 23

ANALYSIS OF LOCKING POLICIES IN DATABASE MANAGEMENT SYSTEMS

Dominique POTIER
Philippe LEBLANC

Juillet 1980

ANALYSIS OF LOCKING POLICIES IN DATABASE MANAGEMENT SYSTEMS

Dominique POTIER
Philippe LEBLANC

* INRIA - Domaine de Voluceau - BP 105 - 78150 Le Chesnay - FRANCE

RESUME

L'accès concurrent de plusieurs transactions à un système de base de données impose la mise en place d'un mécanisme de contrôle de cohérence. Le contrôle est généralement réalisé en partageant la base de données en unités de verrouillage ou granules, et en définissant des règles de verrouillage des granules qui assure l'intégrité de l'information. Toutefois, le maintien de l'intégrité par ces mécanismes se fait au détriment des performances globales du système. Ceci est dû aux restrictions d'accès des transactions à la base imposée par les politiques de verrouillage, et à la charge supplémentaire induite par la gestion des verroux. Une approche d'analyse quantitative du rôle de ces facteurs est présentée dans cette étude. Dans une première étape, les facteurs les plus importants qui déterminent le comportement final du système du point de vue de ses performances sont mis en évidence et analysés indépendamment. Les résultats ainsi obtenus sont réunis dans une seconde étape pour conduire une évaluation des performances finaleq. Au cours de cette approche de modélisation hiérarchisée, diverses techniques de résolution analytique sont utilisées et les résultats obtenus sont illustrés par des exemples numériques. Les conclusions de l'article indiquent la sensibilité des résultats pbservés aux hypothèses concernant le comportement des transactions et la nécessité de données expérimentales dans ce domaine.

ANALYSIS OF LOCKING POLICIES IN DATABASE MANAGEMENT SYSTEMS

ABSTRACT

Consistency control has to be enforced in database management systems (DBMS) where several transactions may access concurrently the database. This control is usually achieved by dividing the database into locking units or granules, and by specifying a locking policy which ensures integrity of the information. However, a drawback of integrity enforcement through locking policies is the degradation of the global system performance. This is mainly due to the restriction imposed by the locking policies to the access of transactions to the database, and to the overheads involved with the management of locks. A framework for the quantitative analysis of the impact of these factors on the performance of DBMS is presented in this paper. In a first step, the main factors which determine the behavior of these systems are pointed out and analyzed independently. The results hereby obtained are aggregated in a second step to yield a global performance evaluation. Throughout this hierarchical modeling approach various analytical techniques are used and the result are illustrated by numerical examples. The paper concludes by pointing out final results sensitivity to some basic assumptions concerning transaction behavior and the need for more experimental studies in this area.

KEY WORDS AND PHRASES : consistency, concurrency, database management, performance modeling, queueing models, queueing networks.

C.R. CATEGORIES : 3.50, 4.33, 8.1

I - INTRODUCTION

Quantitative analysis of locking mechanisms and of their impact on the performance of transaction systems have received relatively little attention [RiS77, RiS79, BlG79]. Although numerous concurrency mechanisms have been proposed and implemented, there is an obvious lack of experimental as well as analytical studies of their behavior and their influence on system performance. The present state of the art in this area can be compared to the situation in the late sixties when the first virtual memory systems were being implemented : at this time the now classical concepts of program behaviour (locality of references, working-sets, memory management policy, etc...) were only emerging and it took several years to fully understand and master them.

In order to perform a quantitative analysis of transaction systems of the performance, it is essential to point out the factors that determine the behavior of these systems. Three main factors may be identified :

i) transaction behavior : it may be described by the pattern of references of the transactions to the subsets (logical or physical) of the database and by the way the Central Processing Unit (CPU) and Input/Output (IO) resources of the system are used by the transactions. Given a locking mechanism, the reference pattern is obviously a determining factor : depending on the way references are distributed over the database, the locking mechanisms will have a more or less drastic effect. However, as noted above, little experimental evidence on reference patterns of transactions is available. In particular, the existence of properties of sequentiality or locality in the access pattern is still a contoversial issue [RoH75, TuR76, Smi78, Rod76]. Faced with this situation, any analysis relies on rather strong assumptions. For instance, in the analysis of locking granularity presented in [Ris77], Ries and Stonebraker characterize the behavior of a transaction by the number of entities referenced, and assume that the number of granules locked by a transaction is proportionnal to the number of entities referenced.

In our analysis, we shall use a somewhat more sophisticated probabilistic model of transaction behavior although it also makes a strong assumption on the distribution of references over the database.

ii) locking scheme : locking schemes may by physical or logical depending on whether the units of locking are physical or logical subsets of the databases. We restrict our attention in this paper to physical locking. A physical locking mechanism is characterized by three main factors :

- the Lock Acquisition Policy (LAP) which specifies at which instant of the transaction lifetime locks are requested. LAP's are basically split into two categories: static LAP's (SLAP) and dynamic LAP's (DLAP). Under SLAP all the locks needed for executing transaction are requested at the initiation of the transaction. The transaction is executed only if all its requested locks are granted ; otherwise it is blocked. Under a DLAP, locks are requested by a transaction on demand, whenever a transaction references a granule that is not already locked.

- the Lock Release Policy (LRP) which specifies when the locks acquired by a transaction are released. In order to maintain the consistency of the database a static LRP (SLRP) has to be used, i.e. a LRP where a transaction releases its locks only when its execution has been completed.

Following [RiS77], the analysis presented in this paper is carried out by assuming that the locking policies used are SLAP and SLRP. The main advantages of this combination of LAP and LRP is that deadlocks are avoided and that, from the point of view of the analyst, its study is easier. However, it should be kept in mind that locks mean lifetimes are longer under a SLAP than under a DLAP, and hence that a SLAP restrict the transaction's accesses to the database more severely than a DLAP.

- the size of the physical locking units or granules (assuming that all locking units have the same size). Given a LAP and a LRP, the remaining

design parameter is the granule size. Obviously, in theory, the smaller the granules, the smaller the constraints imposed by the locking mechanism. However, at this point, secondary factors such as locking overheads have to be taken into account, since their influence may become important when the number of granules is large.

iii) the multiprogramming environment : the multiprogramming environment describes the resources (CPU, IO) that the transaction system uses for the actual processing of transactions. These resources, their characteristics and organization have an important impact on the global performance. Of the three factors we have identified, the latter is also presently the best understood and mastered. Within the past ten years much experimental an analytical effort has been devoted to this area [DeK76, MeP78], and the appropriate methods and tools for the analysis of the multiprogramming environment do exist.

Once these factors have been identified, we must select which approach will be used to analyze them and quantify their influence. In [RiS77, MuK77] simulation models are used. Although, this technique has the advantage of imposing no restriction on the complexity of the analyzed mechanisms, it cannot always provide a clear insight and understanding of the system performance and of the role of key assumptions.

Our approach in this paper is based on hierarchical analytical modeling [ChS78]. The basic idea underlying this approach is that in a first step each critical factor should be analyzed independently, clearly stating assumptions and their consequences ; the results hereby obtained are aggregated in a second step to yield a global performance evaluation and final conclusions.

A broad presentation of this approach is given in section II. Section III is devoted to the model of transaction behavior and the analysis of the locking mechanism. The multiprogramming environment is studied in

in section IV. Final results on the global performance are presented and discussed in section V. Conclusions are given in section VI.

II - PRESENTATION OF THE APPROACH

The transaction system is analyzed using a hierarchical modeling approach. Three levels of modeling are considered as represented in figure 1.

At level 1, the different stages transactions go through during their lifetime are described. The model used at this level is represented in figure 2. Transactions are issued from a set of interactive terminals. Upon arrival of a new transaction in the system all the locks needed for the complete execution of the transaction are requested. Lock requests are processed by the CPU and I/O resources of the system. During the "locks request" phase the transaction is in the state REQUEST. Upon completion of this phase, locks requested are either granted, either denied. If locks are granted, the transaction enters the state ACTIVE and it is executed by the CPU and I/O resources of the system until completion. It then returns to the terminal that has issued it. If locks are denied, the transaction is sent into the BLOCKED queue and enters the state BLOCKED.

BLOCKED transactions are removed from the BLOCKED queue in order to enter a new "locks request" phase whan an ACTIVE transaction leaves the system and releases its locks. The maximum number of BLOCKED transactions removed from the BLOCKED queue at each departure is a control parameter of the transactions scheduling policy.

The organization and operations of the CPU and I/O resources of the system are described and analyzed at level 2. These resources are used by the transactions when they are either requesting locks (state REQUEST), or being executed (state ACTIVE). The analysis of level 2 will be conducted to derive the processing rates of ACTIVE transactions and REQUEST transactions depending on the number of these transactions sharing the CPU and I/O resources.

The behavior of transactions during their locks request phase will be analyzed at modeling level 3. Given a simple model of the transaction references requests to the entities of the database and the number of granules in the database, we will derive the probability distribution of the number of blocked granules depending on the number of ACTIVE transactions. These results will be used to obtain the probability that locks are granted to a transaction at the end of its locks request phase depending on the number of ACTIVE transactions.

The three levels of modeling will be presented and analyzed in the following order : Level 3, Level 2 and Level 1.

III - ANALYSIS OF TRANSACTION BEHAVIOUR AND LOCKING MECHANISMS

III.1 - The Physical database

The physical database consists of cells that represent the elementary physical unit of storage (this will be for example a disk sector) and it is divided into m granules, where the granule is the locking unit.

III.2 - Transaction behavior

The behavior of a transaction consists of a sequence of accesses to the cells of the database. This sequence is defined by the following assumptions :

1 - a transaction makes n accesses during its execution,

2 - accesses are uniformly distributed over the cells of the database,

3 - accesses are independent.

Given these assumptions we shall derive the probability $P_k(\ell)$, $\ell = 1,\ldots,m$, that ℓ granules are locked when k transactions are simultaneously active, and the probability q_k that a new transaction enters the set of active transactions when k transactions are already active. These results will then be illustrated by numerical examples.

III.3 - Derivation of $P_k(\ell)$ and q_k

Let $x(p)$, $p = 1,\ldots,m$ be the probability that the number of distinct granules accessed by a transaction is p, given that the transaction makes n accesses. The derivation of $x(p)$ is similar to a standard combinatoric problem : given n like objects and m unlike buckets, what is the probability of leaving p buckets non-empty after having uniformly distributed the objects into the m buckets ? Applying standard results of combinatorics (see for instance [Rio58] pp. 103), we then have

$$(1) \qquad x(p) = \binom{m}{p} \binom{n-1}{p-1} / \binom{m+n-1}{n}$$

and, obviously

$$(2) \qquad P_1(\ell) = x(\ell) \quad , \quad \ell = 1,\ldots,m$$

In order to derive $P_k(\ell)$, we write the following recurrence formula between $P_k(\ell)$ and $P_{k-1}(\ell)$,

$$(3) \qquad P_k(\ell) = \frac{1}{\sum_{j=1}^{m-1} P_{k-1}(j)} \sum_{i=1}^{m-1} P_{k-1}(i) \cdot \frac{X_i(\ell-i)}{\sum_{j=1}^{m-i} X_i(j)}$$

where

$$(4) \qquad X_i(p) = x(p) \binom{m-i}{p} / \binom{m}{p} = \binom{m-i}{p} \binom{n-1}{p-1} / \binom{m+n-1}{n}$$

In (4), $X_i(p)$ is defined as the probability that a transaction is granted p granules given that i granules are already locked. The terms in the denominator of (3) are normalisation factors : since the k-th transaction can become active only if there are less than m granules locked by the (k-1)st transactions, the probabilities $p_{k-1}(i)$, i = 1,...,m-1 are normalized by the sum $\sum_{j=1}^{m-1} p_{k-1}(j)$; in the same fashion, given i, the maximum number of new granules locked by the k-th transaction is m-i, so that the probabilities $X_i(n)$ in (3) are weighted by $\sum_{j=1}^{m-i} X_i(j)$.

From $P_k(\ell)$, we can derive the mean number $\bar{\ell}_k$ of granules locked when k transactions are active

$$(5) \qquad \bar{\ell}_k = \sum_{\ell=1}^{m} i \; . \; P_k(i)$$

and we straightforwardly obtain the probability q_k that a new transaction is granted the locks requested when k transactions are already active :

$$(6) \qquad q_k = \sum_{i=1}^{m-1} P_k(i) \sum_{p=1}^{m-i} X_i(p)$$

III.4 - Discussions and numerical examples

The results obtained in the previous section may help us to have a first insight of the effect of locking on the operation of the transaction system. The mean number $\bar{\ell}_k$ of locked granules and the probability q_k give clear indications on the maximum possible number of simultaneously active transactions, and on the chance that a new transaction has its locks granted.

These indications will have to be analyzed and interpreted with respect to the model of transaction behavior that has been used. Recall that the model assumes that, for a given transaction, its accesses to the

cells of the database are uniformely distributed over the whole set of cells : in other words we assume that transaction references exhibit no locality. We may thus expect with this model of transactions behavior the effect of locking to be severe.

Those observations are illustrated by the numerical results presented in Tables I and II. These results have been obtained for $n = 2^i$, $i = 1, 5$, $m = 2^j$, $j = 1, 8$ and k ranging from 1 to 8. Table I and Table II respectively give $\bar{\ell}_k$ and q_k. Figures 3a and 3b depicts the variation of $\bar{\ell}_k$ as a function of m with $n = 4$ and $n = 16$.

The importance of the no-locality assumption is revealed by the influence of the number n of accesses made by a transaction on the mean number of blocked granules $\bar{\ell}_k$. Transactions with a large number of accesses to the database very rapidly tend to lock most of the granules and to prevent activation of new transactions. On the other hand, this effect is partly allievated by an increase of the number of granules. However, as we shall see in the following sections, the increase of the number of granules causes an increase of the CPU and I/O activity devoted to locks requests, which is detrimental to the processing rate of active transactions.

IV - ORGANIZATION AND OPERATIONS OF THE CPU AND I/O RESOURCES

At a given instant, the transactions in states REQUEST or ACTIVE share the CPU and I/O resources of the system. We shall assume that there is no priority scheme set between ACTIVE and REQUEST transactions, and that the CPU processes these transactions on a processing-sharing (PS) fashion and I/O units process them according to a FIFO policy. The system consists of one Central Processing Unit (CPU) and a set of B identical independent Disk Units (DU). The I/O operations of a transaction are uniformly distributed over the B DU's.

We shall now describe the different CPU and I/O operations performed by REQUEST and ACTIVE transactions.

IV.1 - Operations performed by REQUEST transaction

Let v(m) be the mean number of distinct granules requested by a transaction. A transaction in the state REQUEST will thus perform on the average v(m) elementary lock request operations. An elementary lock request operation consists of :

- a CPU service time with mean t_R
- a DU service time with mean s_R.

An expression of v(m) is simply obtained from the results presented in the previous section. We have

$$v(m) = \sum_{p=1}^{m} p.x(p)$$

and, from (1),

$$(7) \qquad v(m) = \frac{mn}{m+n-1} .$$

It should be noted here that we have assumed that the elementary operations involved in a lock request, i.e. the CPU and DU services do not depend on m. This is certainly an optimistic assumption since management of locks tables may become more cosly as m increases, and, for instance, several disk accesses may be needed per lock request for a large value of m. Therefore, results obtained for fixed values of t_R and s_R will have to be interpreted with respect to this assumption.

IV.2 - OPERATIONS PERFORMED BY ACTIVE TRANSACTIONS

A transaction in the ACTIVE state performs n accesses to the entities of the database, each access being followed by a CPU processing time. The activity of an ACTIVE transaction will thus consists of n elementary operations comprising :

- a CPU service time with mean t_A
- a DU service time with mean s_A.

We assume that the CPU and DU service times are independent random variables, and that DU service times have a negative exponential distribution.

IV.3 - Processing rates of ACTIVE and REQUEST transactions

Let k and p be respectively the number of ACTIVE transactions and REQUEST transactions processed by the resources at a given instant of time. Due to the fact that ACTIVE and REQUEST transactions have different behaviors their processing rates depend on k and p through functions of the form $\mu_A(k, p)$ and $\mu_R(k, p)$.

In order to derive $\mu_A(k, p)$ and $\mu_R(k, p)$, we follow the standard approach used for the throughput analysis of multiprogramming systems [DeK76]. As represented in figure 4, fixed numbers k and p of ACTIVE and REQUEST transactions are maintained in the system, and the sharing of the CPU and DU resources by the two types of transactions is analyzed using a closed queueing network model with different classes of customers [BaC75].

An analysis of this model will yield for instance the throughput rate $X_A(k, p)$ and $X_R(k, p)$ of ACTIVE and REQUEST transactions through station CPU. Since ACTIVE and REQUEST transactions require on the average respectively n and v(m) CPU service times, we have

$$\mu_A(k, p) = X_A(k, p)/n \ ,$$
$$\mu_R(k, p) = X_R(k, p)/v(m).$$

The functions $\mu_A(k, p)$ and $\mu_R(k, p)$ aggregate the different effects of resource sharing by ACTIVE and REQUEST transactions. Using the equivalent server approach [ChS78], they will be used at modeling level 1 to analyse the global performance of the transaction system.

It should be noted that $\mu_A(k, p)$ and $\mu_R(k, p)$ depend on the granularity m through the function v(m). Increasing the number of granules will increase the number of elementary lock requests performed by a transaction in the state REQUEST, and limit the availability of the CPU and DU resources to ACTIVE transaction. As a consequence, the processing rate of ACTIVE transactions will be degraded.

In order to illustrate this effect, we have plotted in figure 5 $\mu_R(k, p)$ as a function of p for m ranging from 2 to 2^8 with n = 4 and k = 2. The results have been derived with the following values of the parameters :

t_A = 50 msec.

t_R = 10 msec.

$s_A = s_R$ = 50 msec.

B = 3 (3 identical DU's)

They have been obtained by using the solution package QNAP [MeP78, Ver79] developped by CII-Honeywell Bull, and INRIA. The listing of the QNAP program used for this computation and example of its outputs are given in Table III. Note that the value of $1/\mu_A(k,p)$ and $1/\mu_R(k,p)$ are stored in an array after each resolution so as to be used later on when global performance is analyzed.

It can be observed from figure 5 that, due to the specific behavior of the function v(m), the effect of granularity on the locking overheads is particularly important as m increases from 2 to 2^5.

V - ANALYSIS OF GLOBAL PERFORMANCE

Having analyzed modeling levels 2 and 3, we now dispose of the aggregate results that are needed for the analysis of the global performance. The interactions between the different factors which have already been investigated are summarized in Figure 6. Analysis will be conducted at modeling level 1 by using a multiclass queueing network model as presented in figure 7. The set of K terminals is modeled by an infinite servers station TERMINAL with mean thinking time T, and the set of CPU-I/O resources is replaced by two exponential FIFO stations RESREQ and RESACT, with processing dependent rates $\mu_A(k, p)$ ($\mu_R(k, p)$) when k ACTIVE transactions and p REQUEST transactions are respectively present in these stations.

When a REQUEST transaction has completed its lock requests, its locks are granted with probability q_k, if k transactions are active, and it then enters the set of ACTIVE transactions ; with probability $1-q_k$, its locks are denied and the transaction proceeds to the BLOCKED queue.

Transactions are removed from the BLOCKED queue when ACTIVE transactions terminate and release their locks. Thus removal of transactions from the BLOCKED queue is trigerred by departures from the station RESACT. Let r be the maximum number of transactions removed from queue BLOCKED. If i if the current number of transactions in queue BLOCKED, the actual number of transactions removed from queue BLOCKED at each departure is min(i, r).

Due to the strong dependence between queues BLOCKED, RESACT and RESREQ, standard results on multiclass queueing network models are not longer applicable. Given the assumptions of the model, the only exact solution technique available is through a Markovian analysis [Ste78]. This is done by using again the solution package QNAP. The text of the QNAP program is given in Table IV. Transfers from the BLOCKED queue to the RESREQ queue are specified by the primitive MOVE. Since QNAP uses only linear arrays a macro-instruction IND(I) is used to access the element of the arrays TA and TR where $1/\mu_A(k,p)$ and and $1/\mu_R(k, p)$ are stored.

The results are presented in figure 8 and 9 . They have been obtained with the following values of the parameters of the model :

$n = 4,$

$m = 2^0$ to $2^8,$

$t_A = 50$ msec.,

$t_R = 10$ msec.,

$s_A = s_R = 50$ msec.,

$B = 3,$

$r = 1,$

$T = 1.5$ sec.,

Figure 8 displays the mean number of ACTIVE, REQUEST and BLOCKED transactions in the system as a function of m. The main observation is that as m increases the mean number of BLOCKED transactions decreases rapidly, while the mean number of ACTIVE transactions steadily increases.

The mean number of REQUEST transactions reaches a maximum for m=32 and slowly decreases afterwards. It is thus to be expected thay the global throughput will increase with m for large values of m.

This is illustrated in figure 9 where is plotted the throughput D(m) of the system, for n ranging from 2 to 2^5, expressed in terms of mean number of transactions processed per unit of time. It is observed thay D(m) decreases through a minimum and then increases with m.

This observation, which contradicts the results presented in [RiS77] where D(m) exhibited a maximum, is to interpreted with respect to the set of assumptions that have been used through the analysis. As it appears from the results of modeling levels 2 and 3, since q_k increases with m, the ratio of the number of ACTIVE transactions to the number of REQUEST transactions tends to increase ; on the other hand, since the mean number of locks v(m) tested by REQUEST transaction increases with m, REQUEST transactions tend to need more processing as m increases. However, as indicated by the behavior of the throughput function D(m), this behavior is determined by the increase of q_k rather than by the increase of v(m). This points out again the key role played by locking overheads, here quantified by the function v(m). In order to explain the discrepancies observed between our results and those presented in [RiS77], it should be recalled that the function v(m) used in this study is of the form $v(m) = \alpha m$. In general, assumptions on transaction behavior used in those two studies are basically different : Ries and Stonebraker use a "linear" model of transaction behavior ; we use a "non-linear" model.

VI - CONCLUSIONS

We have presented in this paper an analytical framework for the performance analysis of locking mechanisms in transaction systems, and we have illustrated this approach by a detailed analysis based on a simple probabilistic model of transaction behavior.

This analysis provides a clear understanding of the various factors that determine global performance. It also raises many new issues that can only be solved by further extensive experimental and analytical studies. Two particular topics deserve special attention : the modeling of transaction behavior and the modeling of locking overheads. As noted above, the model of transaction behavior we have used makes a strong assumption on the distribution of references over the database : this model has to be validated by comparing its results to those of more complex models, involving for instance localities or sequentialities in the reference pattern. We have also pointed out in Section V the key role played by locking overheads. Here again a more refined analysis is needed, in relation with the modeling of transaction behavior. We are currently working along those directions and we hope that this paper will initiate parallel investigations.

ACKNOWLEDGEMENTS

The authors would like to thank J.L. COLOMER of the Facultad de Informatica, Universidad de Barcelona, SPAIN ; R. BALTER of the Centre Scientifique CII-Honeyqell Bull in Grenoble, FRANCE ; and M. SCHOLL of INRIA for their helpfull contributions.

REFERENCES

[BaC75] F. BASKETT, K.M. CHANDY, R.R. MUNTZ, J. PALACIOS

"Open, closed and mixed networks with different classes of customers" - J.ACM 22, 2, April 1975, pp. 248-260.

[BlG79] M. BLASGEN, J. GRAY, M. MITOMA, T. PRICE

"The convoy phenomena" - ACM Operating Systems Review, 3, 2, April 1979, pp. 20-25.

[ChS78] K.M. CHANDY, C.H. SAUER

"Approximate methods for analyzing queueing network models of computing systems" - ACM Comp. Surveys, 10, 3, Sept. 1978, pp. 281-317.

[DeK76] P.J. DENNING, K.C. KAHN, J. LEROUDIER, D. POTIER, R. SURI

"Optimal multiprogramming" - Acta Inf. 7, 2, 1976, pp. 197-216.

[Gil78] D.C. GILBERT

"Modelling spin locks with queueing networks" - ACM Operating Systems Review, 2, 1, Jan. 78, pp. 29-42.

[MeP78] D. MERLE, D. POTIER, M. VERAN

"A tool for computer system performance analysis" - Proc. Int. Conf. on Performance of Computer Installations 1978, North-Holland Publishing Company, pp. 195-214.

[MuK77] R.R. MUNTZ, G.K. KRENZ

"Concurrency in database systems - simulation study" - Proc. ACM SIGMOD Int. Conf. on Manag. of Data, Toronto, Ont., Canada, Aug. 1977.

[Rio58] J. RIORDAN

"An introduction to combinatorial analysis" - John Wiley and Sons, Inc., New York 1958.

[RiS77] D.R. RIES, M.R. STONEBRAKER

"Effects of locking granularity in a database management system" - ACM-TODS, 2, 3, Sept. 1977, pp. 233-246.

[RiS79] D.R. RIES, M.R. STONEBRAKER

"Locking granule revisited" - ACM-TODS, 4, 2, June 1979, pp. 210-227.

[Rod76] J. RODRIGUEZ-ROSELL

"Empirical data reference behaviour in database systems" - Computer 9, 11, Nov. 1976, pp. 9-13.

[RoH75] J. RODRIGUEZ-ROSELL, D. HILDEBRAND

"A framework for evaluation of database management system" - Proc. Intern. Computing Symposium 1975, E. Gelenbe, D. Potier Eds., North-Holland Publishing Company, 1975.

[Smi78] A.J. SMITH

"Sequentiality and prefetching in database systems" - ACM-TODS, 3, 3, Sept. 1978, pp. 223-247.

[Ste78] W.J. STEWART
"A comparison of numerical techniques in Markov modelling" - Com. ACM, 21, 2, Feb. 78, pp. 144-151.

[TuR76] W.G. TUEL, J. RODRIGUEZ-ROSELL

"A methodology for evaluation of database systems" - Research Report RJ 1668, IBM Research Laboratory, Yorktown Heights, New York, 1976.

[Ver79] M. VERAN

"QNAP description language" - Research Report, Antenne Scientifique CII-Honeywell-Bull, Grenoble, 1979.

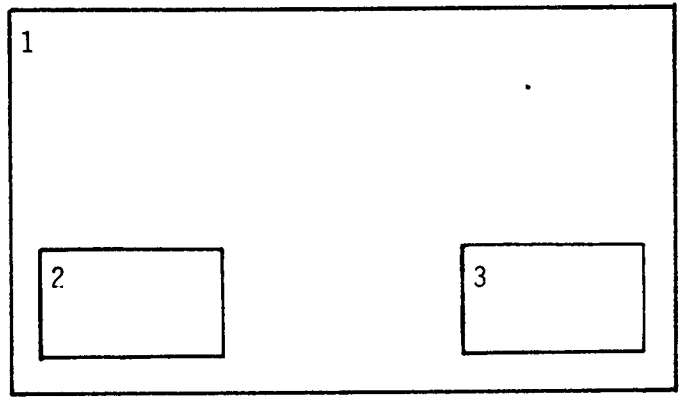


Figure 1

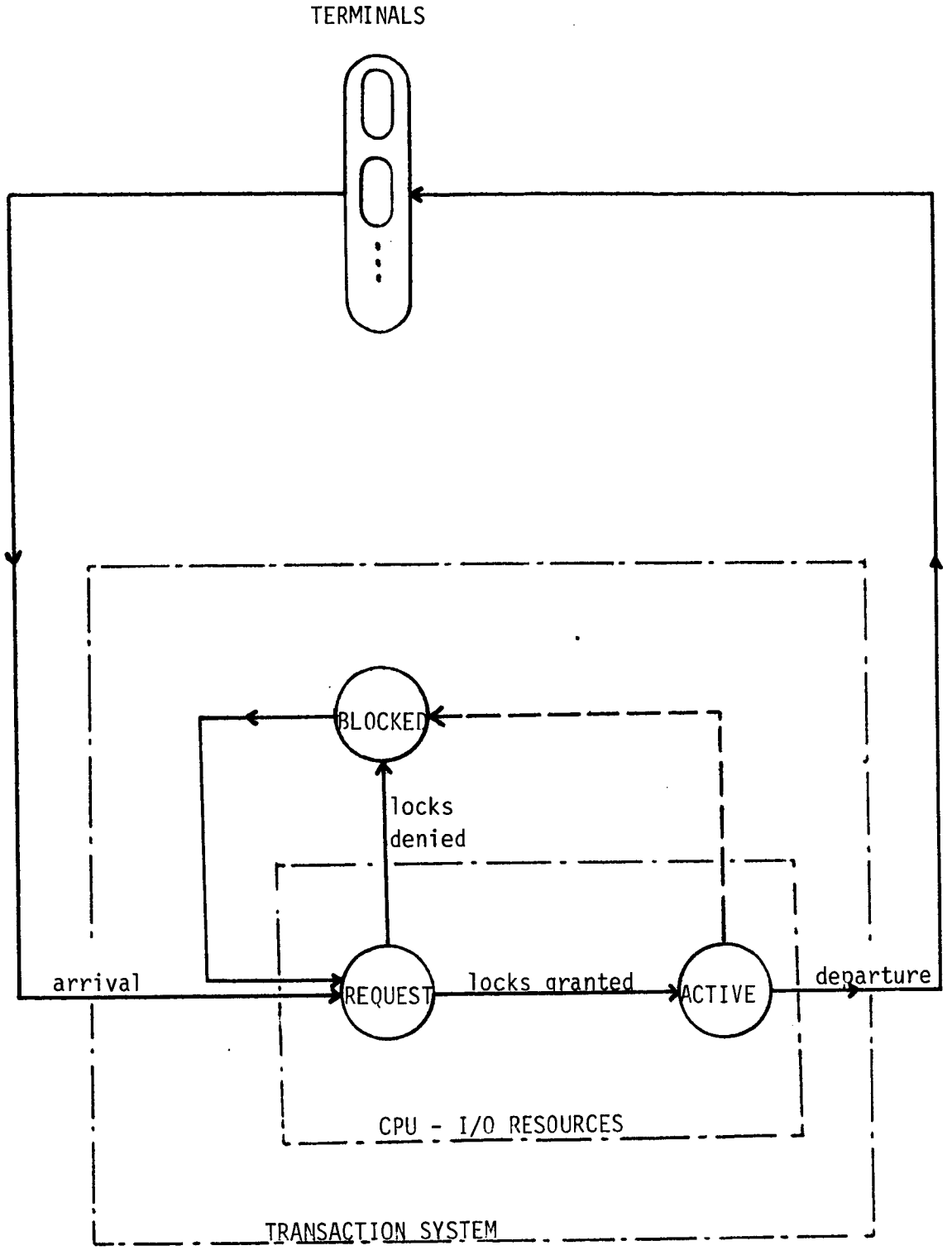

TERMINALS
BLOCKED
locks
denied
arrival
REQUEST
locks granted
ACTIVE
departure
CPU - I/O RESOURCES
TRANSACTION SYSTEM

Figure 2

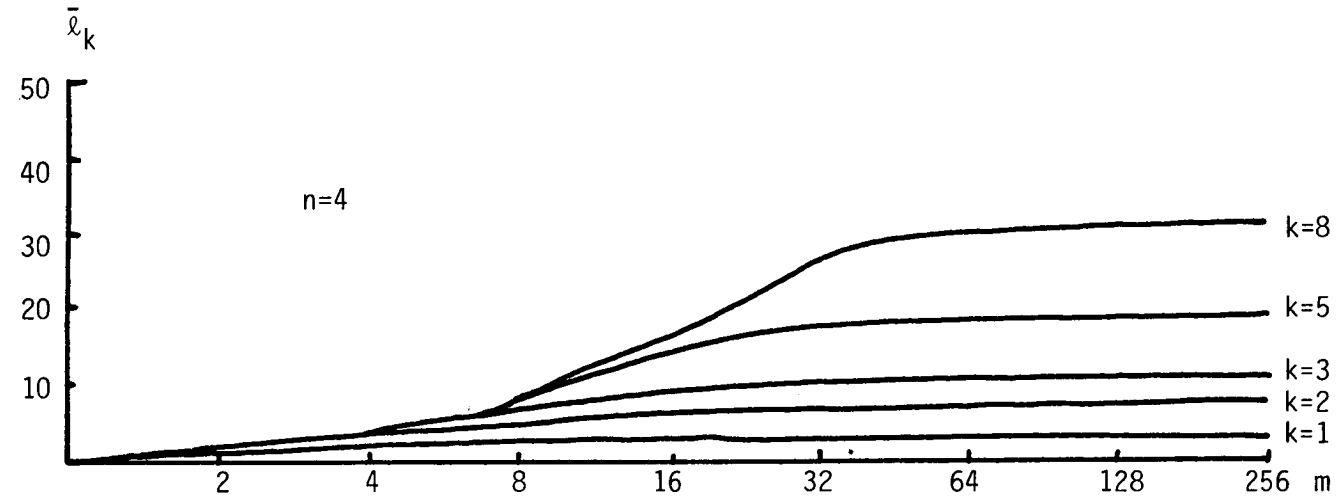


Figure 3a : $\bar{\ell}_k$ function of m, n=4

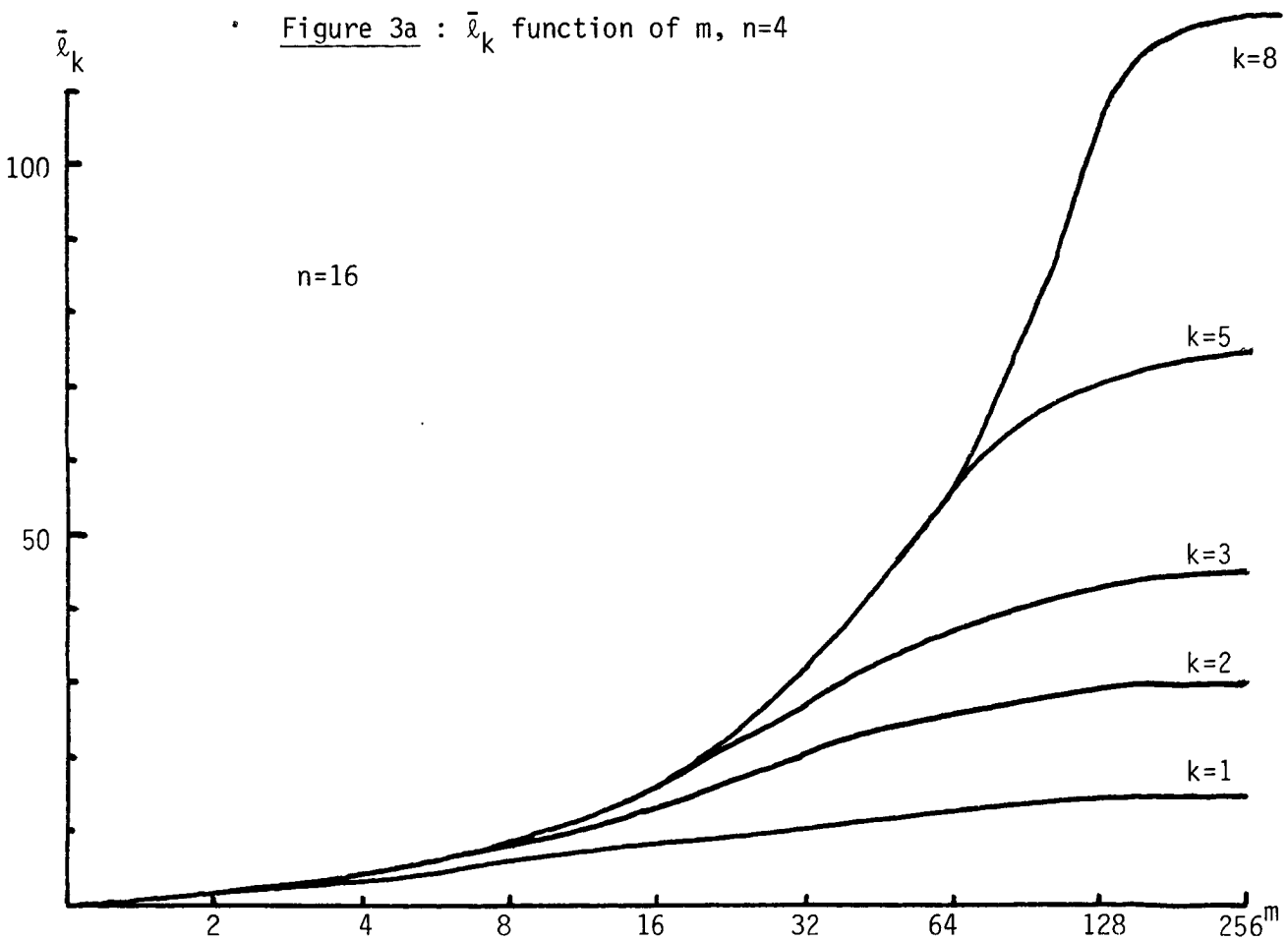


Figure 3b : $\bar{\ell}_k$ function of m, n=16

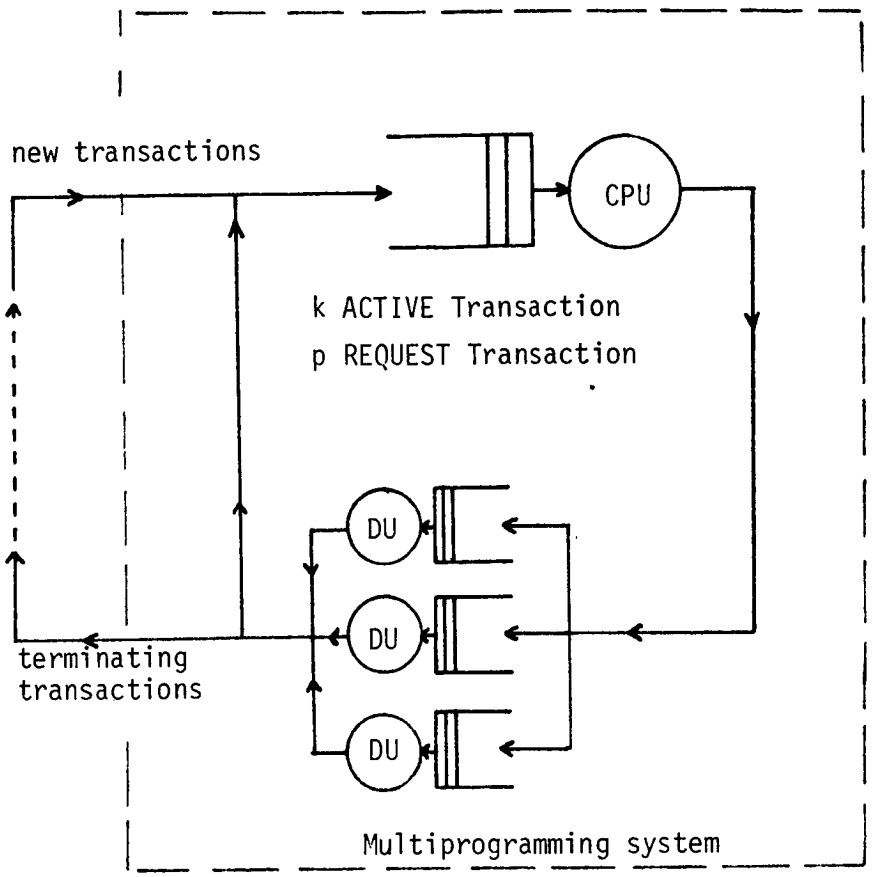

new transactions
CPU
k ACTIVE Transaction
p REQUEST Transaction
DU
DU
DU
terminating
transactions
Multiprogramming system

Figure 4

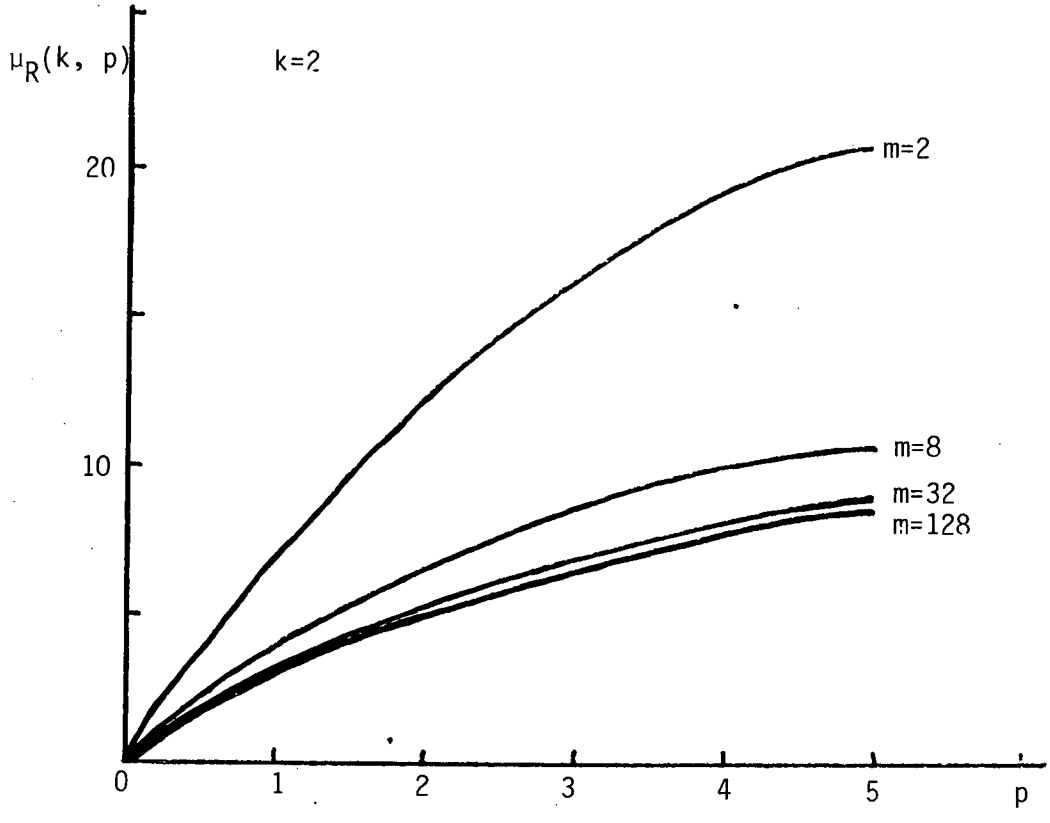


Figure 5 : $\mu_R(k, p)$ function of p

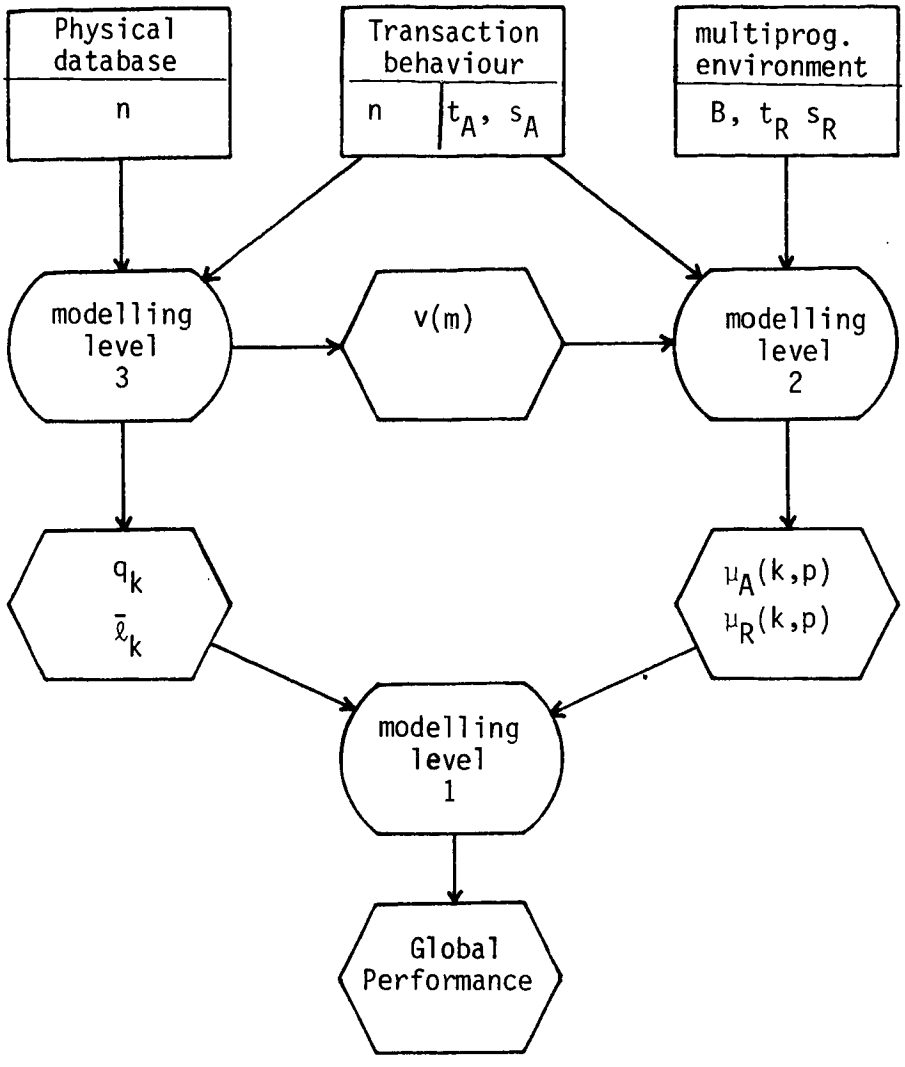
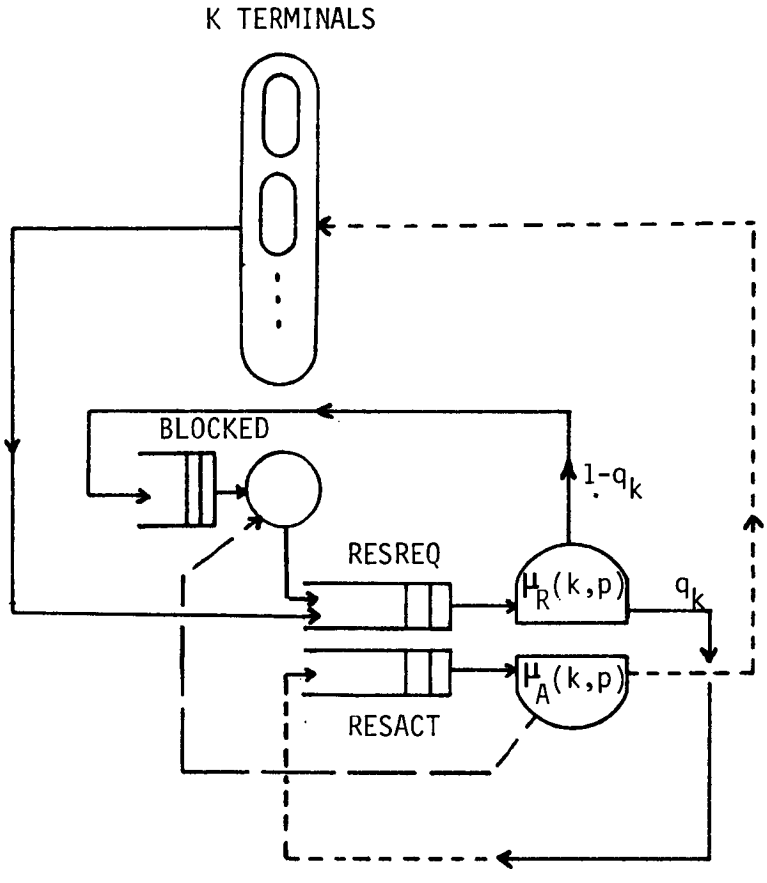

Physical database
n
Transaction behaviour
n
t_A, s_A
multiprog. environment
B, t_R s_R
modelling level 3
v(m)
modelling level 2
q_k
$\bar{\ell}_k$
$\mu_A(k,p)$
$\mu_R(k,p)$
modelling level 1
Global Performance

Figure 6



——→ REQUEST transactions

- - - - - - → ACTIVE transactions

——— → Removal of transaction

Figure 7

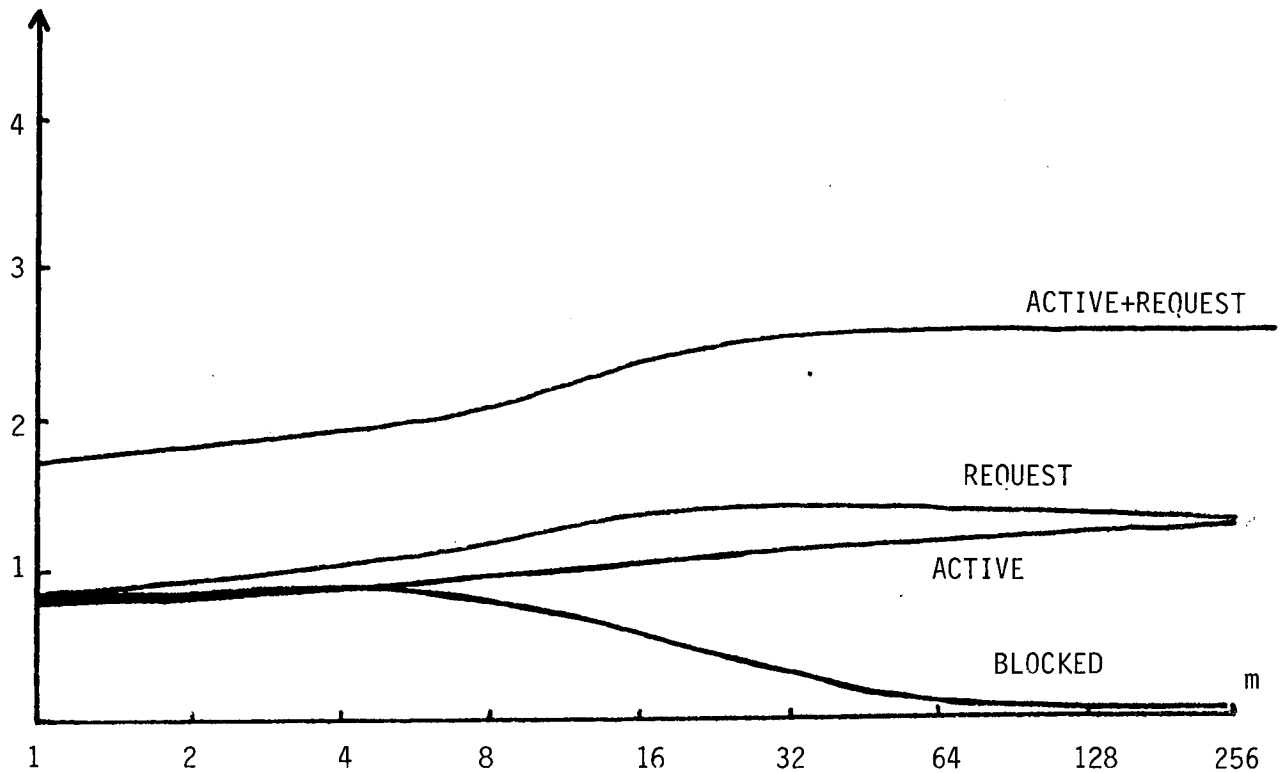


Figure 8

Mean number of transactions in queues

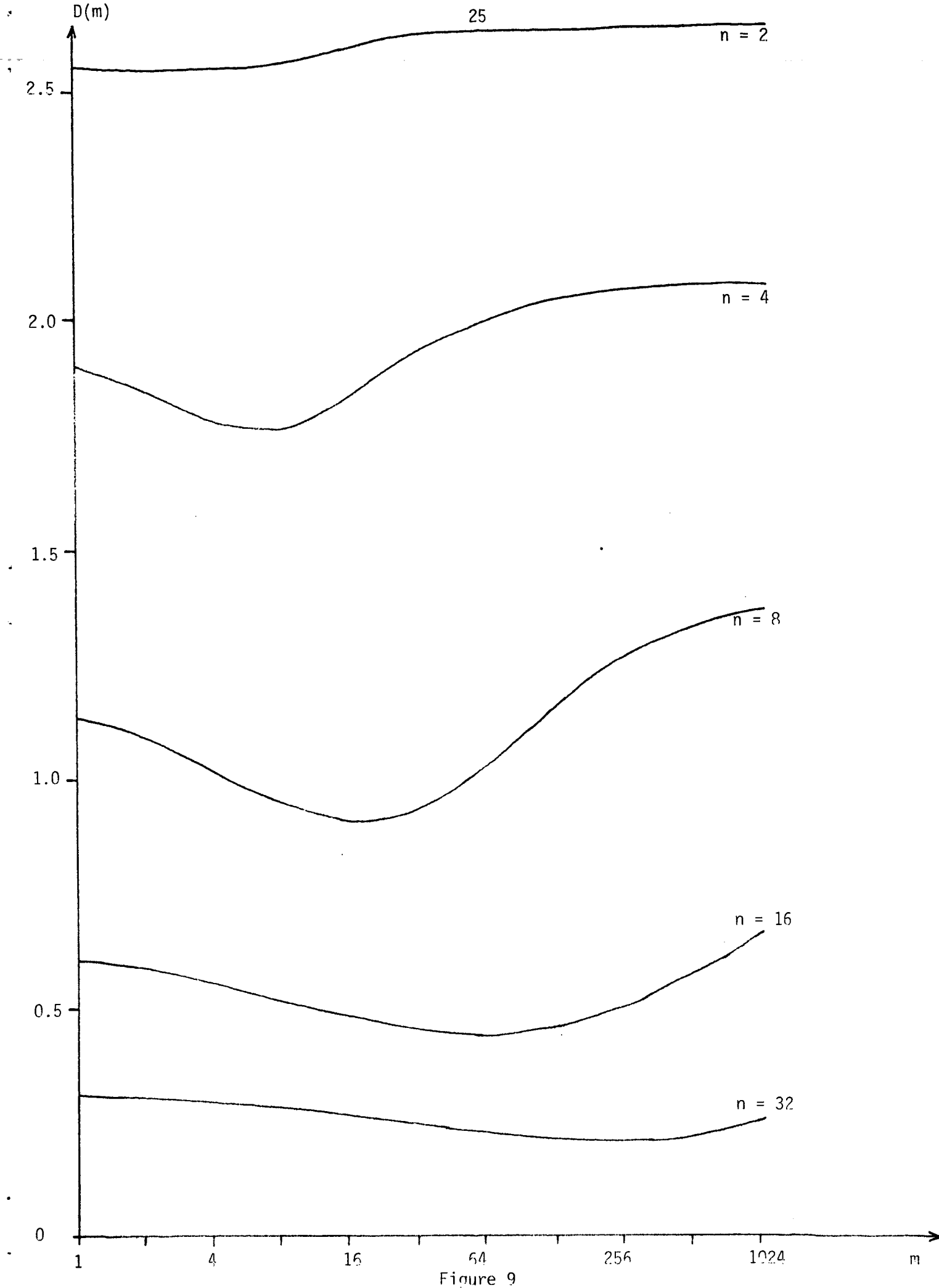


Figure 9

n	m	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
2	2	.333	.133	.095	.065	.049	.038	.030	.024
3	3	.303	.134	.079	.053	.040	.030	.024	.019
4	4	.275	.130	.077	.052	.038	.029	.023	.018
5	5	.257	.123	.073	.050	.037	.028	.022	.018
6	6	.237	.117	.070	.048	.035	.027	.022	.017
7	7	.222	.112	.067	.046	.034	.026	.021	.017
8	8	.210	.107	.064	.044	.033	.026	.021	.016
3	4	.429	.171	.097	.063	.045	.034	.027	.022
4	4	.352	.141	.081	.054	.039	.030	.024	.019
5	4	.324	.131	.076	.051	.037	.029	.023	.019
6	4	.307	.125	.073	.049	.036	.028	.023	.018
7	4	.294	.121	.071	.048	.035	.028	.022	.018
8	4	.284	.117	.069	.047	.035	.027	.022	.018
4	6	.429	.171	.097	.063	.045	.034	.027	.022
5	6	.352	.141	.081	.054	.039	.030	.024	.019
6	6	.324	.131	.076	.051	.037	.029	.023	.019
7	6	.307	.125	.073	.049	.036	.028	.023	.018
8	6	.294	.121	.071	.048	.035	.028	.022	.018
5	8	.429	.171	.097	.063	.045	.034	.027	.022
6	8	.352	.141	.081	.054	.039	.030	.024	.019
7	8	.324	.131	.076	.051	.037	.029	.023	.019
8	8	.307	.125	.073	.049	.036	.028	.023	.018
6	10	.429	.171	.097	.063	.045	.034	.027	.022
7	10	.352	.141	.081	.054	.039	.030	.024	.019
8	10	.324	.131	.076	.051	.037	.029	.023	.019
7	12	.429	.171	.097	.063	.045	.034	.027	.022
8	12	.352	.141	.081	.054	.039	.030	.024	.019
8	14	.429	.171	.097	.063	.045	.034	.027	.022

Table I : q_k

n	m	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
2	2	1.333	2.000****	******	******	******	******	******	******
4	2	1.600	2.000****	******	******	******	******	******	******
8	2	1.778	2.000****	******	******	******	******	******	******
16	2	1.882	2.000****	******	******	******	******	******	******
32	2	1.939	2.000****	******	******	******	******	******	******
2	4	1.600	[illegible]	3.833	4.000****	******	******	******	******
4	4	2.286	3.671	3.969	4.000****	******	******	******	******
8	4	2.909	3.918	3.995	4.000****	******	******	******	******
16	4	3.368	3.977	3.999	4.000****	******	******	******	******
32	4	3.657	3.994	4.000	4.000****	******	******	******	******
2	8	1.778	3.5[illegible]	5.132	6.596	7.597	7.899	7.977	[illegible]
4	8	2.909	5.413	7.194	7.850	7.969	7.993	7.999	8.000
8	8	4.267	7.037	7.985	7.986	7.998	8.000	8.000	8.000
16	8	5.565	7.740	7.995	7.999	8.000	8.000	8.000	8.000
32	8	6.564	7.943	7.999	8.000	8.000	8.000	8.000	8.000
2	16	1.882	3.750	5.500	7.427	9.213	10.954	12.616	[illegible]
4	16	3.368	6.599	9.626	12.327	14.462	15.586	15.92[illegible]	15.96[illegible]
8	16	5.565	10.341	13.864	15.509	15.939	15.998	15.997	16.000
16	16	8.258	13.664	15.727	15.979	15.999	16.000	[illegible]	[illegible]
32	16	11.894	15.37[illegible]	15.976	15.999	16.000	16.000	[illegible]	[illegible]
2	32	1.939	3.875	5.816	7.773	9.654	11.508	13.374	15.277
4	32	3.657	7.276	10.841	14.344	17.761	21.061	24.182	27.26[illegible]
8	32	6.564	12.826	18.689	23.914	28.168	30.852	31.707	31.957
16	32	10.394	23.235	27.228	30.987	31.9[illegible]	31.999	31.999	[illegible]
32	32	16.254	26.994	31.768	31.971	31.998	[illegible]	[illegible]	[illegible]
2	64	1.969	3.937	5.905	7.871	9.835	11.799	13.76[illegible]	[illegible]
4	64	3.821	7.671	11.428	15.217	18.991	22.77[illegible]	[illegible]	[illegible]
8	64	7.211	14.337	21.345	28.216	34.938	41.339	[illegible]	[illegible]
16	64	12.962	25.326	36.848	47.157	55.556	61.[illegible]	[illegible]	[illegible]
32	64	21.559	40.879	53.966	61.685	63.834	63.99[illegible]	63.999	[illegible]
2	128	1.984	3.969	5.953	7.936	[illegible]	[illegible]	[illegible]	[illegible]
4	128	3.909	7.814	11.717	15.616	19.51[illegible]	23.404	[illegible]	[illegible]
8	128	7.585	15.146	22.670	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
16	128	14.322	28.556	42.761	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
32	128	25.761	[illegible]	73.10[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2	256	1.992	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4	256	3.954	7.907	[illegible]	[illegible]	[illegible]	23.711	[illegible]	[illegible]
8	256	7.797	15.568	23.341	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
16	256	15.114	30.176	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
32	256	28.544	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Table II : $\bar{\ell}_k$

```
/DECLAR/QUEUE CPU,DU1,DU2,DU3;
      CLASS ACTIVE,REQUEST;
      REAL XA,XR,Y,NA,NR,TA(100),TR(100),M,N;
       INTEGER KTR,IA,IR,I;
&
/STATION/NAME=CPU;
      SCHED=PS;
      SERVICE(ACTIVE)=EXP(XA);
      SERVICE(REQUEST)=EXP(XR);
      TRANS=DU1,1,DU2,1,DU3,1;
      INIT(ACTIVE)=IA;
      INIT(REQUEST)=IR;
&
/STATION/NAME=DU1;
        SERVICE=EXP(Y);
       TRANS=CPU;
&
/STATION/NAME=DU2;COPY=DU1;
&
/STATION/NAME=DU3;COPY=DU1;
&
&
/PARAM/CLASS=CPU;
/EXEC/BEGIN
      XA:=0.05;
       XR:=0.01;
      N:=4;
      M:=1;
&
      NA:=4;
      NR:=M*N/(M+N-1);
&
      Y:=0.05;
      KTR:=5;
&
      IA:=0;
      WHILE IA<=KTR DO BEGIN
             IR:=0;
              WHILE IR<=KTR DO BEGIN
              ANALYTIC;
              I:=(KTR+1)*IA+IR+1;
              TA(I):=NA/MTHRUPUT(CPU,ACTIVE);
              TR(I):=NR/MTHRUPUT(CPU,REQUEST);
              PRINT(I,IA,IR,TA(I),TR(I));
             IR:=IR+1;
                      END;
                 IA:=IA+1;
                 END;
          END;
```

```
          **** ANALYTIC RESOLUTION ****
****SUBCHAIN   1 TOTAL NUMBER OF CUSTOMERS=   3
****SUBCHAIN   2 TOTAL NUMBER OF CUSTOMERS=   5
*************************************************************************
* NAME   *  SERVICE * BUSY PCT *  CUST NB * RESPONSE *  THRUPUT *
*************************************************************************
*        *          *          *          *          *          *
*  CPU   * .2027E-01* .8535    * 2.637    * .6262E-01* 42.12    *
* (ACTI) * .5000E-01* .5404    * 1.528    * .1413    * 10.81    *
* (REQU) * .1000E-01* .3131    * 1.110    * .3545E-01* 31.31    *
*        *          *          *          *          *          *
*  DU1   * .5000E-01* .7020    * 1.788    * .1273    * 14.04    *
*        *          *          *          *          *          *
*  DU2   * .5000E-01* .7020    * 1.788    * .1273    * 14.04    *
*        *          *          *          *          *          *
*  DU3   * .5000E-01* .7020    * 1.788    * .1273    * 14.04    *
*        *          *          *          *          *          *
*************************************************************************
SPACE USED :   3/100
          **** END OF ANALYTIC RESOLUTION ****
```

Table III

```
/DECLAR/QUEUE TERMINAL,RESREQ,RESACT,BLOCK;INTEGER
        K1,K2;REAL TREF,TREQUEST,TACTIVE,QQ;
        REAL Q(10)=1.0,0.94,0.883,0.828,0.776,0.727,0.68,0.635
                        ,0.592,0.332;
$
$ MACRO IND(I)
        IA:=CUSTNB(RESACT);
        IR:=CUSTNB(RESREQ);
        I:=(KTR+1)*IA+IR+1;
$ END
$
/STATION/NAME=TERMINAL;
         TYPE=DELAY;
         SERVICE=EXP(TREF);
         TRANS=RESREQ(REQUEST);
         INIT(ACTIVE)=KTR;
/STATION/NAME=RESREQ;
      SERVICE=BEGIN
                 I:=CUSTNB(RESACT);
                 QQ:=Q(I+1);
                 $IND(I)
                 EXP(TR(I));
               END;
      TRANSIT=RESACT(ACTIVE),QQ,BLOCK;
/STATION/NAME=RESACT;
      SERVICE=BEGIN
                 K1:=1;
                 $IND(I)
                 EXP(TA(I));
               END;
      TRANSIT=TERMINAL;
/PARAM/OPTION=RESULT;TEST=BEGIN
      IF K1=1 THEN  BEGIN K1:=0;MOVE(BLOCK,RESREQ);END;
       END;
/EXEC/BEGIN
       NETWORK(TERMINAL,RESREQ,RESACT,BLOCK);
      K1:=0;
      TREF:=1.5;
      MARKOV;
      PRINT(MCUSTNB(BLOCK),MTHRUPUT(BLOCK),MRESPONSE(BLOCK));
      END;
```

Table IV