No.443

An Analytic Model for Locking Preformance in Database

by

Shoichi Nishimura & Yong Jiang

October 1990

An Analytic Model for Locking Performance in Database

Shoichi Nishimura [1]
Yong Jiang [1]
Institute of Socio-Economic Planning
University of Tsukuba

August 1990

ABSTRACT: We propose an analytic model to predict the mean value performances of database concurrency control. The predictions are made in two steps. In the first step we give preliminary estimations of the mean value performances. Using the results of the preliminary estimations, we predict the system performances based on the Closed BCMP Networks in the second step. The model is characterized as a one-dimensional aggregate system, from which the approximate mean values of throughput and restart rate are estimated. We also obtain some important equilibrium distributions of the system as byproducts. The model is validated through a variety of cases, such as uniform and non-uniform access, write and read lock, fixed and indeterminate length of transactions. The comparisons show that the predictions of the model fit well with the simulation results.

[1] Authors' address : Institute of Socio-Economic Planning, University of Tsukuba, Tsukuba, Ibaraki 305, JAPAN

1. Introduction

A database consists of a finite set of data entities which may be shared by many users who update the database with transactions. A transaction is a sequence of requests of read and/or write of data entities resulting from the execution of users' programs. If several transactions concurrently access a database, synchronization anomalies may arise. The problem of avoiding synchronization anomalies is called the concurrency control problem for database systems [4].

Many concurrency control methods have been proposed. The most popular one is locking. There is one lock per data entity in its basic form. A transaction which wants to access a data entity must first obtain a lock for the data entity. The transaction locks on the data entity until the end of operation. Two kinds of lock, writelock and readlock, are often used. If the transaction puts writelock on the data entity, other transactions can not access it. On the other hand, if the transaction puts a readlock on the data entity, then it can simultaneously accessed by other transactions with read request.

Correctness of locking and correctness of concurrency control algorithms in general is well known in [2],[4],[7],[16]. The research on the locking performance can be classified into two parts. One is the simulation of locking performance, which can be found in [6],[21]; the other is analytic models of locking performance, see for example [8],[11],[14],[15],[17],[23], [24],[25] and [27]. [18] uses both simulation and analytic models. Most of the analytic models are based on queuing networks or Markov process. The difficulty appeared in the models is the tedious computation caused by a multidimensional state space. To simplify analysis, several strong assumptions are made. On the other hand, Mitra, for example, proposed aggregated models to compute transaction throughput and other locking performance measures. The method avoids the tedious computation to some extent, see [14] and [15]. Although they made a successful effort on modeling one or two locking schemes, their models cannot be extended to other locking schemes, as shown by Lavenberg [13]. In contrast with the models based on queuing networks or Markov process, Tay, Suri and Goodman [23],[24] proposed some practically interesting models for computing the approximate mean

values of locking performance measures. We refer this model as the T Model. The T Model is general enough for dealing with a variety of locking schemes such as uniform and nonuniform access, static and dynamic locking, fixed-length transactions (one class) and indeterminate-length transactions. Compared with simulation results, the analytical results are quite accurate. To simplify analysis, the T Model makes an approximation assumption that the state of a transaction is not affected by its behavior. In reality, a transaction's behavior depends not only on the environment of the database, such as the state of locked and unlocked data entities of the database, for example, but also on the state of the transaction, i.e., how many and what kinds data entities are writelocked and (or) readlocked by the transaction.

In this paper, we try to relax the approximation assumption made in the T Model and predict the system performance at a relatively high precision level. The predictions are made by two steps. In the first step we adopt the approximate assumption for the time being. Through a quantitative analysis of database structure, we give a preliminary estimation of the mean values of performance measures. In the second step we consider both effects of database environment and transaction state and then give the final estimations based on Closed BCMP Networks. To avoid the tedious computations cuased by multidimensional state space, we propose a one-dimensional aggregate system, from which the transaction throughput and restart rate are estimated.

We consider dynamical locking, i.e., locking is done when it is needed, and no waiting, i.e., all conflicts are resolved by restart.

A description of the model is given in section 2. There are n indeterminate-length transactions in the system. Each of them makes a sequence of uniform or non-uniform accesses to the database. If no conflicts occur, the transaction holds a writelock or readlock on a data entity, otherwise, the transaction restarts without waiting.

In section 3, we give the formulation for solving mean value performance measures of the concurrency control model.

Under the approximation assumption, we give a quantitative analysis of

database structure in section 4, from which the preliminary estimations are made.

In section 5 we try to relax the approximation assumption and consider both the effects of database structure and transaction state. Using the results of the preliminary estimation, we give a final estimation based on Closed BCMP Networks. To avoid the tedious computations, we establish an aggregate system to estimate approximate mean value of throughput and restart rate. Meanwhile we obtain several very important equilibrium probabilities of the system as byproducts. In the computation, we will deal with a partition function which is a special form of the normalization constants of Closed BCMP networks. In section 6, we give the recursive relation of the partition function and then a convolution algorithm for calculation is proposed.

We validate our model with simulation results in section 7. Several cases, including uniform and non-uniform access, writelocking and readlocking, fixed length and indeterminate length, are considered. We also make some comparisons with the T Model's results to show the improvement in accuracy.

In section 8, we give the proofs of some propositions. Section 9 is the conclusion.

2. The Model

In this section we give a brief description of the model. The diagram of this model is shown in Fig. 1.

We consider that a database is a set of granules and denote the set by DB. A granule is the smallest entity in the database that may be locked. We assume that there are D granules in DB. Let DB be partitioned into ν kinds of granules, and denote DB_u as a partition of DB, i.e., $DB_u \bigcap DB_{u'} = \emptyset$ for $u \neq u'$, $u, u' = 1, 2, \cdots, \nu$ and $DB = \bigcup_{u=1}^{\nu} DB_u$. Let c_u be the proportion of DB_u to the DB such that $c_u > 0$, $\sum_{u=1}^{\nu} c_u = 1$.

We assume that multiprogramming level is n, which means that n transactions can be processed simultaneously in the system. Since the computer system is closed, n is a constant.

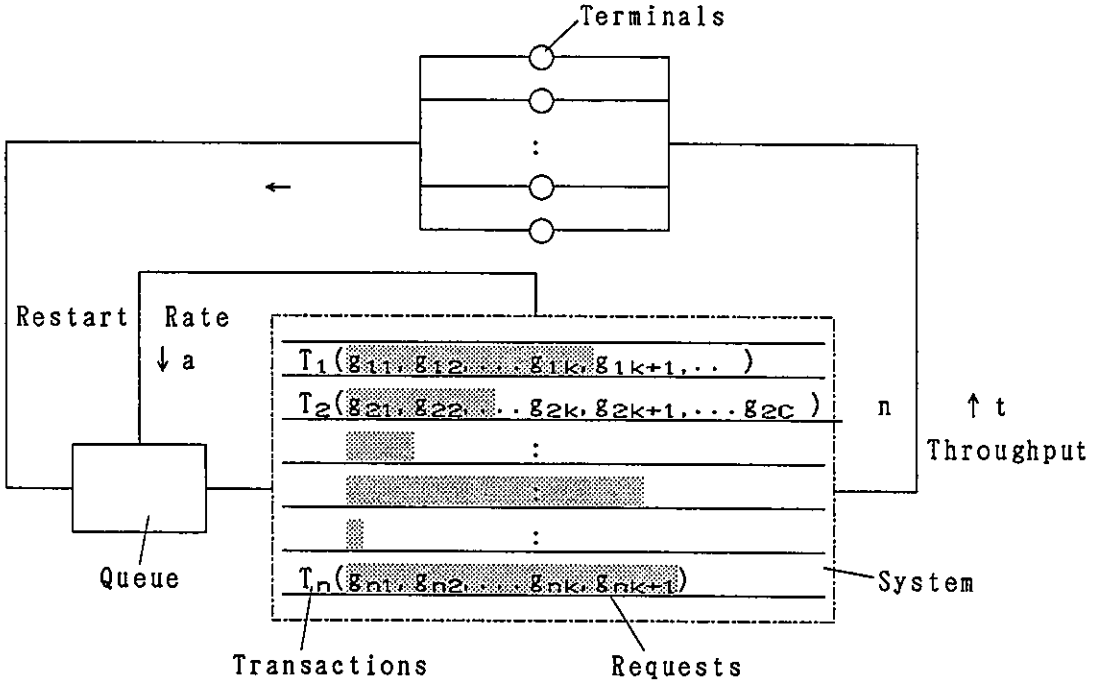

Fig. 1 The diagram of the system. There are n transactions in the system. Each transaction makes a sequence of requests for locking granules of database. t is throughput. a is restart rate.

A transaction has a sequence of requests for locking granules, so its behavior depends not only on the environment of the database, such as how many and what kinds of granules have been locked, but also on the state of the transaction itself, i.e., how many and what kinds of granules have been writelocked and (or) readlocked by the transaction.

The number of granules required by a transaction is called the *length* of the transaction and we classify transactions by their lengths. Let π_k be the probability of transaction in class k and $\pi_0 = 0$. Since the granules required by a transaction are distinct, the transaction must be terminated before the D^{th} request. Hence there is some $C \leq D$, $\pi_k = 0$ for all $k > C$ and $\sum_{k=1}^{C} \pi_k = 1$. Among the requests of a transaction, the first request is for start and the last is for termination. Let p_k, $k = 1, 2, \cdots, C$, be the probability that the k^{th} request is for termination, $p_0 = 0$, then we have that

$$p_k = \frac{\pi_k}{\sum_{j=k}^{C} \pi_j}. \tag{2.1}$$

Assume that every transaction makes requests, independently of its class,

such that it requires a granule of DB_u with the probability of b_u and in DB_u every granule is required equally likely. This means that different kinds of granules may be accessed with different probabilities. Suppose $\nu = 2$, for example, if $c_1 < b_1$, the granules of DB_1 are more often accessed than that of DB_2. So we call the granules of DB_1 regular granules and the that of DB_2 non-regular granules. If $c_1 = b_1$, all granules of DB will be accessed with the same probability. In this case we say DB is uniformly accessed by transactions. In section 7 we will give the detailed description and numerical results of this example.

If several transactions concurrently access a granule of DB, some synchronization anomalies may arise. To avoid such synchronization anomalies, transactions have to lock the required granules before access them. Two kinds of locks, writelock and readlock, are considered in this paper. Writelock, as a exclusive lock, cannot be shared by any other locks. Readlock, however, can be shared but only by other readlocks. Let b_u^w and b_u^r be the probabilities that a transaction requires writelock and readlock on DB_u, respectively. Then $b_u = b_u^w + b_u^r$.

Assume that a transaction holds I_u writelocks and J_u readlocks on DB_u, $u = 1, 2, \ldots, \nu$, and let K be the total number of locks which the transaction held, then $K = \sum_{u=1}^{\nu}(I_u + J_u)$. Let i_u and j_u be the realizations of I_u and J_u, respectively. Hence the vector $\mathrm{s} = (\, i_1, j_1, \ldots, i_\nu, j_\nu \,)$ represents a state of the transaction. We define the state space of a transaction by

$$\mathrm{S} = \{\, \mathrm{s} \mid \mathrm{s} = (\, i_1, j_1, \ldots, i_\nu, j_\nu \,), \quad 0 \le |\mathrm{s}| \le C \,\}, \tag{2.2}$$

where $|\mathrm{s}| = \sum_{u=1}^{\nu} i_u + j_u = k$. Let $N(\mathrm{s})$ be the number of transactions in state s for $\mathrm{s} \in \mathrm{S}$, N_k the number of transactions with k locks and N_L the number of locks held by all transactions. Then we have that

$$n = \sum_{\mathrm{s}\in\mathrm{S}} N(\mathrm{s}) \tag{2.3}$$

$$N_k = \sum_{|\mathrm{s}|=k} N(\mathrm{s}) \quad 0 \le k \le C \tag{2.4}$$

$$N_L = \sum_{\mathrm{s}\in\mathrm{S}} |\mathrm{s}|\, N(\mathrm{s}) \,. \tag{2.5}$$

Some of the information is summarized in Fig. 2. We refer to each node in Fig. 2 as a stage. In stage k, $k = 1, 2, \cdots, C$, there are N_k transactions with k locks. Denoting the throughput of class k by t_k, the throughput of the system by t and the restart rate by a, we have $t = \sum_{k=1}^{C} t_k$.

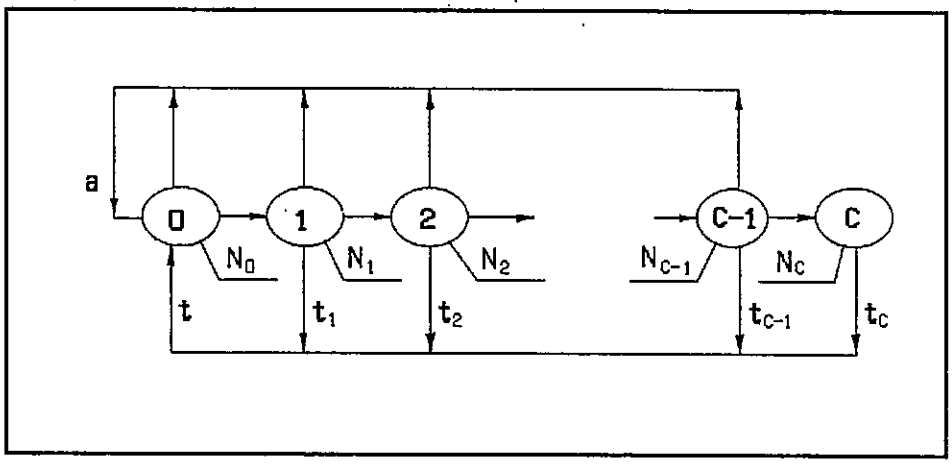

Fig. 2 The flow diagram of the system. We refer each node as a stage. In stage k, $k = 1, 2, \cdots, C$, there are N_k transactions with k locks. t_k is the throughput from class k. t and a are throughput and restart rate of the system, respectively.

Let M be the total number of locked granules of DB, $M \leq D$.

REMARK. *The existence of readlock makes* $M \leq N_L$.

Denote M_u as the number of locked granules of DB_u such that $M = \sum_{u=1}^{\nu} M_u$. Let M_u^w and M_u^r be the number of writelocked and readlocked granules of DB_u, respectively, $M_u = M_u^w + M_u^r$. In Fig. 3, we show the construction of the database when $\nu = 2$.

All requests are handled by a scheduler. The scheduler is a part of database management system which is responsible for the concurrency control.

The time for processing requested granules is independent and identically distributed according to exponential distribution with rate μ.

Suppose that the k^{th} request of a transaction is for writelock (or readlock) a granule of DB_u, if the granule has already been locked (writelocked) by any

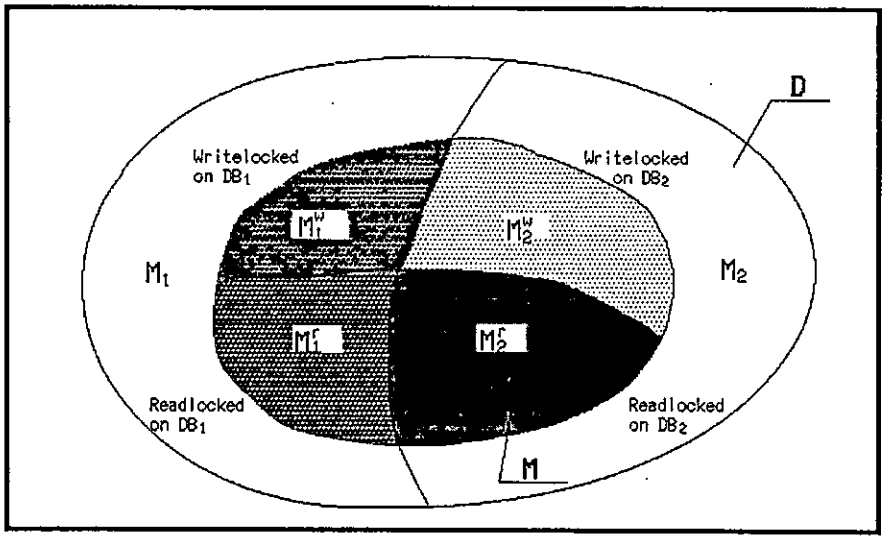

Fig. 3. The construction of the database when $\nu = 2$.

other transaction, we say that the transaction runs into a *conflict* with other transaction. If not, the scheduler grants a writelock (readlock) to the transaction, and then the transaction makes next request. In this way, if the transaction obtains its last required lock and completes processing all its required granules without encountering any conflict, scheduler releases all its locks, the transaction *terminates*. If transaction encounters a conflict at some step of its requests, scheduler releases all its locks and the transaction *restarts*. A restarting transaction stays in a queue waiting for the termination of the conflict transaction, then starts again. While the transaction is waiting, scheduler lets another transaction enter the system. Hence there are n transactions processing in the system all the time.

As a summary of this section, we give the system parameters in List. 1.

D	the number of granules in the database
n	the number of transactions (multiprogramming level)
c_u	the proportion of DB_u to the DB , $u = 1, 2, \cdots, \nu$
b_u	the probability that a transaction requires a granule of DB_u , $u = 1, 2, \cdots, \nu$

b_u^w	the probability that a transaction requires a writelock on DB_u, $u = 1, 2, \cdots, \nu$
b_u^r	the probability that a transaction requires a readlock on DB_u, $u = 1, 2, \cdots, \nu$
k	the length of transaction, $0 \leq k \leq C$.
$N(\mathrm{s})$	the number of transactions in state s, $\mathrm{s} \in \mathrm{S}$
N_k	the number of transactions with k locks
N_L	the total number of locks held by all transactions
π_k	the probability that a transaction is in the class k, $1 \leq k \leq C$, $\pi_0 = 0$
p_k	the termination probability at class k, $1 \leq k \leq C$, $p_0 = 0$
$p_{c,k}$	the conflict probability when a transaction makes its k^{th} request
μ	processing rate
t_k	mean throughput of class k, $1 \leq k \leq C$
t	mean throughput of the system
a	mean restart rate

List 1.

3. The Formulation

The principal mean performance measures we concerned with are throughput and restart rate.

Let $n(\mathrm{s})$ be the realization of $N(\mathrm{s})$ for $\mathrm{s} \in \mathrm{S}$. Then $n(\mathrm{s})$ denotes the number of transactions in state s. We define the state of the system by $\mathbf{n} = (n(0), \cdots, n(\mathrm{s}), \cdots)^\top$. Hence the state space is

$$\Phi = \{\mathbf{n} \mid \mathbf{n} = (n(0), \cdots, n(\mathrm{s}), \cdots)^\top, n(\mathrm{s}) \geq 0,\ \mathrm{s} \in \mathrm{S},\ \sum_{\mathrm{s} \in \mathrm{S}} n(\mathrm{s}) = n\ \}. \quad (3.1)$$

The system is formulated as an irreducible Markov process with finite states. To treat the system, we have to determine how to describe the shifting between two states of the system at first. Rather than using a cumbersome component vector

$(n(0), \cdots, n(\mathrm{s}), \cdots)^{\mathsf{T}}$, we define some shift operators to describe the states. Let $\mathrm{n} \in \Phi$, $\mathrm{s} \in \mathrm{S}$, $\mathrm{s}^{+}_{u,w} = (i_1, j_1, \cdots, i_u+1, j_u, \cdots, i_\nu, j_\nu)$, $\mathrm{s}^{+}_{u,r} = (i_1, j_1, \cdots, i_u, j_u+1, \cdots, i_\nu, j_\nu)$, $u = 1, 2, \cdots, \nu$. Then the shift operators are defined in Table 1.

Shift Operator	*Interpretation*
$\mathrm{S}^{u,w}(\mathrm{n},\ \mathrm{s}) = (n(0), \cdots, n(\mathrm{s}) - 1, \cdots, n(\mathrm{s}^{+}_{u,w}) + 1, \cdots)^{\mathsf{T}}$	$*$
$\mathrm{S}^{u,r}(\mathrm{n},\ \mathrm{s}) = (n(0), \cdots, n(\mathrm{s}) - 1, \cdots, n(\mathrm{s}^{+}_{u,r}) + 1, \cdots)^{\mathsf{T}}$	$**$
$\mathrm{S}^{R}(\mathrm{n},\ \mathrm{s}) = (n(0) + 1, \cdots, n(\mathrm{s}) - 1, \cdots)^{\mathsf{T}}$	$***$

Table 1. When the system is in the state n and a transaction is in the state s , $(*)$ a write request of the transaction is accepted; $(**)$ a read request of the transaction is accepted; $(***)$ a locking request (writelock or readlock) of the transaction is rejected or the transaction terminates at state s .

Let m_u and m^w_u be the realizations of M_u and M^w_u , respectively. Then for $\mathrm{n}, \mathrm{n}' \in \Phi$ we can write the mean transition rate from state n to state n' , denoted by $q(\mathrm{n}, \mathrm{n}')$, as follows. For $\mathrm{s} = (i_1, j_1, \cdots, i_\nu, j_\nu) \in \mathrm{S}$ and $k = |\mathrm{s}|$,

$$
\begin{cases}
q(\mathrm{n}, \mathrm{S}^{u,w}(\mathrm{n},\ \mathrm{s})) = \ \mu(1-p_k) b^w_u \dfrac{c_u D - m_u}{c_u D - i_u - j_u} n(\mathrm{s}) & 0 \le k \le C \\
q(\mathrm{n}, \mathrm{S}^{u,r}(\mathrm{n},\ \mathrm{s})) = \ \mu(1-p_k) b^r_u \dfrac{c_u D - m^w_u}{c_u D - i_u} n(\mathrm{s}) & 0 \le k \le C \\
q(\mathrm{n}, \mathrm{S}^{R}(\mathrm{n},\ 0)) = \ -\mu n + \mu \sum_{u=1}^{\nu} \{ \dfrac{b^w_u m_u}{c_u D} + \dfrac{b^r_u m^w_u}{c_u D} \} n(0) & \\
q(\mathrm{n}, \mathrm{S}^{R}(\mathrm{n},\ \mathrm{s})) = \ \mu(1-p_k) \sum_{u=1}^{\nu} \{ b^w_u \dfrac{m_u - i_u - j_u}{c_u D - i_u - j_u} & \\
\qquad + b^r_u \dfrac{m^w_u - i_u}{c_u D - i_u} \} n(\mathrm{s}) + \mu p_k n(\mathrm{s}) & 1 \le k \le C \\
q(\mathrm{n}, \mathrm{n}') = \ 0 & \text{otherwise.}
\end{cases}
\tag{3.2}
$$

The interpretation of (3.2) is simple. From state n , for instance, the system enters the state $\mathrm{S}^{u,w}(\mathrm{n},\ \mathrm{s})$ if and only if there is no conflict when a transaction makes its k^{th} request for writelock on DB_u . Because of the distinction among requested granules and the uniform access within every DB_u , the system enters $\mathrm{S}^{u,w}(\mathrm{n},\ \mathrm{s})$ with the probability of $(1-p_k) b^w_u \frac{c_u D - m_u}{c_u D - i_u - j_u}$, which depends on the state n and s . If a conflict occurs when a transaction makes its k^{th} request or the k^{th} request is for the termination, the scheduler releases all its locks and then the transaction starts again. The system enters the state of $\mathrm{S}^{R}(\mathrm{n},\ \mathrm{s})$ with

the probability of

$$(1-p_k)\sum_{u=1}^{\nu}\{b_u^w\frac{m_u-i_u-j_u}{c_uD-i_u-j_u}+b_u^r\frac{m_u^w-i_u}{c_uD-i_u}\}+p_k\ . \tag{3.3}$$

Let $P_{\mathbf{n}}$ be the equilibrium probability of the system in state $\mathbf{n}\in\Phi$. Then we can obtain the mean value of throughput and restart rate by

$$t_k=\sum_{\mathbf{n}\in\Phi}P_{\mathbf{n}}\sum_{|\mathbf{s}|=k}\mu p_k n(\mathbf{s}) \qquad k=1,2,\cdots,C \tag{3.4}$$

$$t=\sum_{k=1}^{C}t_k \tag{3.5}$$

$$a=\sum_{\mathbf{n}\in\Phi}P_{\mathbf{n}}\sum_{k=0}^{C-1}\sum_{|\mathbf{s}|=k}\mu(1-p_k)$$
$$\sum_{u=1}^{\nu}\{b_u^w\frac{m_u-i_u-j_u}{c_uD-i_u-j_u}+b_u^r\frac{m_u^w-i_u}{c_uD-i_u}\}n(\mathbf{s}) \tag{3.6}$$

Obviously, the multidimensional state space Φ makes computation very hard. We estimate the mean performance measures by two steps. In the first step, we give a preliminary estimation of the mean value performance measures of the system. Using the results of the preliminary estimations, we give the final estimations based on Closed BCMP Networks in the second step. To avoid the tedious computations, we propose an aggregated stochastic structure to estimate the mean performance measures. In following two sections we give the detailed description of the two step process.

4. Preliminary Estimation

In this section we shall give a preliminary estimation of the mean performance measures of the system.

From (3.2) we know that a transaction of state $\mathbf{s}$ holds a writelock or readlock on DB_u with the probability of

$$b_u^w\frac{c_uD-m_u}{c_uD-i_u-j_u} \qquad \text{or} \qquad b_u^r\frac{c_uD-m_u^w}{c_uD-i_u}\ , \tag{4.1}$$

respectively, where $u=1,2,\cdots,\nu$. To obtain the preliminary estimations, we make one approximation.

APPROXIMATION. *(1) The number of u^{th} kind of granules which are locked by a transaction is small compared to the number of granules of DB_u, i.e., for any $u = 1, 2, \cdots, \nu$, $i_u + j_u \ll c_u D$. (2) Under the average environment of the database, each transaction transfers from s to s', s, s' $\in$ S, according to an independent Markov process as a whole. Therefore, we will approximate the probabilities of writelock and readlock on DB_u by the mean values*

$$x_u = b_u^w(1 - \frac{\bar{M}_u}{c_u D})$$
$$y_u = b_u^r(1 - \frac{\bar{M}_u^w}{c_u D}) \qquad 1 \le u \le \nu \ , \tag{4.2}$$

respectively, where $\bar{M}_u$ and $\bar{M}_u^w$ are the mean number of locked and writelocked granules of DB_u, respectively.

Let $q = \sum_{u=1}^{\nu}(x_u + y_u)$; q is the non-conflict probability under the Approximation. From (4.2) q can be given by

$$q = \sum_{u=1}^{\nu} \{b_u^w(1 - \frac{\bar{M}_u}{c_u D}) + b_u^r(1 - \frac{\bar{M}_u^w}{c_u D})\} \ . \tag{4.3}$$

For given x_u and $y_u, u = 1, 2, \cdots, \nu$, (4.2) and (4.3) imply that from any state a transaction will enter the next state with an independent and identical probability distribution of $x_1, y_1, \cdots, x_\nu, y_\nu, 1-q$. Let $p(\mathrm{s})$ be the steady-state probability of the transaction in the state of s, s $\in$ S. Then we can solve $p(\mathrm{s})$ easily based on the Markov process. From the Approximation, the system Φ is formulated as a Closed BCMP Network with infinite servers in each stage. By using of $p(\mathrm{s})$, we can obtain $P_{\mathbf{n}}$, the equilibrium probability of the system in state n in the next section. In the following proposition we give the steady-state probability $p(\mathrm{s})$. The proof of the proposition can be found in Appendix 1.

PROPOSITION 1. *Let $\alpha_k = \prod_{i=1}^{k-1}(1 - p_i)$, $k = |\mathrm{s}| = 1, 2, \ldots, C$, $\alpha_0 = 1$. Under the Approximation the steady-state probability $p(\mathrm{s})$ can be given by*

$$p(\mathrm{s}) = p(0)\alpha_k \frac{k!}{i_1! j_1! \cdots i_\nu! j_\nu!} \prod_{u=1}^{\nu}(x_u^{i_u} y_u^{j_u}) \ , \tag{4.4}$$

where

$$p(0) = \frac{1}{\sum_{k=0}^{C} \alpha_k q^k} \ . \tag{4.5}$$

Proof: See Appendix 1. □

Let $\bar{N}(\mathrm{s}), \bar{N}_k$ and $\bar{N}_L$ be the expectations of $N(\mathrm{s}), N_k$ and N_L, respectively. Because $\bar{N}_k$ is the mean number of transactions with k locks, and $\bar{N}_L$ is the mean number of locks held by all transactions, we have

$$\bar{N}_k = \sum_{|\mathrm{s}|=k} \bar{N}(\mathrm{s}) \tag{4.6}$$

$$\bar{N}_L = \sum_{\mathrm{s}\in\mathrm{S}} |\mathrm{s}| \bar{N}(\mathrm{s}) \ . \tag{4.7}$$

Since the number of transactions in the system is a constant,

$$n = \sum_{k=0}^{C} \bar{N}_k, \qquad \text{or} \qquad n = \sum_{\mathrm{s}\in\mathrm{S}} \bar{N}(\mathrm{s}) \ . \tag{4.8}$$

For any $q \in (0,1)$ define $f(q) = \frac{\sum_{k=1}^{C} k\alpha_k q^k}{\sum_{k=0}^{C} \alpha_k q^k}$. Then we have that

PROPOSITION 2.

$$\bar{N}_k = \alpha_k q^k \bar{N}_0 \tag{4.9}$$

$$\bar{N}_0 = \frac{n}{\sum_{k=0}^{C} \alpha_k q^k} \tag{4.10}$$

$$\bar{N}_L = n f(q) \ . \tag{4.11}$$

Proof: From (4.4) it is obvious that

$$\bar{N}(\mathrm{s}) = \bar{N}(0) \alpha_k \frac{k!}{i_1! j_1! \cdots i_\nu! j_\nu!} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \ . \tag{4.12}$$

Because $\bar{N}(0) = \bar{N}_0$ and from (4.6) and (4.12) we have

$$\begin{aligned} \bar{N}_k &= \sum_{|\mathrm{s}|=k} \bar{N}_0 \alpha_k \frac{k!}{i_1! j_1! \cdots i_\nu! j_\nu!} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \\ &= \bar{N}_0 \alpha_k (x_1 + y_1 + \cdots + x_\nu + y_\nu)^k \\ &= \alpha_k q^k \bar{N}_0 \ , \end{aligned}$$

it claims (4.9). (4.10) follows from (4.8) and (4.9). (4.11) follows from (4.7) and (4.9). □

From (4.3) we know that the key to solving for q is to estimate $\bar{M}_u$ and $\bar{M}_u^w$, $u = 1, 2, \cdots, \nu$, the structure of the database. We show the estimations in the following proposition.

PROPOSITION 3. *Let $\bar{N}_L$, $\bar{M}_u$ and $\bar{M}_u^w$ be the expectation of N_L, M_u and M_u^w, let $\bar{K}$, $\bar{I}_u$ and $\bar{J}_u$ be the expectation of K, I_u and J_u, respectively. Then we have*

$$\bar{I}_u + \bar{J}_u = \frac{x_u + y_u}{q}\bar{K} \tag{4.13}$$

$$\bar{I}_u = \frac{x_u}{q}\bar{K} \ . \tag{4.14}$$

$$\bar{M}_u \approx \frac{x_u + y_u}{q}\bar{N}_L \tag{4.15}$$

$$\bar{M}_u^w = \frac{x_u}{q}\bar{N}_L \ . \tag{4.16}$$

Proof: Because $\bar{I}_u \equiv E(I_u)$, then we have

$$\begin{aligned}
\bar{I}_u &= \sum_{\mathrm{s}\in\mathrm{S}} i_u p(\mathrm{s}) \\
&= \sum_{k=0}^{C} \sum_{|\mathrm{s}|=k} i_u p(0)\alpha_k \frac{k!}{i_1!j_1!\cdots i_\nu!j_\nu!} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \\
&= p(0)\sum_{k=0}^{C} \alpha_k \Big\{ \sum_{|\mathrm{s}|=k} \frac{i_u k!}{i_1!j_1!\cdots i_\nu!j_\nu!} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \Big\} \\
&= p(0)x_u \sum_{k=0}^{C} k\alpha_k q^{k-1} \\
&= \frac{x_u}{q} p(0) \sum_{k=0}^{C} k\alpha_k q^k \ .
\end{aligned}$$

From (4.5) and a similar computation for $\bar{J}_u$, we have that

$$\bar{I}_u = \frac{x_u}{q} f(q) \tag{4.17}$$

$$\bar{J}_u = \frac{y_u}{q} f(q) \ . \tag{4.18}$$

Recall that $K = \sum_{u=1}^{\nu}(I_u + J_u)$, so $\bar{K} \equiv E(K)$ can be given as follows.

$$\begin{aligned}\bar{K} &= \sum_{\mathbf{s}\in\mathbf{S}} |\mathbf{s}|p(\mathbf{s}) \\ &= \sum_{k=0}^{C}\{ \sum_{|\mathbf{s}|=k} kp(\mathbf{s}) \} \\ &= p(0)\sum_{k=0}^{C} k\alpha_k q^k \ .\end{aligned}$$

Again from (4.5) we have that

$$\bar{K} = f(q) \ . \tag{4.19}$$

Then (4.13) and (4.14) follow from (4.17), (4.18) and (4.19). Since the write-locking is exclusive locking, the number of writelocked granules is equal to the number of writelocks held by transactions. Then from (4.11) we have that $\bar{M}_u^w \equiv E(M_u^w) = n\bar{I}_u = \frac{x_u}{q}\bar{N}_L$, which proves the (4.16).

Readlocking is non-exclusive locking. From the Remark the readlocked granules are not more than the readlocks held by transactions. Since

$$\text{the number of readlocks on } DB_u = nE(J_u) = \frac{y_u}{q}\bar{N}_L \ ,$$

the number of readlocked granules of DB_u can be written as follows:

$$\begin{aligned}\bar{M}_u^r &= c_u D(1-(1-\frac{1}{c_u D})^{\frac{y_u}{q}\bar{N}_L}) \\ &\approx c_u D(1-(1-\frac{1}{c_u D}\frac{y_u}{q}\bar{N}_L)) \\ &= \frac{y_u}{q}\bar{N}_L \ ,\end{aligned} \tag{4.20}$$

hence (4.15) follows from (4.16) and (4.20). □

From (4.15) we have $\sum_{u=1}^{\nu}\bar{M}_u = \bar{M} \approx \bar{N}_L$, which suggests the use of an aggregated stochastic structure of N_L to estimate mean performance measures in the next section.

THEOREM 1. *Let* $\lambda = \frac{n}{D}$ *and* $X(z) = \lambda \frac{f(z)}{z}$. *Then* q *is a root of* $F(z) = 0$, *where* $F(z)$ *is given by*

$$F(z) = \sum_{u=1}^{\nu} \frac{b_u - \frac{b_u^w b_u^r}{c_u} X(z)}{1 + \frac{b_u^w}{c_u} X(z) - \frac{b_u^w b_u^r}{c_u^2} X^2(z)} - z \ . \tag{4.21}$$

Proof: Let $q_u = x_u + y_u$, $u = 1, 2, \cdots, \nu$, where x_u and y_u are given by (4.2). From Proposition 3, we have that

$$\begin{aligned} q_u &= b_u^w (1 - \frac{\bar{M}_u}{c_u D}) + b_u^r (1 - \frac{\bar{M}_u^w}{c_u D}) \\ &= b_u^w (1 - \frac{q_u n f(q)}{q c_u D}) + b_u^r (1 - \frac{x_u n f(q)}{q c_u D}) \\ &= b_u - n \frac{f(q)}{q c_u D} (b_u^w q_u + b_u^r x_u) \\ &= b_u - n \frac{f(q)}{q c_u D} [b_u^w q_u + b_u^r (1 - \frac{q_u n f(q)}{q c_u D})] \ . \end{aligned}$$

Replacing $X(q)$ by $\lambda \frac{f(q)}{q}$, we have that

$$q_u = \frac{b_u - \frac{b_u^w b_u^r}{c_u} X(z)}{1 + \frac{b_u^w}{c_u} X(z) - \frac{b_u^w b_u^r}{c_u^2} X^2(z)} \ . \tag{4.22}$$

Since $\sum_{u=1}^{\nu} q_u = q$, $F(q) = \sum_{u=1}^{\nu} q_u - q = 0$, which proves the theorem.

□

THEOREM 2. *When* $b_u^r = 0$, $1 \le u \le \nu$, $F(z)$ *has exactly one root in* $(0, 1)$.

Proof: When $b_u^r = 0$, $b_u = b_u^w$. Hence $F(z)$ can be written as follows.

$$F(z) = \sum_{u=1}^{\nu} \frac{1}{\frac{1}{b_u} + \frac{1}{c_u} \frac{\lambda f(z)}{z}} - z.$$

Let $\tilde{F}(z) = \frac{F(z)}{z}$, we have

$$\tilde{F}(z) = \sum_{u=1}^{\nu} \frac{1}{\frac{z}{b_u} + \frac{\lambda}{c_u} f(z)} - 1. \tag{4.23}$$

We know that $\tilde{F}(z)$ and $F(z)$ have the same root in $(0,1)$. Since $f(0)=0$, $\tilde{F}(z)\to\infty$ when $z\to 0$. On the other hand, because of $f(1)>0$, then $\tilde{F}(1)\le\sum_{u=1}^{\nu}\frac{1}{\frac{1}{b_u}+0}-1=0$. Since $\tilde{F}(z)$ is continuous in $(0,1)$, it has at least one root in $(0,1)$. Furthermore, we have that

$$\frac{d}{dz}f(z)=\frac{1}{z}\{\sum_{k=0}^{C}k^2\frac{\alpha_k z^k}{\sum_{i=1}^{C}\alpha_i z^i}-(\sum_{k=0}^{C}k\frac{\alpha_k z^k}{\sum_{i=1}^{C}\alpha_i z^i})^2\}. \tag{4.24}$$

Let $\tilde{p}_k=\frac{\alpha_k z^k}{\sum_{i=1}^{C}\alpha_i z^i}$. Then

$$\frac{d}{dz}f(z)=\frac{1}{z}\{\sum_{k=0}^{C}k^2\tilde{p}_k-(\sum_{k=0}^{C}k\tilde{p}_k)^2\}, \tag{4.25}$$

which means that $z\frac{d}{dz}f(z)$ can be regarded as a variance of some probability distribution. Hence $\frac{d}{dz}f(z)\ge 0$, i.e., $f(z)$ increases in z. From (4.23) it follows that $\tilde{F}(z)$ decreases in z, which means that $\tilde{F}(z)$ has only one root in $(0,1)$.

□

When $b_u^r\neq 0$, we can prove the existence of a root $q\in(0,1)$ for some small $\lambda=\frac{n}{D}$, but we can not prove its uniqueness. However through a great number of simulation results, we found that the root is always unique even if $b_u^r\neq 0$. Now we obtain the preliminary estimation of the mean throughput and restart rate, which are given in the following theorem.

THEOREM 3. *Under the Approximation, the preliminary estimation of the mean throughput and restart rate can be written as follows.*

$$t_k=\mu p_k\bar{N}_k=\mu p_k\frac{nq^k}{\sum_{k=0}^{C}\alpha_k q^k} \tag{4.26}$$

$$t=\sum_{k=1}^{C}t_k \tag{4.27}$$

$$a=\sum_{k=0}^{C-1}\mu(1-p_k)p_c\bar{N}_k=\sum_{k=0}^{C-1}\mu(1-p_k)p_c\frac{nq^k}{\sum_{k=0}^{C}\alpha_k q^k}. \tag{4.28}$$

Though Theorem 3 gives almost the same results as [24]'s in appearance, q, which is given by Theorem 1, is different from [24]'s. In the case of uniform access and writelock only, i.e., when $b_u = c_u = b_u^w, b_u^r = 0$, the preliminary estimation gives the same results as [24]. In the rest of cases, the preliminary estimation is more accurate than [24]'s results as will be shown in section 7. As the summary of this section, we give the algorithm of the preliminary estimation.

program Preliminary Estimation
begin
read($D, n, C, c_u, b_u^w, b_u^r, \pi_k$) ;
solve $F(z) = 0$ for q ;
calculate q_u, x_u, $\bar{M}_u$, $\bar{M}_u^w$;
calculate p_k, p_c ;
calculate t_k, t, a ;
end.

5. Aggregated Model

In this section we try to relax the Approximation and consider both the effects of the database environment and the transaction state. Using the results of the preliminary estimation given in previous section, we make the final estimation based on Closed BCMP Networks ([1],[9]). To avoid the tedious computation mentioned in section 3, we shall propose a aggregated stochastic structure to estimate the mean value of throughput t and restart rate a. We consider that among system parameters shown in List 1, the most important one which associated with conflict behaviors is M, the total number of locked granules in DB. Since a direct measure of M is difficult, from Proposition 3, we can use N_L, the total number of locks held by all transactions, to estimate M indirectly. We will call this the N_L Model. Let m be a realization of N_L, then the state space of N_L can be written as follows.

$$\Psi = \{m \mid 0 \leq m \leq nC \}. \tag{5.1}$$

Clearly, it is a one-dimensional irreducible, finite state Markov process. We define a mapping as follows.

$$\{\ \mathrm{n} \mid \mathrm{n} = (n(0), \cdots, n(\mathrm{s}), \cdots)^{\top} \in \Phi,\ \sum_{\mathrm{s} \in \mathrm{S}} |\mathrm{s}| n(\mathrm{s}) = m\ \} \quad \overset{mapping}{\longmapsto} \quad m \in \Psi, \tag{5.2}$$

In this way states of Φ which satisfy the right hand of (5.2) are aggregated as one state of Ψ. Hence through N_L the multidimensional state space Φ can be aggregated into the one-dimensional state space Ψ.

When the database is in state $m \in \Psi$, i.e., there are m locks held by all transactions, from Proposition 3, we know that m_u and m_u^w, the number of locks and writelocks holding on DB_u, respectively, can be estimated by

$$m_u \approx \frac{q_u}{q} m\ , \qquad m_u^w \approx \frac{x_u}{q} m\ ,$$

where $q_u = x_u + y_u$ and x_u, y_u are given by (4.2). If a transaction is in state $\mathrm{s} \in \mathrm{S}$ and $|\mathrm{s}| = k$, also from Proposition 3, we can estimate $i_u + j_u$ and i_u, the number of locks and writelocks which the transaction holds on DB_u, respectively, as follows,

$$i_u + j_u \approx \frac{q_u}{q} k\ , \qquad i_u \approx \frac{x_u}{q} k\ .$$

Denote the conflict probability of a transaction of state s when $N_L = m$ by $p_{c,\mathrm{s}}(m)$ and the conflict probability at the k^{th} request of a transaction when $N_L = m$ by $p_{c,k}(m)$, where $k = |\mathrm{s}|$. Then we have the following approximation relation between $p_{c,\mathrm{s}}(m)$ and $p_{c,k}(m)$.

$$\begin{aligned} p_{c,\mathrm{s}}(m) &= \sum_{u=1}^{\nu} \Big\{ b_u^w \frac{m_u - i_u - j_u}{c_u D - i_u - j_u} + b_u^r \frac{m_u^w - i_u}{c_u D - i_u} \Big\} \\ &\approx \sum_{u=1}^{\nu} \Big\{\ b_u^w \frac{m-k}{c_u \dfrac{q}{q_u} D - k} + b_u^r \frac{m-k}{c_u \dfrac{q}{x_u} D - k}\ \Big\} = p_{c,k}(m)\ . \end{aligned} \tag{5.3}$$

For $m, m' \in \Psi$ let $q_{m,m'}$ be the steady-state mean transition rate from state m to state m'. In Fig. 4 we show the state transition diagram of the system.

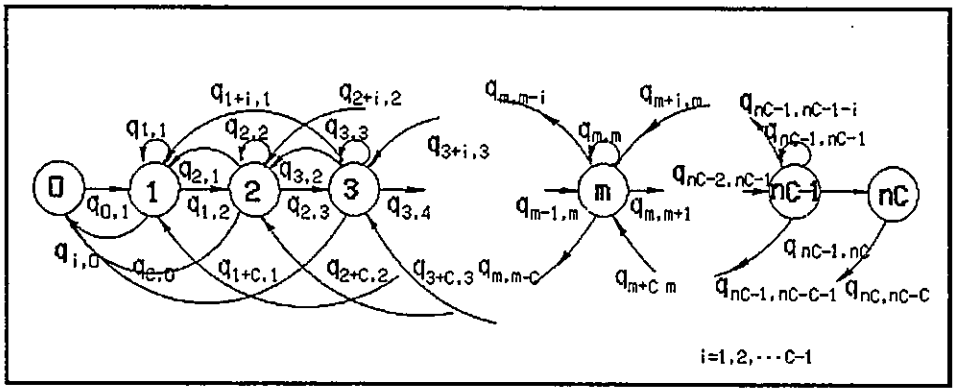

Fig. 4. The state transition diagram of the N_L system

We define the region

$$\Re_{n,m} = \{\ \mathrm{n} \mid \mathrm{n} \in \Phi, \sum_{\mathrm{s}\in\mathrm{S}} N(\mathrm{s}) = n,\ \sum_{\mathrm{s}\in\mathrm{S}} |\mathrm{s}| N(\mathrm{s}) = m\ \}\ . \tag{5.4}$$

Let $\bar{N}_k(\mathrm{s}|m)$ be the steady-state conditional expectation of $N(\mathrm{s})$ such that

$$\begin{aligned} \bar{N}(\mathrm{s}|m) &= E\{\ N(\mathrm{s})|\ \mathrm{n} \in \Re_{n,m}\ \} \\ &= \sum_{\mathrm{n}\in\Re_{n,m}} \frac{n(\mathrm{s}) P_{\mathrm{n}}}{\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}}}\ . \end{aligned} \tag{5.5}$$

Let $\bar{N}_k(m)$ be the steady-state conditional expectation of N_k, i.e., the number of transactions with k locks when $N_L = m$, so that $\bar{N}_k(m) = \sum_{|\mathrm{s}|=k} \bar{N}(\mathrm{s}|m)$, then we have

PROPOSITION 4. *Let $\bar{N}_k(m)$, $0 \le k \le C$, $m \in \Psi$, be the steady-state conditional expectation of N_k and $p_{c,k}(m)$ be the conflict probability. Then the mean transition rate $q_{m,m'}$, m, $m' \in \Psi$, can be given as follows.*

$$\begin{cases} q_{m,m} = -n\mu + \mu p_{c,0}(m)\bar{N}_0(m) & m = 1,\cdots,nC-1 \\ q_{m,m+1} = \mu \sum_{k=0}^{C-1}(1-p_k)(1-p_{c,k}(m))\bar{N}_k(m) & m = 0,1,\cdots,nC-1 \\ q_{m,m-C} = \mu \bar{N}_C(m) & m = C, C+1,\cdots,nC \\ q_{m,m-k} = \mu((1-p_k)p_{c,k}(m) + p_k)\bar{N}_k(m) & m = 1,\cdots,nC-1; \\ & k = 1,\cdots,\min\{m, C-1\} \\ q_{m,m'} = 0 & \text{otherwise.} \end{cases} \tag{5.6}$$

Proof: See Appendix 2. □

To distinguish the results of the Preliminary Estimation from that of the N_L Model's, we denote $\tilde{t}_k$, $\tilde{t}$ and $\tilde{a}$ as the mean performance measures given by the N_L Model. Let $P(m)$ be the steady-state probability of the system in state m , $m \in \Psi$. Then we have that

$$p_{c,k} = \sum_{m=0}^{nC} p_{c,k}(m)P(m) \tag{5.7}$$

$$\tilde{t}_k = \sum_{m=k}^{nC} \mu p_k \bar{N}_k(m)P(m) \tag{5.8}$$

$$\tilde{t} = \sum_{k=1}^{C} \tilde{t}_k \tag{5.9}$$

$$\tilde{a} = \sum_{k=0}^{C-1} \sum_{m=k}^{nC-1} \mu(1-p_k)p_{c,k}(m)\bar{N}_k(m)P(m). \tag{5.10}$$

To solve for $p_{c,k}$, $\tilde{t}$ and $\tilde{a}$, we have to calculate $\bar{N}_k(m)$, $0 \le k \le C$, and $P(m)$, $m \in \Psi$ first. In the rest of this section, we give the estimations of $\bar{N}_k(m)$ based on Closed BCMP Networks, and then we give $P(m)$ through solving a Markov process.

Approximate Calculation of $\bar{N}_k(m)$

In order to calculate $\bar{N}_k(m)$ we have to know the equilibrium distribution of Φ given in (3.1). To simplify analysis, we adopt the Approximation here once again. Under the Approximation, we can regard the system of Φ as a Closed BCMP Queuing Networks System. Then a product form equilibrium distribution, denoted by P_{n} , $\mathrm{n} \in \Phi$, can be obtained as follows.(see [1],[9],[10],[19]).

$$P_{\mathrm{n}} = \frac{1}{G(n)} \prod_{0 \le |\mathrm{s}| \le C} \frac{p(\mathrm{s})^{n(\mathrm{s})}}{n(\mathrm{s})!}, \tag{5.11}$$

where

$$G(n) = \sum_{\mathrm{n} \in \Phi} \prod_{0 \le |\mathrm{s}| \le C} \frac{p(\mathrm{s})^{n(\mathrm{s})}}{n(\mathrm{s})!} \tag{5.12}$$

and $p(\mathrm{s})$, $\mathrm{s} \in \mathrm{S}$, is given in (4.4) and (4.5). Since $\bar{N}_k(\mathrm{s}|m)$ is the steady-state conditional expectation of $N(\mathrm{s})$, then from (5.5) we have that

PROPOSITION 5.

$$\bar{N}(\mathrm{s}|m) = p(\mathrm{s})\frac{G^1(m-k,n-1,C)}{G^1(m,n,C)}\ , \qquad ((5.13))$$

where $k = |\mathrm{s}|$, *and*

$$G^1(m,n,C) = \sum_{\mathrm{n}\in\Re_{n,m}} \prod_{0\le k\le C} \frac{p(\mathrm{s})^{n(\mathrm{s})}}{n(\mathrm{s})!}\ . \qquad (5.14)$$

Proof: Because

$$\bar{N}(\mathrm{s}|m) \equiv E\{\ N(\mathrm{s})|\ \mathrm{n}\in\Re_{n,m}\ \},$$

then it follows from (5.11) that

$$\begin{aligned}
\bar{N}(\mathrm{s}|m) &= \sum_{\mathrm{n}\in\Re_{n,m}} \frac{n(\mathrm{s})P_{\mathrm{n}}}{\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}}} \\
&= \sum_{\mathrm{n}\in\Re_{n,m}} \frac{p(\mathrm{s}) \prod\limits_{\substack{0\le|\mathrm{S}'|\le C\\ \mathrm{S}'\neq\mathrm{S}}} (\frac{p(\mathrm{s}')^{n(\mathrm{S}')}}{n(\mathrm{s}')!})(\frac{p(\mathrm{s})^{(n(\mathrm{S})-1)}}{(n(\mathrm{s})-1)!})}{G^1(m,n,C)} \\
&= \frac{p(\mathrm{s}) \sum\limits_{\mathrm{n}'\in\Re_{n-1,m-k}} \prod\limits_{0\le|\mathrm{s}|\le C} \frac{p(\mathrm{s})^{n'(\mathrm{s})}}{n'(\mathrm{s})!}}{G^1(m,n,C)} \\
&= p(\mathrm{s})\frac{G^1(m-k,n-1,C)}{G^1(m,n,C)}.
\end{aligned}$$

□

We define a *partition function* $H(m,n,C)$ by

$$H(m,n,C) = \sum_{\substack{n_0+\cdots+n_C=n\\ n_1+2n_2+\cdots+Cn_C=m}} \prod_{k=0}^{C} \frac{\alpha_k^{n_k}}{n_k!}. \qquad (5.15)$$

Note that in our model, n and C are constants. For the sake of convenience in computation of the partition function, we represent n and C as variables in (5.15).

THEOREM 4. *Under the Approximation, we have that*

$$\begin{aligned}\bar{N}_k(m) &= \sum_{|\mathrm{s}|=k} \bar{N}(\mathrm{s}|m) \\ &= \alpha_k \frac{H(m-k, n-1, C)}{H(m,n,C)} \ . \end{aligned} \tag{5.16}$$

Proof: From (5.13), we have that

$$\begin{aligned}\bar{N}_k(m) &= \sum_{|\mathrm{s}|=k} \bar{N}(\mathrm{s}|m) \\ &= \sum_{|\mathrm{s}|=k} p(\mathrm{s}) \frac{G^1(m-k, n-1, C)}{G^1(m,n,C)} \\ &= p(0)\alpha_k q^k \frac{G^1(m-k, n-1, C)}{G^1(m,n,C)}), \end{aligned} \tag{5.17}$$

where $p(0)$ is given by (4.5). From (5.14) we have that

$$\begin{aligned} G^1(m,n,C) &= \sum_{\mathrm{n}\in\Re_{n,m}} \prod_{0\le k\le C} \frac{p(\mathrm{s})^{n(\mathrm{s})}}{n(\mathrm{s})!} \\ &= \sum_{\substack{\sum_{\mathrm{s}\in\mathrm{S}} n(\mathrm{s})=n \\ \sum_{\mathrm{s}\in\mathrm{S}} |\mathrm{s}| n(\mathrm{s})=m}} \prod_{k=0}^{C} \frac{1}{n_k!} \Big\{ \frac{n_k!}{\prod_{|\mathrm{s}|=k} n(\mathrm{s})!} \prod_{|\mathrm{s}|=k} p(\mathrm{s})^{n(\mathrm{s})} \Big\} \\ &= \sum_{\substack{\sum_{k=0}^{C} \{\sum_{|\mathrm{s}|=k} n(\mathrm{s})\}=n \\ \sum_{k=1}^{C} \{|\mathrm{s}| \sum_{|\mathrm{s}|=k} n(\mathrm{s})\}=m}} \prod_{k=0}^{C} \frac{1}{n_k!} \Big\{ \frac{n_k!}{\prod_{|\mathrm{s}|=k} n(\mathrm{s})!} \prod_{|\mathrm{s}|=k} p(\mathrm{s})^{n(\mathrm{s})} \Big\} \\ &= \sum_{\substack{\sum_{k=0}^{C} n_k=n \\ \sum_{k=1}^{C} k n_k=m}} \prod_{k=0}^{C} \frac{1}{n_k!} \Big\{ \sum_{|\mathrm{s}|=k} \frac{n_k!}{\prod_{|\mathrm{s}|=k} n(\mathrm{s})!} \prod_{|\mathrm{s}|=k} p(\mathrm{s})^{n(\mathrm{s})} \Big\} \\ &= \sum_{\substack{\sum_{k=0}^{C} n_k=n \\ \sum_{k=1}^{C} k n_k=m}} \prod_{k=0}^{C} \frac{1}{n_k!} \Big\{ \sum_{|\mathrm{s}|=k} p(\mathrm{s}) \Big\}^{n_k} \\ &= \sum_{\substack{\sum_{k=0}^{C} n_k=n \\ \sum_{k=1}^{C} k n_k=m}} \prod_{k=0}^{C} \frac{1}{n_k!} \Big\{ p(0)\alpha_k q^k \Big\}^{n_k} . \end{aligned}$$

It follows that

$$G^1(m,n,C) = p^n(0)q^m H(m,n,C)\ . \tag{5.18}$$

Combining (5.17) and (5.18) we have the claim. □

The Calculation of $P(m)$

Now we begin to solve $P(m)$, the equilibrium probability of the N_L system at state m , $m \in \Psi$. Let $\mathbf{p} = [P(0), P(1), \cdots, P(nC)]^\top$ and $\mathbf{Q}$ be the transition matrix with the elements given by (5.6). Then we obtain $\mathbf{p}$ by solving the following equations.

$$\mathbf{p}^\top \mathbf{Q} = \mathbf{0}. \tag{5.19}$$

From (5.6) we know that $\mathbf{Q}$ is the skip-free-to-right transition rate matrix, i.e., the transition from state i to state j is not possible if $j > i+1$. The Markov decision process with skip-free-to-right transition has been discussed by [26]. By using this property, we can solve Eq.(5.19) with the condition of $\sum_{m=0}^{nC} P(m) = 1$. The idea is quite simple: let $P(nC) = 1$ at first, then solve $P(nC-1), P(nC-2), \cdots$, one by one, finally normalize them.

Up to now we have obtained $\bar{N}_k(m)$ and $P(m)$. From (5.7) $\sim$ (5.10), we have

THEOREM 5. *The approximate mean throughput t and restart rate a can be given as follows.*

$$p_{c,k} = \sum_{m=0}^{nC} p_{c,k}(m) P(m) \tag{5.20}$$

$$\tilde{t}_k = \sum_{m=k}^{nC} \mu p_k \frac{H(m-k, n-1, C)}{H(m,n,C)} P(m) \tag{5.21}$$

$$\tilde{t} = \sum_{k=1}^{C} \tilde{t}_k \tag{5.22}$$

$$\tilde{a} = \sum_{k=0}^{C-1} \sum_{m=k}^{nC} \mu(1-p_k) p_{c,k}(m) \frac{H(m-k, n-1, C)}{H(m,n,C)} P(m)\ . \tag{5.23}$$

In next section, we give an algorithm for solving the partition function $H(m,n,C)$. As the summary of this section, we give the algorithm of the N_L Model.

program N_L **Model**

begin

read($D, n, C, c_u, b_u^w, b_u^r, \pi_k$) ;

read(q_u, x_u) from Preliminary Estimation ;

calculate $p_{c,k}(m)$;

calculate $H(m,n-1,C)$, $H(m,n,C)$, $m \in \Psi$;

calculate $\bar{N}_k(m)$;

solve $\mathrm{pQ} = 0$ for $\mathrm{p} = [p(0), \cdots, p(nC)]$;

calculate $p_{c,k}$, $\tilde{t}_k$, $\tilde{t}$, $\tilde{a}$;

end.

6. Partition Function

From Theorem 4 and Theorem 5, we know that once the values of partition function $H(m,n,C)$, $H(m,n-1,C)$, $m \in \Psi$, have been calculated, it is possible to compute $\bar{N}_k(m)$, $\tilde{t}_k$ and $\tilde{a}$ efficiently.

$H(m,n,C)$ is a special case of $G(\mathrm{n})$, is the normalization constant of Closed BCMP Networks proposed by [1] and [9]. To compute this normalization constant $G(\mathrm{n})$, the *Convolution Analysis* method [3],[12] and the *Mean Value Analysis* method [5],[20],[22] have been proposed. We use the first one to calculate $H(m,n,C)$. Before giving the convolution algorithm, we give the recursive relation and some special properties of $H(m,n,C)$ in the following proposition.

PROPOSITION 6. *Let the partition function* $H(m,n,k)$ *be given by (5.15). For* $k = 1, 2, \cdots, C$, *we have that*

$$H(m,n,k) = \sum_{n_k = L}^{U} \frac{\alpha_k^{n_k}}{n_k!} H(m - kn_k, n - n_k, k-1) \quad \text{with}$$

$$L = max\{0, m - n(k-1)\}, \quad U = \lfloor \frac{m}{k} \rfloor \tag{6.1}$$

$$H(m,n,k) = H(m,n,k-1) \qquad 0 \le m < k-1 \tag{6.2}$$

$$H(m,1,k) = \alpha_k \tag{6.3}$$

$$H(0,0,k) = 1\ . \tag{6.4}$$

Proof: We give the proof of (6.1) and (6.2). From (5.15) we know that

$$\begin{aligned} H(m,n,k) &= \sum_{\substack{n_0+\cdots+n_k=n \\ n_1+2n_2+\cdots+kn_k=m}} \prod_{i=0}^{k} \frac{\alpha_i^{n_i}}{n_i!} \\ &= \sum_{\substack{n_0+\cdots+n_k=n \\ n_1+2n_2+\cdots+kn_k=m}} \prod_{i=0}^{k-1} \frac{\alpha_i^{n_i}}{n_i!} \frac{\alpha_k^{n_k}}{n_k!} \\ &= \sum_{n_k} \{ \sum_{\substack{\sum n_i = n-n_k \\ \sum i n_i = m-kn_k}} \prod_{i=0}^{k-1} \frac{\alpha_i^{n_i}}{n_i!} \} \frac{\alpha_k^{n_k}}{n_k!} \\ &= \sum_{n_k} \frac{\alpha_k^{n_k}}{n_k!} H(m-nk, n-n_k, k-1)\ . \end{aligned} \tag{6.5}$$

Since $m - nk \ge 0$ and $m - nk \le (n - n_k)(k-1)$, we have

$$max\{0, m-n(k-1)\} \le n_k \le \lfloor \frac{m}{k} \rfloor\ . \tag{6.6}$$

Hence (6.5) and (6.6) imply (6.1). Note that $0 \le m \le k-1$ if and only if $n_k = 0$. Hence

$$\begin{aligned} H(m,n,k) &= \sum_{\substack{n_0+\cdots+n_k=n \\ n_1+2n_2+\cdots+kn_k=m}} \prod_{i=0}^{k} \frac{\alpha_i^{n_i}}{n_i!} \\ &= \sum_{\substack{n_0+\cdots+n_{k-1}=n \\ n_1+2n_2+\cdots+(k-1)n_{k-1}=m}} \prod_{i=0}^{k-1} \frac{\alpha_i^{n_i}}{n_i!} \\ &= H(m,n,k-1)\ . \end{aligned}$$ □

Taking these properties into account, we propose an algorithm which is efficient in saving memory and computing time.

i / $N_i(m)$	0		1		2		3		4		5	
m	ESTIMA:	SIMULA	ESTIMA:	SIMULA	ESTIMA:	SIMULA	ESTIMA:	SIMULA	ESTIMA:	SIMULA	ESTIMA:	SIMULA
0	5.000	5.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	4.000	3.333	1.000	1.667	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	3.333	3.050	1.333	1.717	0.333	0.234	0.000	0.000	0.000	0.000	0.000	0.000
3	2.857	2.997	1.429	1.364	0.571	0.639	0.143	0.000	0.000	0.000	0.000	0.000
4	2.500	2.581	1.429	1.202	0.714	0.909	0.286	0.252	0.071	0.056	0.000	0.000
5	2.222	2.346	1.389	1.286	0.794	0.814	0.397	0.392	0.159	0.133	0.040	0.029
6	1.951	2.049	1.366	1.295	0.854	0.871	0.488	0.480	0.244	0.222	0.098	0.083
7	1.705	1.643	1.311	1.348	0.918	0.859	0.574	0.699	0.328	0.302	0.164	0.149
8	1.488	1.557	1.238	1.192	0.952	0.913	0.667	0.661	0.417	0.455	0.238	0.222
9	1.296	1.338	1.157	1.161	0.963	0.968	0.741	0.707	0.519	0.516	0.324	0.310
10	1.121	1.125	1.075	1.099	0.960	0.914	0.799	0.774	0.614	0.634	0.430	0.455
11	0.952	0.956	0.993	1.010	0.952	0.893	0.850	0.842	0.707	0.694	0.544	0.604
12	0.801	0.775	0.897	0.857	0.936	0.992	0.897	0.830	0.801	0.868	0.667	0.678
13	0.667	0.658	0.801	0.836	0.897	0.904	0.936	0.989	0.897	0.801	0.801	0.812
14	0.544	0.589	0.707	0.680	0.850	0.880	0.952	0.880	0.993	1.039	0.952	0.933
15	0.430	0.397	0.614	0.640	0.799	0.736	0.960	0.986	1.075	1.066	1.121	1.176
16	0.324	0.310	0.519	0.498	0.741	0.723	0.963	0.990	1.157	1.193	1.296	1.285
17	0.238	0.193	0.417	0.464	0.667	0.651	0.952	1.059	1.238	1.187	1.488	1.444
18	0.164	0.162	0.328	0.356	0.574	0.573	0.918	0.798	1.311	1.384	1.705	1.726
19	0.098	0.073	0.244	0.199	0.488	0.510	0.854	0.925	1.366	1.259	1.951	2.033
20	0.040	0.038	0.159	0.156	0.397	0.549	0.794	0.758	1.389	1.447	2.222	2.054
21	0.000	0.000	0.071	0.077	0.286	0.500	0.714	0.652	1.429	1.791	2.500	1.980
22	0.000	0.000	0.000	0.000	0.143	0.000	0.571	0.633	1.429	1.023	2.857	3.344
23	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.250	1.333	1.500	3.333	3.250
24	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	4.000	5.000
25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	5.000	5.000

Table 2. The numerical results of $\bar{N}_k(m)$ with $D = 200, n = 5, C = 5$. We compare the results with simulation ones. ESTIMA=estimation results, SIMULA=simulation results (average values).

program H **computation**

begin

read(C, n);

calculate $H(m, n, 1)$;

for $k = 2$ **to** C **do**

  if $k =$ odd **then** $c_1 := 1, c_2 := 2$

  else $c_1 := 2, c_2 := 1$;

  for $n' = 1$ **to** n **do**

    for $m = 0$ **to** $k - 1$ **do**

      $H(m, n', c_1) := H(m, n', c_2)$;

    for $m = k$ **to** kn' **do**

      $H(m, n', c_1) := \sum_{n_k=L}^{U} \frac{\alpha_k^{n_k}}{n_k!} H(m - n'k, n' - n_k, c_2)$;

end.

Making use of $H(m, n, C)$, we can estimate $\bar{N}_k(m)$, the mean values of $N_k(m)$ (see (5.16)). We consider that it is the key of aggregated model. In Table.2 we give the numerical results of (5.16) and compare them with the simulation runs.

7. Performance Evaluation and Validation

In this section, we shall validate our analytical results against the simulation ones. By changing the system parameters, we apply the model to the following four cases.

Table 3.

Case	*Subject*
1	Uniform access, fixed length and writelock only
2	Non-uniform access, fixed length and writelock only
3	Non-uniform access, fixed length and write-read lock
4	Non-uniform access, variable length and write-read lock

In each case, we predict the main performance measures, throughput and restart rate of the system and verify these predicted values through simulation results. We also make some comparisons with the T Model's results [24] to show the improvement in accuracy. In non-uniform access cases (case 2 $\sim$ case 4), we let $\nu = 2$, i.e., there are two kinds of granules in the DB. To simplify the representation, let $b = b_1, (b_2 = 1 - b)$, $c = c_1, (c_2 = 1 - c)$, and $c < b$. We know that each request of a transaction will make two choices: the choice of granules and the choice of locking. In following numerical computation, we assume that these choices are independent. Let p_w be the probability of a writelock request. Then it follows that

$$b_1^w = bp_w \ , \qquad b_1^r = b(1 - p_w) \ ,$$
$$b_2^w = (1 - b)p_w \ , \qquad b_2^r = (1 - b)(1 - p_w) \ .$$

The values of system parameters used in the following numerical computations are given in Table 4. Before showing the numerical results, we give a brief description of the simulator.

Table 4.

Case	D	n	C	b	c	p_w	π_k
1	1000	$5 \sim 50$	15	$b = c$	$b = c$	1.0	$\pi_{15} = 1.0$
2	2500	$5 \sim 50$	15	vary	vary	1.0	$\pi_{15} = 1.0$
3	2500	$5 \sim 50$	15	0.8	0.2	vary	$\pi_{15} = 1.0$
4	2500	$5 \sim 50$	15	0.8	0.2	0.7	vary

The simulator is a discrete-event simulation program. Each transaction has a sequence of required granules. By the rule of distinctness and uniform access in one of DB_u, these granules are chosen when transaction makes request.

The time for processing all requests is according to exponential distribution with the rate of μ (See section 2). The *run time* is measured with a *clock*. When the simulation begins, the clock is set to 0, the transactions are initiated, and then the time for processing the first requests of these transactions are chosen. These requests are put on the *request queue* and then dequeued one by one. When a request departs from the queue, the clock is incremented by the processing time specified by this request.

On the turn of a transaction's request, the transaction declares the request of granule and lock. If the request is accepted, the time for processing the next request is chosen and then the request is put into the request queue. If the request is rejected, i.e., a conflict occurs, then the transaction restarts. When a transaction restarts or terminates, all its locks are released and another transaction is initiated with new requests.

The granules and lock of a transaction needed and the processing time of all requests are chosen with the help of a pseudorandom number generator.

The length of a simulation run is specified by the number of requests generated. We choose the length as $20,000$ and all our simulation results are the mean values of 10 runs.

To simplify representation, in Fig. 5 $\sim$ Fig.11 we refer to the preliminary estimation model mentioned in section 4 as the PE Model, and the aggregated model mentioned in section 5 as the N_L Model.

Case 1.

Let $b_u = c_u$, $u = 1,2$, $\pi_{15} = 1.0$, $\pi_k = 0$, $k = 0,1,\cdots,14$, and $b_u^r = 0$, $u = 1,2$. Then we apply the model to the case of uniform access, writelocking only and fixed length transaction. We begin to validate our model from this simplest case. Fig.5 gives the comparison of the predicted values of the N_L Model with the T Model's and simulation results. We omit the preliminary

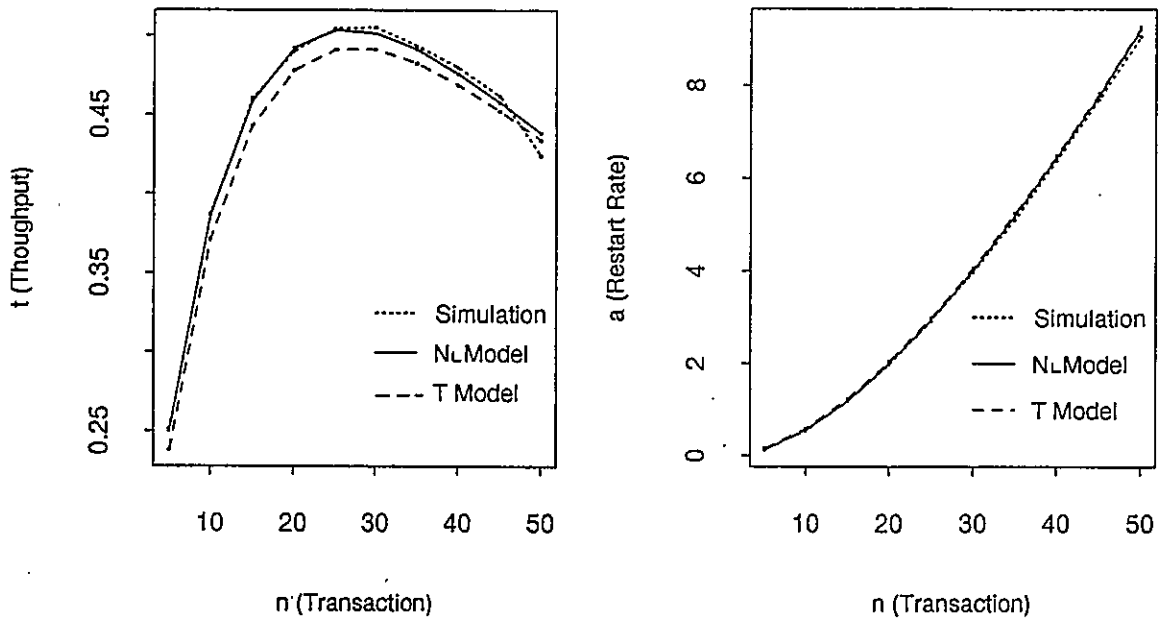

Fig. 5. Validation for case 1: uniform access, fixed length and writelock only. $D = 1000, C = 15, \mu = 1$.

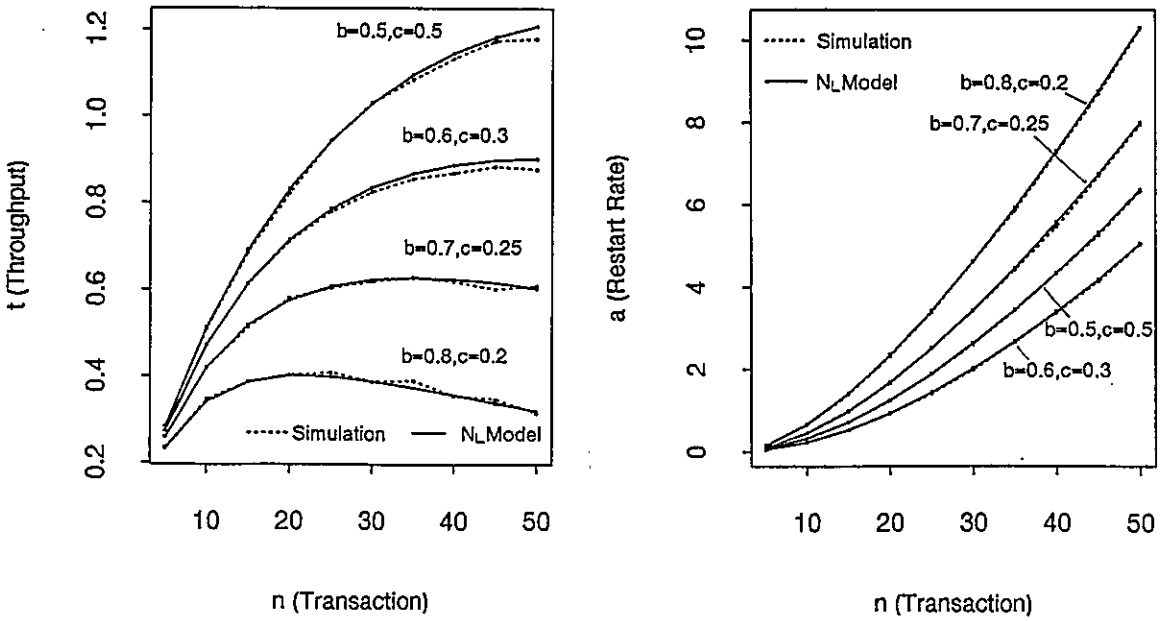

Fig. 6. Case 2: non-uniform access, fixed length and writelock only. The effect of change in b and c. $D = 2500, C = 15, \mu = 1$.

estimation results here, because they are the same as the T Model's results in case 1 (see Theorem 3). Fig. 5 shows that the N_L Model makes the better predictions than the T Model do. The system parameters used in case 1 are given in table 4.

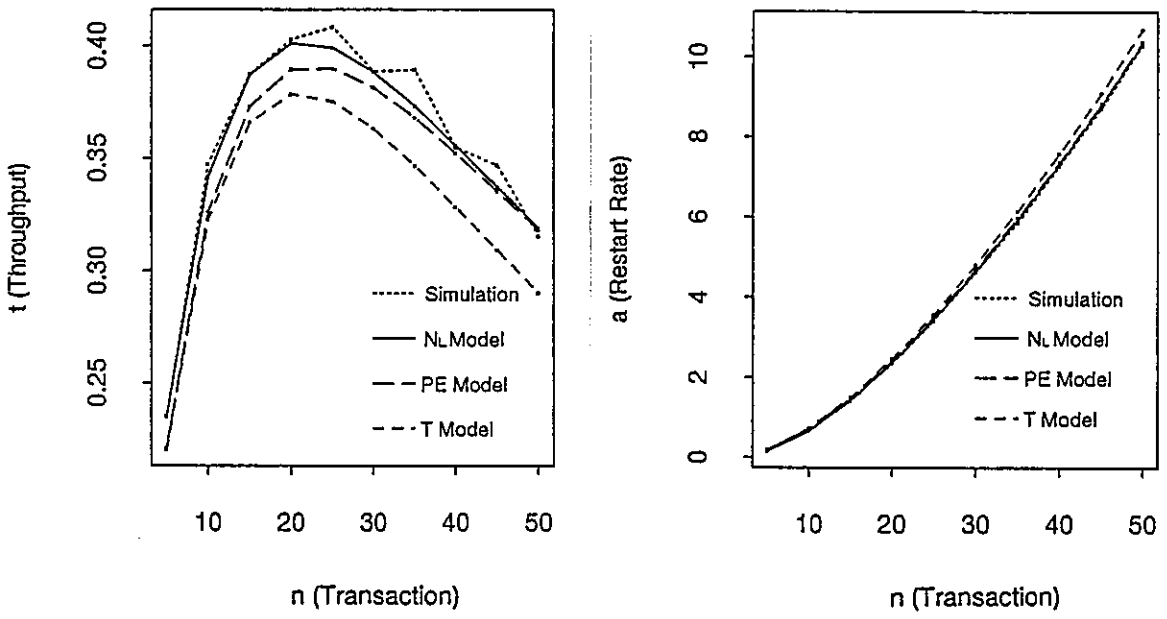

Fig. 7. Validation for case 2: non-uniform access, fixed length and writelock only. $D = 2500, C = 15, b = 0.8, c = 0.2, \mu = 1$.

Case 2.

By varying the values of b and c, we relax the uniform access here. Let $b = 0.8, c = 0.2$, for instance, which means that 80 percent of transactions' requests fall within 20 percent of the database. Fig. 6 shows that the more concentrated transactions' requests are, the more conflict will occur. Fig. 7 gives a validation of our analytical results with $b = 0.8, c = 0.2$. Fig. 6 and Fig. 7 show that the PE Model and the N_L Model improve the T Model step by step. The N_L Model represents DC-thrashing very accurately.

Case 3.

In Case 3 we change the values of p_w to see how the system performance will be affected. Fig. 8 shows that with the decrease of writelock requests, conflict will be relaxed. Fig. 9 validates analytical results in the write-read locking case. The validation results show that the N_L Model is accurate enough for practical application. If the estimation of $\tilde{M}_u^r$ (see (4.20)) can be improved, a better result may be obtained.

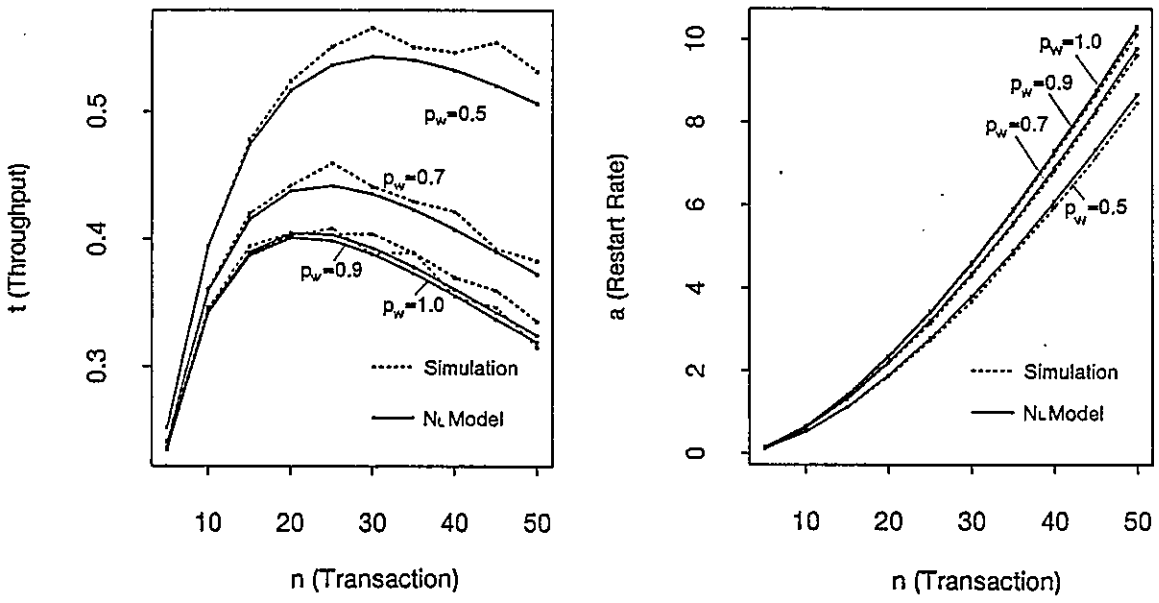

Fig. 8. Case 3: non-uniform access, fixed length and write-read lock. The effect of change in p_w . $D = 2500, C = 15, b = 0.8, c = 0.2, \mu = 1$.

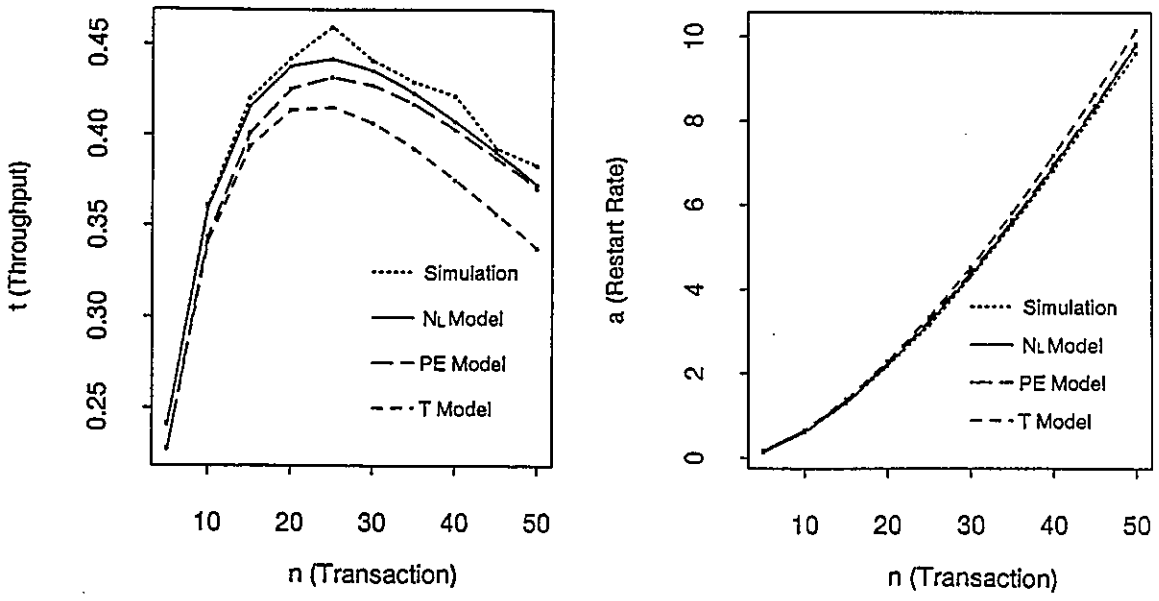

Fig. 9. Validation for case 3: non-uniform access, fixed length and write-read lock. $D = 2500, C = 15, b = 0.8, c = 0.2, p_w = 0.7, \mu = 1$.

Case 4.

Finally, we consider the variance of $\pi_k, 0 \leq k \leq C$. We know that the more granules required by transactions, the more granules will be locked In contrast with the fixed transaction length cases, variable length transaction will reduce the mean requests for granules, and so should result in the relaxation of conflict. Fig. 10 tests the fact. Fig. 11 gives the validation of case 4.

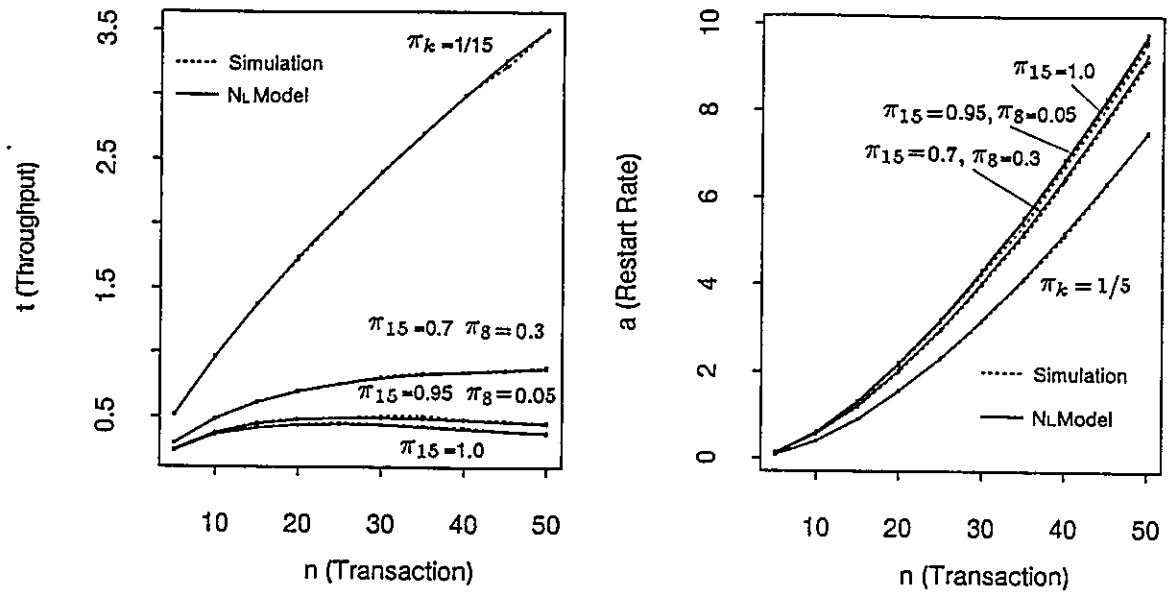

Fig. 10. Case 4: non-uniform access, variable length and write-read lock. The effect of change in π_k . $D = 2500, C = 15, b = 0.8, c = 0.2, p_w = 0.7, \mu = 1$.

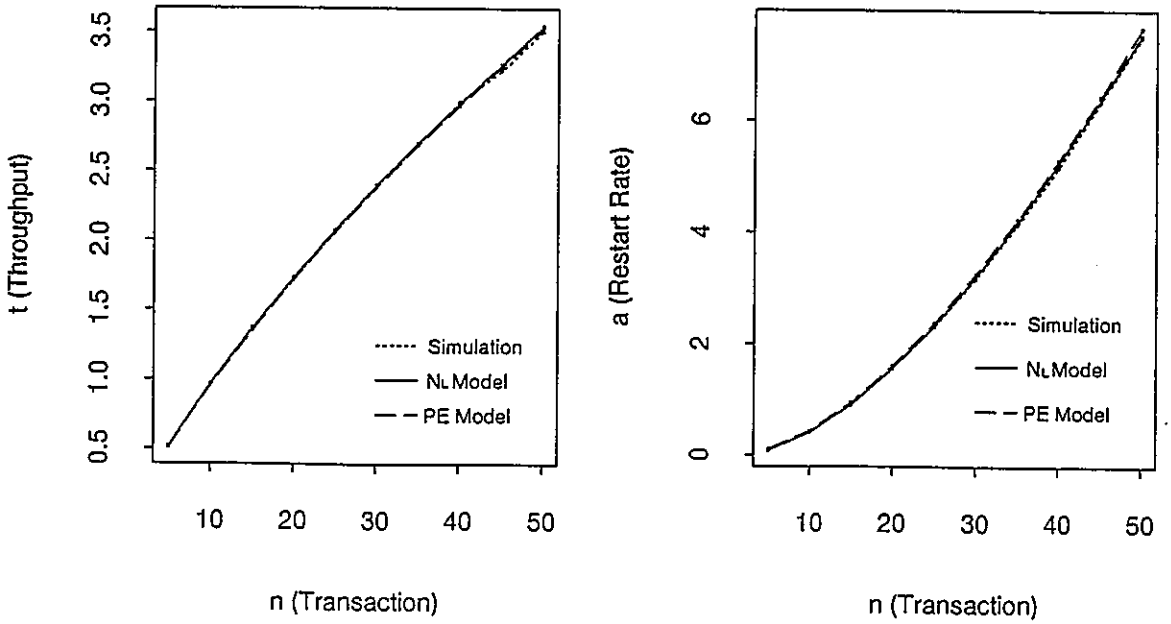

Fig. 11. Validation for case 4: non-uniform access, variable length and write-read lock. $D = 2500, C = 15, b = 0.8, c = 0.2, p_w = 0.7, \pi_k = 1/15, k = 1, 2, \cdots, C, \mu = 1$.

We have validated the model under the four cases and shown that our model fits well with simulation results. The comparisons of our model with the T Model show an improvement in accuracy. Through the quantitative analysis of the database structure in the preliminary estimation, the PE Model makes the first improvement over the T Model. Since we consider the both effects of the database environment and transaction state in the final estimation, the N_L Model makes a further improvement over the T Model. As further studies, we

think that improvement of (4.20) and the extension of the no-waiting model to the waiting model are needed.

8. Appendix

(Appendix 1.) The Proof of Proposition 1

Proof: Let $\mathrm{s} \in \mathrm{S}$, $\mathrm{s}^-_{u,w} = (i_1, j_1, \cdots, i_u - 1, j_u, \cdots, i_\nu, j_\nu)$ and $\mathrm{s}^-_{u,r} = (i_1, j_1, \cdots, i_u, j_u - 1, \cdots, i_\nu, j_\nu)$. The steady rates flowing into and flowing out from state s are

$$\text{input rate} = \mu(1 - p_{k-1}) \sum_{u=1}^{\nu} \{x_u p(s^-_{u,w}) + y_u p(s^-_{u,r})\} ,$$

$$\begin{aligned} \text{output rate} &= \mu \sum_{u=1}^{\nu} (x_u + y_u) p(\mathrm{s}) + \mu(1 - \sum_{u=1}^{\nu} (x_u + y_u)) p(\mathrm{s}) \\ &= \mu p(\mathrm{s}) . \end{aligned}$$

Then we have the flow-balance equation such that

$$p(\mathrm{s}) = (1 - p_{k-1}) \sum_{u=1}^{\nu} \{x_u p(\mathrm{s}^-_{u,w}) + y_u p(\mathrm{s}^-_{u,r})\} . \tag{8.1}$$

Replacing (4.4) into the right hand side of (8.1), we have that

$$\begin{aligned} &\text{the right hand side of (8.1)} \\ &= p(0)(1 - p_{k-1}) \alpha_{k-1} \sum_{u=1}^{\nu} \Big\{ \frac{x_u (k-1)!}{i_1! j_1! \cdots (i_u - 1)! \cdots j_\nu!} \frac{1}{x_u} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \\ &\qquad + \frac{y_u (k-1)!}{i_1! j_1! \cdots (j_u - 1)! \cdots j_\nu!} \frac{1}{y_u} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \Big\} \\ &= p(0)(1 - p_{k-1}) \alpha_{k-1} \sum_{u=1}^{\nu} \Big\{ \frac{(k-1)!}{i_1! j_1! \cdots (i_u - 1)! \cdots j_\nu!} \\ &\qquad + \frac{(k-1)!}{i_1! j_1! \cdots (j_u - 1)! \cdots j_\nu!} \Big\} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \\ &= p(0) \alpha_k \frac{k!}{i_1! j_1! \cdots i_\nu! j_\nu!} \prod_{u=1}^{\nu} (x_u^{i_u} y_u^{j_u}) \\ &= \text{the left hand side of (8.1)}, \end{aligned}$$

hence (4.4) is a root of (8.1). Since $\sum_{\mathrm{S}} p(\mathrm{s}) = 1$, (4.5) follows. □

(Appendix 2.) The Proof of Proposition 4

Proof: We give the proof of $q_{m,m+1}$. The rest of the equations can be proved in the same way. According to the aggregate relation between the state space Φ and the state space Ψ , we have that

$$q_{m,m+1} = \sum_{\mathrm{n}\in\Re_{n,m}} \sum_{\mathrm{s}\in\mathrm{S}} \sum_{u=1}^{\nu} \{q(\mathrm{n}, \mathrm{S}^{u,w}(\mathrm{n},\ \mathrm{s})) + q(\mathrm{n}, \mathrm{S}^{u,r}(\mathrm{n},\ \mathrm{s}))\} \times P_{\mathrm{n}} \Big(\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}} \Big)^{-1} \tag{8.2}$$

From (3.2) and (8.2), we have that

$$\begin{aligned} q_{m,m+1} &= \sum_{\mathrm{n}\in\Re_{n,m}} \sum_{k=0}^{C} \sum_{|\mathrm{s}|=k} \sum_{u=1}^{\nu} \mu(1-p_k)\Big\{b_u^w \frac{m_u - i_u - j_u}{c_u D - i_u - j_u} + b_u^r \frac{m_u^w - i_u}{c_u D - i_u}\Big\} \\ &\quad \times n(\mathrm{s}) P_{\mathrm{n}} \Big(\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}} \Big)^{-1} \\ &\approx \mu \sum_{\mathrm{n}\in\Re_{n,m}} \sum_{k=0}^{C} (1-p_k)(1-p_{c,k}(m)) \sum_{|\mathrm{s}|=k} n(\mathrm{s}) P_{\mathrm{n}} \Big(\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}} \Big)^{-1} \\ &= \mu \sum_{\mathrm{n}\in\Re_{n,m}} \Big\{ \sum_{k=0}^{C} (1-p_k)(1-p_{c,k}(m)) \Big\} n_k P_{\mathrm{n}} \Big(\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}} \Big)^{-1} \\ &= \mu \sum_{k=0}^{C} (1-p_k)(1-p_{c,k}(m)) \sum_{\mathrm{n}\in\Re_{n,m}} n_k P_{\mathrm{n}} \Big(\sum_{\mathrm{n}\in\Re_{n,m}} P_{\mathrm{n}} \Big)^{-1} \\ &= \mu \sum_{k=0}^{C-1} (1-p_k)(1-P_{c,k}(m)) \bar{N}_k(m) \ . \end{aligned}$$ □

9. Conclusions

In this paper we have proposed an analytic model to predict the mean value performance of database concurrency control. The predictions are made through two steps: the preliminary estimation and final estimation. The model is general enough to handle the cases of uniform and non-uniform access, writelock and readlock, and fixed and indeterminate length of transactions. The validation of the model under a variety of cases shows that our model improves the T Model in accuracy by two steps. The contributions of this paper are : (1) Based on Closed BCMP Networks, we give a quantitative analysis of database structure. (2) A one-dimensional aggregate system was used. Since the system avoids cumbersome representation and tedious computations, the analysis and predictions can be done at a relatively high precision level.

References

[1] F. Baskett, K.M. Chandy, R.R. Muntz and F.G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers", *Journal of the Association for Computing Machinery* Vol.22 No.2 (1975) 248-260.

[2] P.A. Bernstein and N. Goodman, "Concurrency control in distributed database systems", *Computing Surveys* Vol.13 No.2 (1981) 185-221.

[3] J.P. Buzen, "Computational algorithms for closed queueing networks with exponential servers", *Communications of the ACM* Vol.16 No.9 (1973) 527-531.

[4] M.A. Casanova, *The Concurrency Control Problem for Database Systems* (Springer-Verlag, Berlin, 1981).

[5] A.E. Conway, E. de Souza e Silva and S.S. Lavenberg, "Mean value analysis by chain of product form queueing networks", *IEEE Transactions on Computers* Vol.38 No.3 (1989) 432-442.

[6] C. Devor, "Structural locking mechanisms and their effect on database management system performance", *Information Systems* Vol.7 No.4 (1982) 345-358.

[7] K.P. Eswaran, "The notations of consistency and predicate locks in a database system", *Communications of the ACM* Vol.19 No.11 (1976) 624-633.

[8] B.I. Galler and L. Bos, "A model of transaction blocking in database", *Performance Evaluation* 3 (1983) 95-122.

[9] W.J. Gordon and G.F. Newell, "Closed queuing systems with exponential servers", *Operations Research* 15 (1967) 254-265.

[10] D. Gross, *Fundamentals of Queueing Theory* (John Wiley & Sons, New York, 1985).

[11] C.A. Hartzman, "The delay due to dynamic two-phase locking", *IEEE Transactions on software engineering* Vol.15 No.1 (1989) 72-83.

[12] S.S. Lam and Y.L. Lien, "A tree convolution algorithm for the solution of queueing networks", *Communications of the ACM* Vol.26 No.3 (1983) 203-215.

[13] S.S. Lavenberg, "A perspective on queueing models of computer performance", in: *Queueing Theory and its Applications* (North- Holland, Amsterdam, 1988) pp. 59-94.

[14] D. Mitra, "Probabilistic models and asymptotic results for concurrent processing with exclusive and non-exclusive locks", *SIAM Journal of the Computers* Vol.14 No.4 (1985) 1030-1051.

[15] D. Mitra and P.J. Weinberger, "Probabilistic models of database locking: solutions, computational algorithms, and asymptotics", *Journal of the Association for Computing Machinery* Vol.31 No.4 (1984) 855-878.

[16] C. Papadimitrion, *The Theory of Database Concurrency Control* (Computer Science Press, USA, 1986).

[17] D. Potier and Ph. Lerlanc, "Analysis of locking policies in database management systems", *Communications of the ACM* Vol.23 No.10 (1980) 584-593.

[18] K.H. Pun and G.G. Belford, "Performance study of two locking in single-site database systems", *IEEE Transactions on software engineering* Vol.13 No.12 (1987) 1311-1328.

[19] M. Reiser and H. Kobayashi, "Queueing networks with multiple closed chain: Theory and computational algorithms ", *IBM J. Res. Dev.* 19 (1975) 283-294.

[20] M. Reiser and S.S. Lavenberg, "Mean value analysis of closed multichain queueing networks ", *Journal of the Association for Computing Machinery* Vol.27 No.2 (1980) 313-322.

[21] D.R. Ries and M. Stonebraker, "Effects of locking granularity in a database management system", *ACM Transactions on Database Systems* Vol.2 No.3 (1977) 233-246.

[22] E. de Souza e Silva and R.R. Muntz, "Simple relations among moments of queue lengths in product form queueing networks", *IEEE Transactions Computers* Vol.37 No.9 (1988) 1125-1129.

[23] Y.C. Tay, N. Goodman and R. Suri, "Locking performance in centralized database", *ACM Transactions on Database Systems* Vol.10 No.4 (1985) 415-462.

[24] Y.C. Tay, R. Suri and N. Goodman, "A mean value performance model for locking in database: the no-waiting case", *Journal of the Association for Computing Machinery* Vol.32 No.3 (1985) 618-651.

[25] A. Thomasian and I.K. Ryu, "A recursive solution method to analyze the performance of static locking systems", *IEEE Transactions on software engineering* Vol.15 No.10 (1989) 1147-1156.

[26] J. Wijngaard and S. Stidham, "Forward recursion for Markov decision processes with skip-free-to-the-right transition, part 1: theory and algorithms", *Mathematics of Operations Research* Vol.11 (1986) 295-308.

[27] S.W. Yeh, C. Ellis, A. Ege and H.F. Korth, "Performance analysis of two concurrency control schemes for design environments", *Information Sciences* Vol.49 No.1 (1989) 3-33.