PUC

OPTIMISTIC VERSUS PESSIMISTIC CONCURRENCY CONTROL MECHANISMS IN DATABASE MANAGEMENT SYSTEMS

by

Daniel A. Menascé
Pontificia Universidade Católica do Rio de Janeiro

Tatuo Nakanishi
Escola Federal de Engenharia de Itajubá

3/80

Technical Report DB 127901 - December 1979
Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 - CEP 22453
Rio de Janeiro — Brasil

OPTIMISTIC VERSUS PESSIMISTIC CONCURRENCY CONTROL MECHANISMS IN DATABASE MANAGEMENT SYSTEMS

by

Daniel A. Menascé

Tatuo Nakanishi

OPTIMISTIC VERSUS PESSIMISTIC CONCURRENCY CONTROL MECHANISMS IN DATABASE MANAGEMENT SYSTEMS*

Daniel A. Menascé
Tatuo Nakanishi

Departamento de Informática
Pontifícia Universidade Católica do
Rio de Janeiro, Brasil

1. INTRODUCTION

Several concurrency control mechanisms for database systems have appeared in the literature during the past few years. Some of them use locking as the basic serialization mechanism [GRAY 78, ESWA 76] . Locking oriented concurrency control can be considered pessimistic since database resources are locked eventhough transactions may not conflict with other executing transactions. Instead of locking, a conflict oriented scheme could be used. A transaction is divided into a read phase, a computation phase and a test and update phase. If the data objects read are detected as still valid during the test and update phase, the values calculated during the computation phase are used to update the database. Otherwise the transaction is rejected and it must restart again. This approach can be considered as optimistic since data objects are not locked hoping that they will not be modified by other transactions. Examples of optimistic concurrency control mechanisms can be found in [KUNG 79, BADAL 79] .

---

* This research was partially supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)

This paper presents a performance evaluation of locking oriented and conflict oriented concurrency control mechanisms. Analitic models as well as simulation models are used to obtain the average response time of transactions under both types of mechanisms.

Section 2 presents the necessary background. Section 3 presents an analysis which considers the database as a single resource. In the following section, results are obtained for a model which considers the database as a collection of resources. The paper concludes with an analysis of the results obtained.

All the results presented in the paper are proved in the appendix.

2. CONCURRENCY CONTROL

The semantic integrity of a database may be violated when the database is subject to concurrent access. Therefore, the database management system must provide concurrency control mechanisms to control the update activity in the database. Two kinds of concurrency control mechanisms may be envisioned: locking oriented concurrency control (LOCC) and conflict oriented concurrency control (COCC) mechanisms.

A lock* is a serialization mechanism wich protects the owner of the lock from modifications from other users. Therefore, while a lock on object X is active, no other user can acess that object. In a sense, the use of locking is pessimistic since the object must be locked eventhough no other user may want to access the object concurrently. Certain rules must be followed to lock and unlock objects in order to obtain serializable logs. Two phase locking [ESWA 76] is an example of such rules.

Our model of a LOCC mechanism is as follows. Upon arrival, a transaction will try to acquire all the database resources it needs. If it succeeds it will start executing, otherwise it must wait in a queue. Deadlocks are prevented by acquiring all the locks at once, at the beginning of transaction. When a transaction completes, it may free resources which are needed by transactions waiting in the queue. At this point, all the transactions wich have all the necessary resources may start to execute. Ties between conflicting transactions are broken on a FCFS basis.

The second type of concurrency control mechanism considered here is the conflict oriented one.

---

* We will only be referring to exclusive locks in this paper.

A transaction is divided into three phases: a read phase, a computation phase and a test and update phase.

During the read phase the transaction reads all the necessary data (the read set). We will assume that there is a timestamp associated to each database resource as in [THOMAS 79]. These timestamps are read along with the data and will be used later during the test and update phase.

During the computation phase, the new values are generated for the data objects to be updated. When all the computation is done, the test and update phase begins. During this phase, the current values for the time stamps of the read set of the transaction are obtained and compared with the timestamps obtained during the read phase. If all the timestamps are current, the updates generated during the previous phase are applied to the database. Otherwise, the transaction is rejected and it must restart from the read phase.

This approach can be considered optimistic, in the sense that the database resources are not locked hoping that no other transaction will interfere with the current one.

In the following sections we will analyze the response time for transactions under LOCC and COCC mechanisms.

3. THE SINGLE RESOURCE MODEL

In this section the database is modeled as a single resource. For the LOCC mechanism, this is equivalent to saying that every transaction must lock the entire database. For the COCC mechanism, this is equivalent to assigning a single timestamp to the whole database.

We will assume that transactions arrive at a rate λ transactions per second and that they are generated from a Poisson process.

3.1. Analysis of the COCC Mechanism

In this section we will obtain the probability density function (p.d.f.) for a transaction response time, R. Let the time to execute the transaction once (read phase + computation phase + test and update phase) be exponentially distributed with average $\frac{1}{\mu}$ seconds. Let p denote the probability that a transaction does not conflict during its test and update phase. This is illustrated in figure 1.

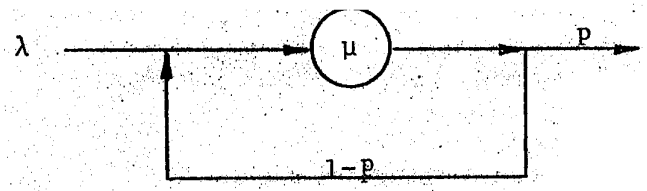

Figure 1 - COCC Mechanism

The results given in this paper will be proved in the Appendix.

Result 1: The probability $p = (1+\rho)^{-1}$, where $\rho = \lambda/\mu$.

In order to validate the results we have simulated the system. Table 1 shows several values of p as a function of ρ according to the analytic model and according to the simulation model. These results are also plotted in figure 2.

p \ ρ	0.25	0.5	1.0	2.0	3.0	4.0	5.0
Analytic Results	0.80	0.66	0.50	0.33	0.25	0.20	0.16
Simulation Results	0.79	0.65	0.47	0.28	0.20	0.17	0.13

Table 1 - Single Resourse Model (COCC)

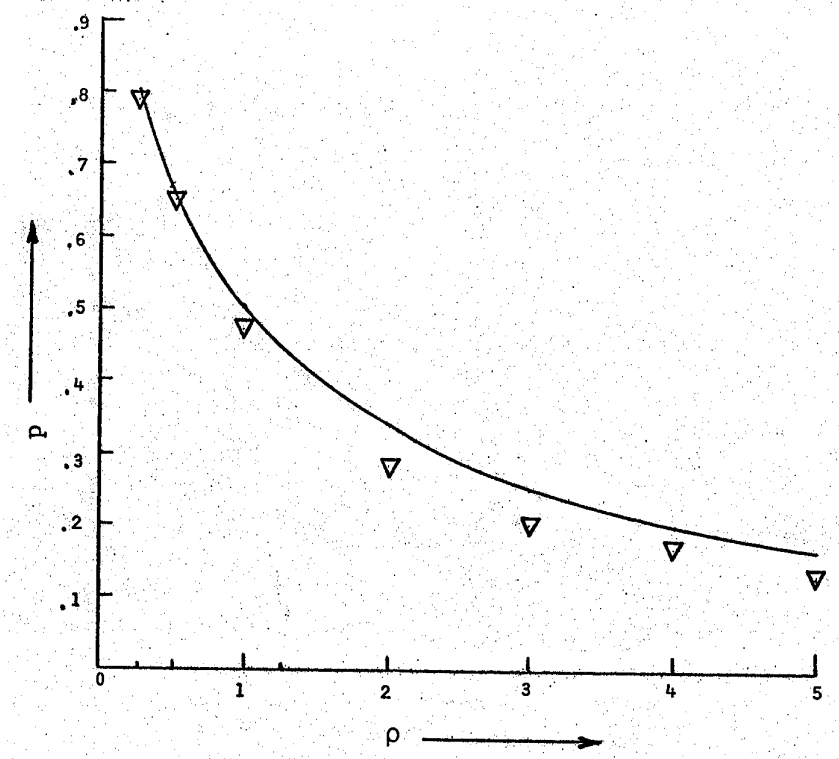

Figure 2 - Probability of Success COCC
Single Resource

—— analytic result ∇ simulation results

Given the probability p, we can find the Laplace Transform, $R^*(s)$, of the response time, $\tilde{R}$, of a transaction.

Result 2: The Laplace transform $R^*(s)$ is given by

$$R^*(s) = \frac{p\, B^*(s)}{1-(1-p)B^*(s)}$$

where $B^*(s)$ is the Laplace transform of the time to execute the transaction once. In our case, $B^*(s) = \mu/(s+\mu)$. This gives us our next result.

Result 3: When $B^*(s) = \mu/(s+\mu)$, the response time $\tilde{R}$ is exponentially distributed with p.d.f., $r(t)$ given by

$$r(t) = \mu p e^{-\mu p t} \quad \text{for } t \geq 0$$

The average response time, R, is therefore equal to $1/(\mu\rho)$, which as a function of ρ is iqual to

$$R = (1+\rho)/\mu$$

Figure 3 illustrates the variation of μR as a function of ρ

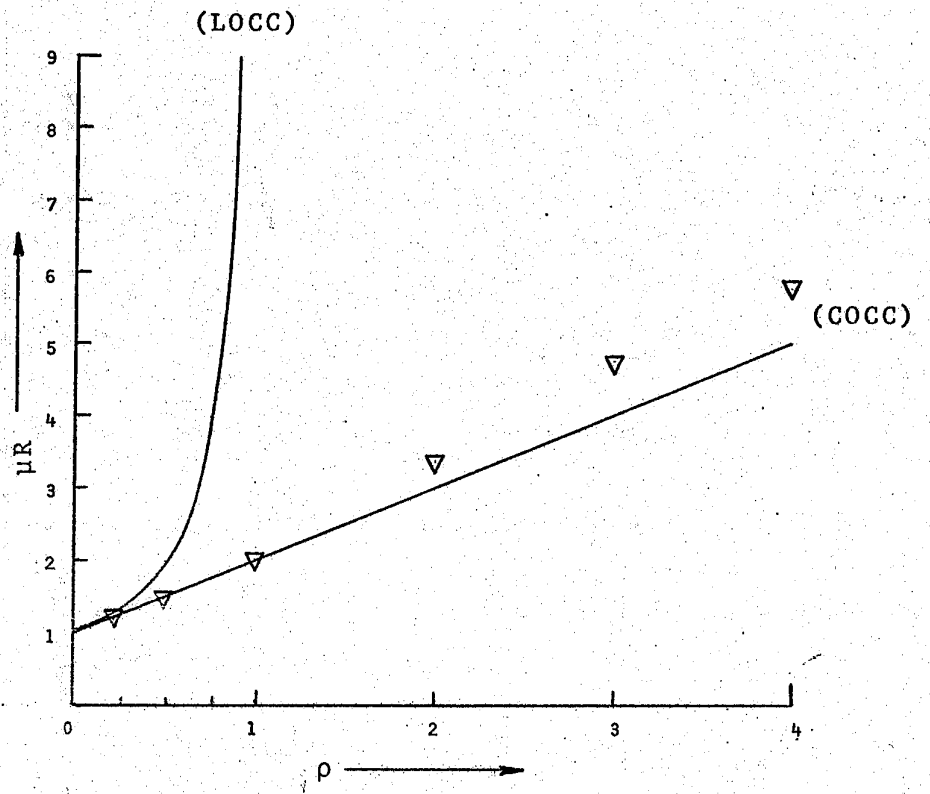

Figure 3 - Normalized Response Time for Single Resource. —— analytic results ▽ simulation results

3.2. Analysis for the LOCC Mechanism

The LOCC mechanism can be modeled as an M/M/1 queueing system since all transactions conflict in this single resource model. Thus, the average response time for a transaction is given by the expression [KLEI 75].

$$R = \frac{1/\mu}{1 - \rho}$$

Figure 3 illustrates the normalized average response time, μR, as a function of ρ for two types of concurrency control mechanisms considered in this paper. It can be seen from the single resource model that conflict oriented concurrency control performs better than locking oriented concurrency control. Next section models the database as a collection of resources.

4. THE MULTIPLE RESOURCE MODEL

This section considers the database as a collection of M resources. Transactions are assumed to update n resources chosen with equal probability out of the M resources which form the database. The execution time of a transaction is again exponentially distributed with average equal to n/μ. Transactions which update more resources take longer to execute.

The results obtained for the COCC case are the following.

Result 4: The probability of success, p, is given by the expression

$$p = \frac{1}{1 + n(1-NC)\rho}$$

where

$$NC = \binom{M-n}{n} \Big/ \binom{M}{n}$$

and

$$\rho = \lambda/\mu.$$

Table 2 shows several values of p as a function of ρ for the analytic model and for the simulation model.

p \ ρ	0.25	0.5	1.0	2.0	3.0	4.0	5.0
Analytic Model	0.87	0.77	0.62	0.45	0.36	0.29	0.25
Simulation Model	0.87	0.77	0.61	0.43	0.33	0.27	0.22

Table 2 - Probability of Success for M = 200 and n = 5

Figure 4 shows the probability of success, p, as a function of ρ for several values of n.

From result 3 we have that the normalized average response time μR is given by

$$\mu R = n[1+n(1-NC)\rho]$$

and is plotted in figure 5 as a function of ρ for several values of n.

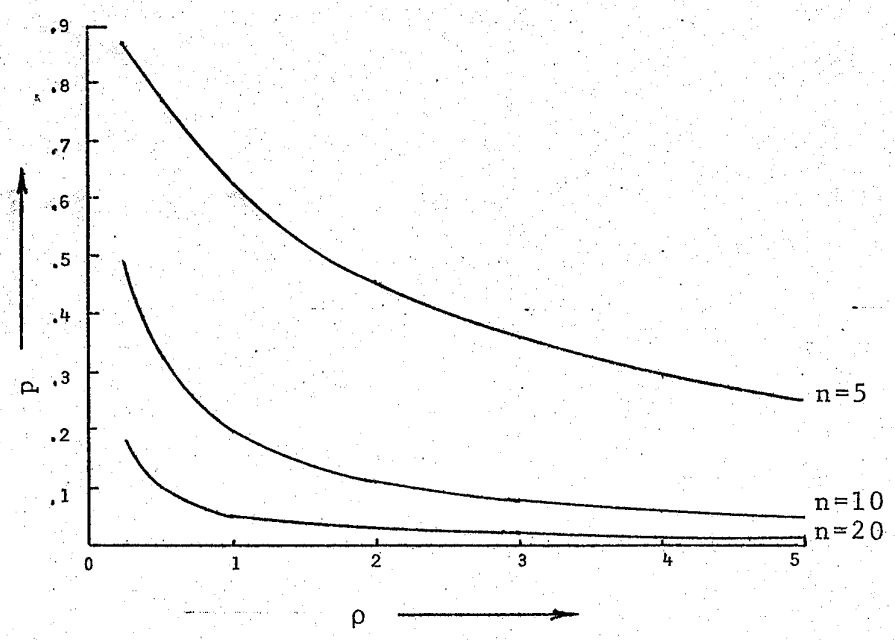

Figure 4 - Probability of Success (COCC, M = 200)

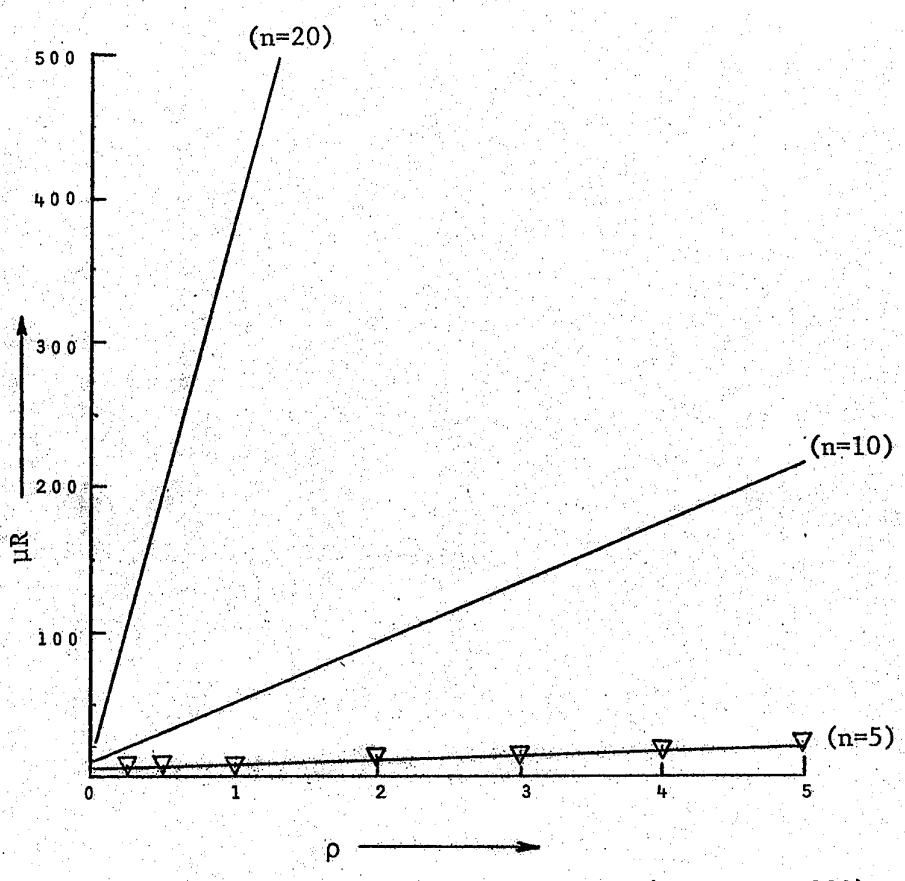

Figure 5 - Normalized Average Response Time (COCC, M = 200)
—— analytic results ▽ simulation results

In order to compare the performance of the COCC mechanism with that of the LOCC one, we built a simulation model for the locking oriented mechanism. Figure 6 shows the normalized average response time μR as a function of ρ for the two types of concurrency control mechanisms. It can be seen from the figure that the normalized average response time for the COCC mechanism is always better than the LOCC one.

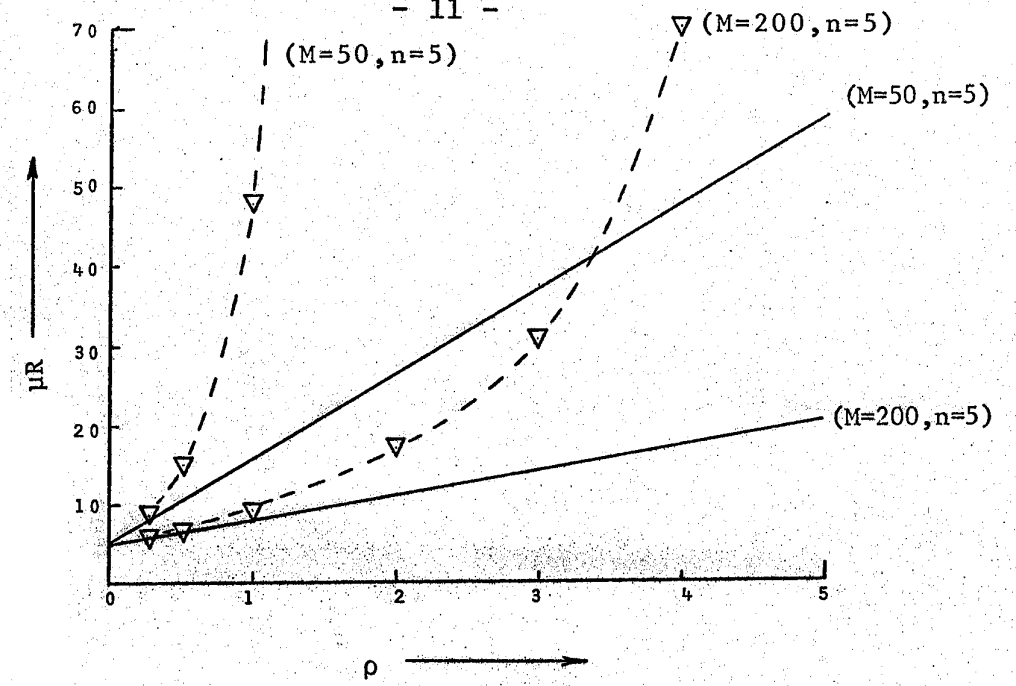

Figure 6 - Nomalized Average Response Time for Multiple Resource Model
—— analytic results COCC; ∇ simulation results LOCC

CONCLUSION

Two types of concurrency control mechanisms for database management systems have been considered in this paper: optimistic and pessimistic ones. In order to analyze the response of transactions under both mechanisms, analytic and simulation models have been developed. The first model, considers the database as being a single resource. The results of this model indicate that the normalized average response time for the class of optimistic mechanisms is always better than that for the class of pessimistic mechanisms. The second and more general model considers the database as a collection of resources. The results in this case confirm the ones obtained from the simple model (the single resource one), i.e., the average response time of transactions under optimistic concurrency control mechanisms is always better than that under pessimistic concurrency control mechanisms. All the analytic models given in the paper have been validated through simulation.

The problem considered in this paper is being analyzed in the context of a distributed database management system and will be subject of a future report.

APPENDIX

This appendix contains the proof for all results in the paper.

Proof for Result 1: Let us consider a flagged transaction with execution time equal to T seconds. Let t_0 be the arrival instant of the transaction. Hence, $t_0 + T$ is the instant at which the transaction completes its execution. Let us divide the interval $(t_0, t_0 + T)$ into m sub-intervals of length $\Delta t = T/m$.

The probability of success p can be calculated as follows. Let

$$y = P_r[\text{zero successful completions in } (t_0, t_0 + T)] \quad (A.1)$$

Let us first calculate the probability $p(t,\Delta t)$ of zero sucessful completions in the interval $(t, t+\Delta t)$ for small Δt. In order to find this probability we will first condition on the number, k, of transactions in progress at time t. $P_k(t)$ is the probability that k transactions are in progress at time t. Therefore

$$p(t,\Delta t|k) = \sum_{i=0}^{k} \binom{k}{i} (1-\mu\Delta t)^i [\mu\Delta t(1-p)]^{k-i}$$

$$= 1 - \mu K_p \Delta t + 0(\Delta t) \quad (A.2)$$

In the summation above, $1-\mu\Delta t$ is the probability that a transaction does not complete in Δt and $\mu\Delta t(1-p)$ is the probability that it completes but it is rejected. Let us now uncondition on k,

$$p(t,\Delta t) = \sum_{k=0}^{\infty} p(t,\Delta t|k)P_k(t)$$

$$= \sum_{k=0}^{\infty} (1-\mu kp\Delta t)P_k(t)$$

$$= 1 - \mu p\Delta t \sum_{k=0}^{\infty} kP_k(t)$$

$$= 1 - \mu p\Delta t \left.\frac{\partial P(z,t)}{\partial z}\right|_{z=1} \qquad (A.3)$$

where $P(z,t) \triangleq \sum_{k=0}^{\infty} z^k P_k(t)$.

The number of transactions in the system can be modeled as the Markov chain shown in figure 7

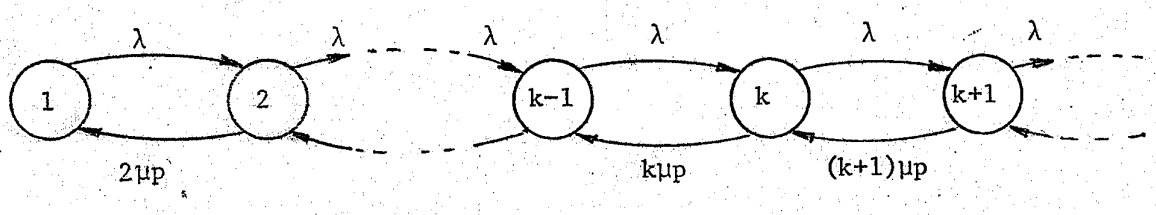

Figure 7 - Markov Chain for the Number of in-Progress Transactions

The expression for P(z,t) is the same as the one for an M/M/∞ queueing system with average service rate equal to μp. From [KLEI 76] we have that

$$P(z,t) = \exp\left[\frac{\lambda}{\mu p}(1 - e^{-\mu pt})(z-1)\right] \qquad (A.4)$$

Therefore,

$$\left.\frac{\partial P(z,t)}{\partial z}\right|_{z=1} = \frac{\lambda}{\mu p}\left(1 - e^{-\mu p t}\right) \qquad (A.5)$$

So,

$$y = \prod_{i=0}^{m-1} p\left(t_0 + \frac{iT}{m}, \frac{T}{m}\right)$$

$$= \prod_{i=0}^{m-1} \left\{1 - \frac{\lambda T}{m}\left[1 - e^{-\mu p\left(t_0 + \frac{iT}{m}\right)}\right]\right\}$$

Taking the natural logarithm of the above expression, we get,

$$\log y = \sum_{i=0}^{m-1} \log\left[1 - \frac{\lambda T}{m} + \frac{\lambda T}{m}\, e^{-\mu p\left(t_0 + \frac{iT}{m}\right)}\right]$$

Taking the limit as $\Delta t \to 0$, or equivalently as $m \to \infty$ we have that

$$\lim_{m\to\infty} (\log y) = \lim_{m\to\infty} \sum_{i=0}^{m-1} \left[e^{-\mu p t_0 - \mu p \frac{iT}{m}} - 1\right] \frac{\lambda T}{m}$$

In the limit, as $m \to \infty$, the summation can be transformed into an integral in $x = \frac{iT}{m}$, where $dx = T/m$, as shown below

$$\lim_{m\to\infty}(\log y) = \int_{x=0}^{T} \lambda e^{-\mu p t_0 - \mu p x}\, dx - \int_{x=0}^{T} \lambda dx$$

$$= -\frac{\lambda e^{-\mu p t_0}}{\mu p}\left(e^{-\mu p T} - 1\right) - \lambda T \qquad (A.6)$$

Since we are interested in the equilibrium probability, p, we must take the limit of the expression given by (A.6) as $t_0 \to \infty$. Thus,

$$\lim_{m\to\infty, t_0\to\infty} (\log y) = -\lambda T$$

Hence, $y = e^{-\lambda T}$ (A.7)

Now, p can be easily found by unconditioning on the time T a transaction takes to execute

$$p = \int_{T=0}^{\infty} e^{-\lambda T} \mu e^{-\mu T} dT = \frac{\mu}{\lambda+\mu}$$

Therefore,

$$p = \frac{1}{1+\rho} \quad \text{where} \quad \rho = \frac{\lambda}{\mu}$$

Proof for result 2: $R^*(s)$ can be obtained by first conditioning on the number, n, of times a transaction must be executed. The probability that this number is equal to n is $(1-p)^{n-1}p$.
Hence

$$R^*(s) = \sum_{n=1}^{\infty} [B^*(s)]^n (1-p)^{n-1} p$$

$$= \frac{p\, B^*(s)}{1-(1-p)B^*(s)}$$

Proof for Result 3: This result can be immediately obtained by substituting $B^*(s) = \mu/(s+\mu)$ into the expression given in Result 2 for $R^*(s)$.

The result is

$$R^*(s) = \frac{\mu p}{s+\mu p} ,$$

which can inverted to give

$$r(t) = \mu p e^{-\mu p t} \quad \text{for } t \geq 0$$

Proof for Result 4: This proof closely parallels the proof for result 1, therefore we will leave out some intermediate steps.

We start by obtaining the probability

$$p(t,\Delta t) = P_r[\text{no transaction which conflicts with our flagged transaction updates the DB in the interval } (t,t+\Delta t)]$$

Let NC be the probability that the flagged transaction does not conflict with another transaction. This probability is given by the expression below.

$$NC = \frac{\binom{M-n}{n}}{\binom{M}{n}}$$

Let x(k) be the probability that the flagged transaction conflicts with k transactions out of a group of G transactions. Then,

$$x(k) = \binom{G}{k} (1-NC)^k NC^{G-k}$$

Then, the probability w that none of the k conflicting transactions update the database in $(t,t+\Delta t)$ is equal to

$$w = \sum_{i=0}^{k} \binom{k}{i} (1 - \frac{\mu}{n} \Delta t)^i \left[\frac{\mu}{n} \Delta t(1-p)\right]^{k-i}$$

$$= (1 - \frac{\mu}{n} p\ dt)^k$$

We are now in a position to calculate $p(t,\Delta t)$

$$p(t,\Delta t) = \sum_{G=0}^{\infty} \sum_{k=0}^{G} w.x(k) P_G(t)$$

where $P_G(t) = P_r$[there are G transactions in the system]

$$p(t,\Delta t) = \sum_{G=0}^{\infty} \sum_{k=0}^{G} (1 - \frac{\mu}{n} p\Delta t)^k \binom{G}{k} (1-NC)^k NC^{G-k} P_G(t)$$

$$= \sum_{G=0}^{\infty} P_G(t) \left[1 - \frac{\mu G}{n} p\Delta t(1-NC)\right]$$

$$= 1 - \frac{\mu}{n} p\Delta t(1-NC) \frac{\partial}{\partial z} P(z,t)\Big|_{z=1}$$

The value of $\frac{\partial}{\partial z} P(z,t)$ can be obtained in a similar way as in result 1 where μ is replaced by μ/n. Hence,

$$\frac{\partial}{\partial z} P(z,t)\Big|_{z=1} = \frac{n\lambda}{\mu p} (1 - e^{-\frac{\mu p t}{n}})$$

Similarly to the proof of result 1, we divide the execution time, T, of the flagged transaction into m subintervals of length T/m. Then, we calculate the probability

$$y = \Pr[\text{no conflicting transaction updates the DB in } (t, t+T) \mid T]$$

$$= \prod_{i=0}^{m-1} \{1 - \frac{\lambda T}{m} (1-NC) [1 - e^{-\frac{\mu p}{n}(t_0 + \frac{iT}{m})}]\}$$

Taking the natural logarithm of y and taking the limit as $\Delta t \to 0$ $(m \to \infty)$ and then as $t_0 \to \infty$ we get

$$y = e^{-\lambda(1-NC)T}$$

Unconditioning on T we have

$$p = \int_{T=0}^{\infty} y \frac{\mu}{n} e^{-\frac{\mu T}{n}} dT$$

$$= \frac{1}{1 + n(1-NC)\rho}$$

REFERENCES

BADAL 79 Badal, D.Z., "Correctness of Concurrency Control and Implications in Distributed Databases", Proc. 1979 IEEE COMPSAC, Chicago, November 1979.

ESWA 76 Eswaran, K.P., J.N. Gray, R.A. Lorie and I.L. Traiger, "The Notions of Consistency and Predicate Locks in a Database System", CACM, Nov. 1976.

GRAY 78 Gray, J.N., "Notes on Database Operating Systems," Operating Systems: an Advanced Course, Springer-Verlag Berlin Heildberg 1978, pp. 394-481.

KUNG 79 Kung, H.T. and J. Robinson, "On optimistic Methods for Concurrency Control", Proc. VLDB 1979, October 1979, Rio de Janeiro.

THOM 79 Thomas, R.H., "A Majority Consensus Approach to Concurrency Control", ACM TODS, Vol. 4, June 2, 1979.