The Effect of Clock Drifts on the Performance of Distributed Timestamp Ordering *

C. J. Bouras **P. G. Spirakis**

Department of Computer Science and Engineering
University of Patras, Greece
and Computer Technology Institute, Greece

Abstract

Timestamp–based algorithms have been proposed to protect distributed database from inconsistencies during concurrent access. In such algorithms, transactions may reach a paricular site out of the order of their timestamps, due to unexpected network delays. This causes conflicts which the distributed concurrency control mechanism has to cope with. In this paper we analyze the essential features of the conflict phenomenon under the realistic assumption of the local site clocks out of total synchrony. We quantify the way in which local clock inaccuracies affect the phenomenon of transaction conflicts. In particular, we express the probability of order–reverse as a particular increasing function of the maximum clock drift.

Keywords : *Concurrency control, conflicts, performance, timestamp, clock drifts, asynchrony, distributed databases*

This research was partially supported by the Ministry of Education of Greece and by the EEC RACE Project TELEMED

1 Introduction

Database concurrency control is concerned with the problems that arise when several users access and update a database simultaneously. Concurrency control algorithms try to maintain the consistency of the database. Since the semantics of transactions are embedded in application programs, it is not easy to design concurrency control algorithms that take advantage knowledge. Most techniques therefore try to preserve (syntactic) serializability if it is produces the same effect on the database as a serial execution of those same transactions. Since a serial execution preserves consistency, a serializable execution also preserves it.

Research in the area of concurrency control for distributed database systems has led to the development of many concurrency control algorithms. Most of these algorithms are based on one of three basic mechanisms : locking, timestamps and optimistic concurrency [BG81]. Given the ever–growing number of available concurrency control algorithms, considerable research has recently been devoted to evaluating the performance of concurrency control algorithms. The behavior of locking has been investigated using both simulation and analytical models [ACL87], [BBD82], [Tay84].

A qualitative study that discussed performance issues for a number of distributed locking and timestamp algorithms was presented in [BG81] and an empirical comparison of several concurrency control schemes was given in [PR83].

Recently, the performance of different concurrency algorithms has been compared in a number of studies. The performance of locking was compared with the performance of basic timestamp ordering [Gal84]. Results of experiments comparing locking to the optimistic method appeared in [Rob82] and the performance of several variants of locking, basic timestamp ordering, and the optimistic method was compared in [ACL87], [LN89]. These performance studies are informative, but the results that have emerged, instead of being definitive, have been very contradictory. This happens because of the different set of assumptions about the models [ACL87].

Although performance models of timestamp–ordering concurrency control were extensively studied [AR87], [BGP84], [BS90], [CGM88], [KBL82], [KKM81], [Li87], [LN89], [SA86], [SG87], [WL86], all models up to now assume global time for timestamping.In contrast real physically distributed database systems use local clocks which do not indicate the same time. However, aside from relatively considerations, it usually holds that there is some bounded proportion between elapsed

local time spans [Vit84]. Techniques such as message passing [Lan78] can be used to keep local clocks almost synchronized [LL84]. In this paper we do assume that local clocks suffer a small bounded drift [KTZ88].

Since network delays relate to the actual transaction origination times but the receiving site associates transaction arrivals to their (logical) timestamps, one has to relate logical and global time in order to quantify the effect of non- synchronized local clocks on the possibility of transaction conflicts noticed by the concurrency control algorithm. In this paper, we analyze the conflicts between transactions. We focus on the few important factors affecting conflicts and develop a simple mathematical model to access their relative significance. Our approach is simpler and slightly more general since it incorporates the effect of clock drifts into the analysis for the first time.

2 Basic Architecture of Distributed Databases

A database is a collection of shared data items. In a database, certain relationships hold among its data items. The set of these relationships for a database is called the consistency assertions of the database. A database is in consistent state if the current values of its data items satisfy all of its consistency assertions. In such a system, a transaction is a program with read, write and other operations. A Distributed Database Managment System($DDBMS$) may be viewed as a collection of nodes connected to a network. Each node consists of a computer running either a Transaction Manager(TM) or a Data Manager(DM) or both. The nodes communicate by sending messages over the network. The network is assumed to be completely reliable, i.e. , if node A sends a message to node B, it is guaranteed that B will receive the message error–free. The architecture of the system is shown in Fig. 1. A TM coordinates the execution of the transaction. A DM manages a local database. From the viewpoint of a single transaction, the system consists of a single TM and a number of DMs, Fig. 1. Neither TMs nor DMs intercommunicate.

There are several reasons why $DDBMS$ are developed. Many organizations are decentralized and a $DDBMS$ approach fits more naturally the structure of the organization. $DDBMS$ are the natural solution when several databases already exit in an organization and the necessity of performing global applications arises. The recent development of small computers, providing (at a lower cost) many of the capabilities which were previously provided by large mainframes, constitutes

Figure 1: Architecture of the $DDBMS$ model

Figure 2: $DDBMS$ model for a single transaction

the necessary hardware support for the development of distributed information systems. The technology of $DDBMS$ is based on two other technologies which have developed a sufficiently solid foundation during the last years, computer networks technology and database technology. More details about $DDBMS$ can be found in [CP84].

3 The Model

We assume that the distributed database consists of S sites. The database is not replicated . This mean that each data object exists at only one site. So, in each site there exists a different Local Data Base. The number of data objects per site (database size) is N. A perfectly reliable network is assumed to connect the S sites. A key parameter in our model is the end–to–end delay which is the ellapsed time from the sending of a transaction at its source to the delivery of the transaction at its destination. As [Li87] points out, the distribution of the end–to–end delay is not mathematically tractable in general due to strong dependencies between the interarrival and service sequences at each channel and between the service times at successive communication channels. We require the delay distribution as an input to our model.

Transactions are generated at different sites as independent Poisson processes. We assume that local processing times are negligible compared to communication delays. We also assume that transaction generations and communication delays are statistically independent. Each transaction is assumed to access M data objects (and we assume $M << N$), which belong in the same Local Data Base . Transactions travel across the network as message packets of reads and writes (one such packet per transaction). The data objects accessed by each transaction are equilprobably selected among the N data objects (uniform access).

The clocks at differents sites are not perfectly synchronized. Instead we assume an ϵ–bounded drift [KTZ88] among the clocks. More specifically, if t is the global time and $LC(j,t)$ the indicatin of the clock of site j at t , then there is an $\epsilon > 0$ such that for all j:

$$|LC(j,t) - t| < \epsilon \tag{1}$$

It is obvious that the unique timestamp which receives each transaction is $LC(j,t)$. Furthermore, the values of $LC(j,t)$ are assumed to be uniformly and independently distributed in $[t \pm \epsilon]$. The constant ϵ is known from the specification of the underlying hardware clocks. Typically ϵ is very small, in the order of 10^{-5} to 10^{-6}.

Note that only perfectly synchronized clocks were considered by the research on the performance of timestamps algorithms up to now.

4 The Essential Factors of Conflicts

4.1 Causes

It is known [BG81] that transaction conflicts (i.e. violations of the serializable processing) arise due to the following two events in conjunction.

Event 1 Two or more transactions may access the same data object and at least one of them is an attempt to write , and

Event 2 Intersite communication delays may cause transactions originating from different sites to arrive at one site in reverse order of their timestamps.

Any timestamp ordering algorithm must either reorder the requests or reject some transactions in order to preserve serializability.

4.2 The issue of pipelined delays

The requests originating from the same site are assumed to be pipelined i.e. they will arrive at the same site in the order of their timestamps. This can be ensured by communication network protocols. The pipelined transmission requirement introduces resequencing delays. The distribution of such delays was analyzed in [AR87],[BGP84],[KBL82],[KKM81],[SG87] , mostly by modelling the network connecting two sites as an M/M/∞, an M/G/∞ or an M/M/k queueing system. Even in such a case, the PDF of the pipelined transmission delay does not have a convenient form. In our analysis, the PDF of this delay is an input parameter.

4.3 Static Conflicts

It is obvious that events 1, 2 of the conflict phenomenon are independent. Event 1 is a prerequisite for conflict. Without it event 2 would not matter. The percentage of potential conflicts (static conflicts) due to event 1 can be estimated by

combinatorial techniques based on assumptions about the creation of readsets and writesets of transactions.

Let each transaction access M data objects, out of N, selected uniformly. Then the probability of two transactions having at least one common data object is

$$p_{cd} = 1 - \frac{\left(\begin{array}{c} N-M \\ M \end{array} \right)}{\left(\begin{array}{c} N \\ M \end{array} \right)} \tag{2}$$

If we assume that $M \ll N$ (which is usualaly the case in practice), then the above equation gives

$$p_{cd} = \frac{M^2}{N} \tag{3}$$

Furthermore, if RO is the fraction of READ operations per transaction and WO the corresponding fraction of WRITE operations ($0 < RO, WO$ and $RO + WO = 1$) then the probability of a static conflict p_{sc} between two particular transactions is calculated as follows:

The probability that two particular transactions have $x > 0$ item in common is

$$p_{cd}^x = \frac{\left(\begin{array}{c} N-M \\ M-x \end{array} \right) \left(\begin{array}{c} M \\ x \end{array} \right)}{\left(\begin{array}{c} N \\ M \end{array} \right)} \tag{4}$$

then the conditional probability of a static conflict provided that x items are in common is

$$p_c^x = (1 - RO^x)^2 \tag{5}$$

and finally the probability of a static conflict is

$$p_{sc} = \sum_{x=1}^{M} p_{cd}^x p_c^x \tag{6}$$

Note that in (5) we have ignored Thomas's Rule [Tho79]. Note also that more complicated assumptions about the way to select readsets and writesets can be handled by modifying the combinatorics.

Figure 3: Local ordering of events

4.4 The Phenomenon of Order Reverses and its Probability

A detailed modeling of this phenomenon would have to take into account the local time intervals shown in Fig. 3. Times x, y, z are insignificant compared to network delays. Thus, a simplified look to the phenomenon of order reverses may assume that all those times are zero.

Let us consider a particular transaction T_1 generated at site i which is then transmitted to site k. Denote by a_{ij} the time we have to wait until we see the next generation of a transaction T_2 (say at site j). Due to the memoryless property of Poisson processes, a_{ij}, is exponentially distributed with mean $1/\lambda_k$ (T_1 and T_2 go to site k).Let t_1 and t_2 be the actual generation times of T_1, T_2 and $LC(i, t_i)$, $LC(j, t_2)$ the corresponding timestamps of T_1 and T_2. Clearly $a_{ij} = t_2 - t_1$. In the sequel we assume $t_1 < t_2$. If d_{ik} and d_{jk} denote the network delays (transmission plus pipeling) for T_1 and T_2 to go to site k, then an order reverse will be found by site k if and only if one of the following two sets of inequalities hold:

Either

$$LC(i, t_1) < LC(i, t_2) \ and \ t_1 + d_{ik} > t_2 + d_{jk}$$

or

$$LC(i, t_1) > LC(i, t_2) \ and \ t_1 + d_{ik} < t_2 + d_{jk} \tag{7}$$

Let

Event E_1 be the inequality $LC(i, t_1) < LC(j, t_2)$
Event E_2 be the inequality $t_1 + d_{ik} > t_2 + d_{jk}$

Event G_1 be the inequality $LC(i,t_1) > LC(j,t_2)$
Event G_2 be the inequality $t_1 + d_{ik} < t_2 + d_{jk}$

We define event E as

$$E = (E_1 \wedge E_2) \vee (G_1 \wedge G_2) \tag{8}$$

Note that all literature up to now considered event E_1 just to be $t_1 < t_2$ (thus ignoring the clock synchronization issue).

Also note that event E_2 can be rewritten as

$$d_{ik} > d_{jk} + a_{ij} \tag{9}$$

In the sequel we assume that d_{ik} and d_{jk} are exponentially distributed with means μ_{ik} and μ_{jk}.

The probability of order reverse from (8) is then

$$R_{ij}^k = Prob\{E\} \tag{10}$$

Since events E_i exclude the $G_i, (i = 1, 2)$ and since E_1 is independent of E_2 (also G_1 independent of G_2) we get

Lemma 1

$$R_{ij}^k = Prob\{E_1\} * Prob\{E_2\} + Prob\{G_1\} * Prob\{G_2\} \tag{11}$$

Note that assumption of total synchrony of local clocks would simplify R_{ij}^k to just be $Prob\{E_2\}$.

Also , due to our assumptions , we have

$$Prob\{G_i\} = 1 - Prob\{E_i\}, i = 1, 2$$

Thus

$$R_{ij}^k = Prob\{E_1\} * Prob\{E_2\} + (1 - Prob\{E_1\}) * (1 - Prob\{E_2\}) \tag{12}$$

Figure 4: Case 1

4.5 Estimation of the probability of order reverse

The following two facts are true about exponentials

Fact 1 Given two independent exponential random variables x_1 and x_2 with rates $\frac{1}{\mu_1}$, $\frac{1}{\mu_2}$ we have

$$Prob\{x_1 \geq x_2\} = \frac{\mu_2}{\mu_1 + \mu_2}$$

Fact 2 Given independent random variables x, y, z such that x is exponential and y is positive, we have

$$Prob\{x > y + z\} = Prob\{x > y\}Prob\{x > z\}$$

Thus, as in [Li87] we get

$$\begin{aligned} Prob\{E_2\} &= \\ &= Prob\{a_{ij} < d_{ik}\}Prob\{d_{jk} < d_{ik}\} \\ &= \frac{\lambda_j}{\mu_{ik} + \lambda_j} \frac{\mu_{jk}}{\mu_{ik} + \mu_{jk}} \end{aligned} \tag{13}$$

As far as $Prob\{E_1\}$ is concerned we have the following two Cases:

Case 1 (Fig. 4)

If $t_1 + \epsilon \leq t_2 - \epsilon$, which means that $a_{ij} = t_2 - t_1 \geq 2\epsilon$, then event E_1 holds with conditional probability 1

Figure 5: Case 2

In this case

$$\begin{aligned} &Prob\{event\ E_1\ in\ Case\ 1\} = \\ &= \ Prob\{LC(i,t_1) \leq LC(j,t_2)\} Prob\{event\ E_1\ given\ Case\ 1\} \\ &= \ Prob\{t_2 - t_1 \geq 2\epsilon\} * 1 \\ &= \ e^{-\lambda_j 2\epsilon} \end{aligned} \tag{14}$$

Case 2 (Fig. 5)

If $t_1 + \epsilon > t_2 - \epsilon$ then $2\epsilon > a_{ij} = t_2 - t_1$

In this case

$$\begin{aligned} &Prob\{event\ E_1\ in\ Case\ 2\} = \\ &= \ Prob\{t_2 - t_1 < 2\epsilon\} Prob\{event\ E_1\ given\ Case\ 2\} \end{aligned} \tag{15}$$

After some algebra (Appendix 1) we get

$$\begin{aligned} &Prob\{event\ E_1\ in\ Case\ 2\} = \\ &= \ \frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2} \left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2 \right) e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2} \right) \end{aligned} \tag{16}$$

Thus, finally from Cases 1 and 2 we have that,

$$\begin{aligned} &Prob\{E_1\} = \\ &= \ \left(e^{-\lambda_j 2\epsilon} + \frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2} \left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2 \right) e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2} \right) \right) \end{aligned} \tag{17}$$

It is obvious that

$$\lim_{\epsilon \to 0} Prob\{E_1\} = 1$$

This result agrees with all previous works [BS90]. However the clock phenomenon is not taken into account. So from (14), (17) and Lemma 1 we have that

$$
\begin{aligned}
R_{ij}^k = & \\
= & \left[e^{-\lambda_j 2\epsilon} + \frac{1-e^{-\lambda_j 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right)e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)\right]\left(\frac{\lambda_j}{\lambda_j+\mu_{ik}}\frac{\mu_{jk}}{\mu_{ik}+\mu_{jk}}\right) + \\
& \left[1 - \left(e^{-\lambda_j 2\epsilon} + \frac{1-e^{-\lambda_j 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right)e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)\right)\right]\left(1 - \left(\frac{\lambda_j}{\lambda_j+\mu_{ik}}\frac{\mu_{jk}}{\mu_{ik}+\mu_{jk}}\right)\right)
\end{aligned}
\tag{18}
$$

Note that (18) express R_{ij}^k as a function of λ_j, μ_{ik},μ_{jk} and ϵ only.If we assume that transactions are generated at any sites as independent Poisson process of rate λ and the network delay for each transaction is exponentially distributed of rate μ, then

$$
\begin{aligned}
R_{ij}^k = & \\
= & \left[e^{-\lambda 2\epsilon} + \frac{1-e^{-\lambda 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda} + \left(\frac{1}{\lambda^2} - 4\epsilon^2\right)e^{-\lambda 2\epsilon} - \frac{1}{\lambda^2}\right)\right]\left(\frac{\lambda}{2(\lambda+\mu)}\right) + \\
& \left[1 - \left(e^{-\lambda 2\epsilon} + \frac{1-e^{-\lambda 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda} + \left(\frac{1}{\lambda^2} - 4\epsilon^2\right)e^{-\lambda 2\epsilon} - \frac{1}{\lambda^2}\right)\right)\right]\left(1 - \frac{\lambda}{2(\lambda+\mu)}\right)
\end{aligned}
\tag{19}
$$

The above probability under our assumotions is the same for each pair of transactions at any site.From now on we will call it R for simplicity.

It is easy to see that

$$\lim_{\epsilon\to 0} R = \frac{\lambda}{2(\lambda+\mu)} \tag{20}$$

Also (Appendix 2)

$$\lim_{\epsilon\to\infty} R = \lim_{\epsilon\to\infty} Prob\{E_1\} = \frac{1}{2} \tag{21}$$

The curves in Fig. 6 show how R varies as a function of ϵ. We note that the probability of order–reverse is an increasing function of ϵ. So, that the overhead to keep site clocks almost synchronized should be traded–off with the increased percentage of transaction restarts(or delays) caused by clock drifts.

For small ϵ, which is usually the case in practice (Appendix 3), (18) gives that

$$R_{apr} = (1 - 3\lambda\epsilon + 4\epsilon^2\lambda^2)\frac{\lambda}{2(\mu+\lambda)} + (3\lambda\epsilon - 4\epsilon^2\lambda^2)\frac{2\mu+\lambda}{2(\mu+\lambda)} \tag{22}$$

The following table gives some numerical results, for $\lambda = 1$ and $\epsilon = 10^{-2}$.

μ	R	R_{apr}	$Prob\{E_2\}$
2.00000E+00	1.79651E-01	1.86400E-01	1.66667E-01
1.00000E+00	2.59738E-01	2.64800E-01	2.50000E-01
6.66667E-01	3.07791E-01	3.11840E-01	3.00000E-01
5.00000E-01	3.39826E-01	3.43200E-01	3.33333E-01
4.00000E-01	3.62708E-01	3.65600E-01	3.57143E-01
3.33333E-01	3.79869E-01	3.82400E-01	3.75000E-01
2.85714E-01	3.93217E-01	3.95467E-01	3.88889E-01
2.50000E-01	4.03895E-01	4.05920E-01	4.00000E-01
2.22222E-01	4.12632E-01	4.14473E-01	4.09091E-01
2.00000E-01	4.19913E-01	4.21600E-01	4.16667E-01

5 The Probability of Conflict

The probability that a transaction is rejected if only three sites are present, is clearly from (6) and (19).

$$\begin{aligned} r &= Prob\{of\ conflict\ in\ case\ of\ three\ sites\} \\ &= Prob\{of\ static\ conflict\}Prob\{out\ of\ order\} \\ &= p_{sc} * R \end{aligned} \tag{23}$$

6 Conclusions and Future Work

We have shown that the probability of order–reverses is an increasing function of the clock drift parameter ϵ and the network delay $1/\mu$. The phenomenon of order–reverses is the main cause of either delays or restarts in any timestamp–ordering based concurrency control algorithm. Therefore, the additional effort needed in order to keep local clocks almost synchronized is well justified since it improves the overall performance of the scheduler.

Our analysis can be genaralized to take into account non–exponential delays and non–uniform drifts. In fact, one can superimpose a distribution on ϵ, making it a random variable and thus parametrizing the quality of clock syncronization protocols. The effect of the degree of asynchrony on DDB protocol performance seems to be an important topic for future research. In this setting, protocols such as two–phase locking, that seem immune to clock drifts, are of particular interest.

Figure 6: R as a function of ϵ, for different λ and μ

REFERENCES

[ACL87] Agrawal R., Carey M., Linvy M., "Concurrency Control Performance modelling: Alternatives and Implications", ACM Transactions on Database Systems, Vol. 12, No 4, Dec. 1987, pp. 609-654

[AR87] Agrawal S., Ramaswamy R., "Analysis of the Resequencing Delay for M/M/∞ systems", ACM SIGMETRICS 1987, pp. 27-35

[BGP84] Bacelli F., Gelenbe E., Plateau B., "An end to end approach to the resequencing problem", JACM Vol. 31, No 3, July 1984

[BBD82] Balter R., Berard P., Decitre P., "Why control of the concurrency level in distributed systems is more fundamental than deadlock management", Proceedings of the 1st ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, Ottawa, Ontario, Aug. 18-20, 1982, pp. 183-193

[BG81] Bernstein A., Goodman M., "Concurrency Control in Distributed Database Systems", Computing Survey, Vol. 13, No. 2, June 1981

[BS90] Bouras C., Spirakis P., "Simplified Performance Models of the Reordering Issue in Timestamp Ordering Concurrency Control in Distributed Databases", ISCIS V, Cappadocia, Turkey, 1990

[CS84] Carey M., Stonebraker M., "The performance of concurrency control algorithns for database management systems", Proceedings of the 10th International Conference on Very Large Data Bases, Singapore, Aug 1984, pp. 107-118

[CGM88] Cellary W., Gelenbe E., Morzy T., "Concurrency Control in Distributed Databases Systems", Studies in Computer Science and Artificial Intelligence, Ed. Kobayashi H., Nivat M., North-Holland, 1988

[CP84] Ceri S., Pelagatti G., "Distributed Databases.Principles and Systems", McGraw-Hill, 1984

[Gal84] Galler B.I., "Concurrency Control Performance Issues", Report CSRG-147 Ph.D Thesis University of Toronto, Toronto Canada, 1984

[KBL82] Kamoun F., Ben Djerad M., LeLann G., "Queueing Analysis of the Ordering Issue in a Distributed Database Concurrency Control: A

General Case", Proceedings of the 3rd International Conference on Distributed Computing Systems, 1982, pp. 447-452

[KKM81] Kamoun F., Kleinrock L., Muntz R., "Queueing Analysis of the Ordering Issue in a Distributed Database Concurrency Control Mechanism", Proceedings of the 2nd International Conference on Distributed Computing Systems, 1981, pp.13-23

[KTZ88] Korach E., Tel G., Zaks S., "Optimal Synchronization of ABD Networks", Tech. Rep. RUU CS-88-23, Ultrect

[Lan78] Le Lann G., "Algorithms for Distributed Data Sharing Systems which Use Tickets",Proceedings of the 3rd Berkeley Workshop on Distributed Data Management and Computer Networks, San Francisco, 1978

[Li87] Li V., "Performance Models of Timestamp-Ordering Concurrency Control Algorithms in Distributed Databases",IEEE Transactions on Computers, Vol. C-36, No. 9, September 1987, pp. 1041-1051

[LN89] Lin W.K., Nolte J., "Distributed Database Control and Allocation", Final Tech. Rept. Vol. II, Computer Corporation of America 1989

[LL84] Lundelins J., Lynch N., "An upper and lower bound for clock synchronization",Information and Control 62(2-3) 190-204, 1984

[PR83] Peinl P., Renter A., "Empirical comparison of database concurrency control schemes", Proceedings of the 9th International Conference on Very Large Data Bases, Florence, Oct. 1983, pp. 97-108

[Ree78] Reed D., "Naming and Synchronization in a decentralized computer systems", Ph.D. Dissertation, Dept. Elec. Eng. Comp. Sc., MIT Sept. 1978

[Rob82] Robinson J., "Design of concurrency control for transactions processing systems", Ph.D. Dissertation , Dept. Comp. Sc. , Carnegie-Mellon University, Pittsburg, 1982

[Sil82] Silberschatz A., "A multiversion concurrency control scheme with no rollbacks", Proc. ACM Symposium on Principles Distributed Computing, Aug. 1982, pp. 216-223

[SA86] Singal M. , Agrawala A., "Performance Analysis of an Algorithm for Concurrency Control in Replicated Database Systems", ACM SIGMETRICS 1986, pp. 216-223

[SG87] Stafylopatis A., Gelenbe E., "Delay Analysis of Resequencing Systems with Partial Ordering", PERFORMANCE '87 , pp. 433-445

[SR81] Stearns R., Rosenkrantz D., "Distributed database concurrency control using before values", Proceedings SIGMOD Conf. Management Data, 1981, pp. 74-83

[Tay84] Tay Y.C., "A Mean Value Performance Model for Locking in Databases", Ph.D. Dissertation, Harvard University TR-04-84

[Tho79] Thomas R., "A majority consensus approach to concurrency control for multiple copy databases", ACM Tranasactions on Database Systems, Vol. 4, 1979, pp. 180-209

[Vit84] Vitanyi P., "Distributed in an Archimedean Ring of Processors", ACM STOC 1984, pp. 542-547

[WL86] Wang C., Li V., "Queueing analysis of the conservative timestamp ordering concurrency control algorithm", Proceeding IEEE International Computing Symposium 1986, pp. 1450-1455

Appendix 1

It is straightforward to observe that

$$Prob\{t_2 - t_1 < 2\epsilon\} = 1 - e^{-\lambda_j 2\epsilon}$$

Also (Fig. 7), conditioning on $t_2 - t_1 = x$, $0 \leq x \leq 2\epsilon$

$$\begin{aligned} Prob\{E_1/Case\ 2\} &= Prob\{A\}Prob\{E_1/A\} + Prob\{B\}Prob\{E_1/B\} \\ &\quad + Prob\{C\}Prob\{E_1/C\} + Prob\{D\}Prob\{E_1/D\} \end{aligned}$$

It is easy to see that

$$Prob\{E_1/A\} = Prob\{t_1+\epsilon < LC(j,t_2) < t_2+\epsilon \ and\ t_2-\epsilon < LC(i,t_1) < t_1+\epsilon\} = 1$$

Similarly

$$Prob\{E_1/B\} = 1$$
$$Prob\{E_1/C\} = \frac{1}{2}$$
$$Prob\{E_1/D\} = 1$$

In Fig. 7, the horizontal axis indicates the possible values of $LC(j, t_2)$ and the vertical the possible values of $LC(i, t_1)$.

Also, by counting areas in Fig. 7, we get that

$$Prob\{A\} = \frac{(t_2 - t_1)(t_1 - t_2 + 2\epsilon)}{4\epsilon^2}$$

$$Prob\{B\} = \frac{(t_2 - t_1)^2}{4\epsilon^2}$$
$$Prob\{C\} = \frac{(t_1 - t_2 + 2\epsilon)^2}{4\epsilon^2}$$
$$Prob\{D\} = \frac{(t_2 - t_1)(t_1 - t_2 + 2\epsilon)}{4\epsilon^2}$$

After this we get that

$$Prob\{E_1/Case\ 2\} =$$
$$= \left[(t_2 - t_1)(t_1 - t_2 + 2\epsilon) + (t_2 - t_1)^2(t_1 - t_2 + 2\epsilon)^2 + \frac{(t_1 - t_2 + 2\epsilon)^2}{2} + (t_2 - t_1)(t_1 - t_2 + 2\epsilon)\right] \frac{1}{4\epsilon^2}$$

Thus, by conditioning on $t_2 - t_1 = x$ we have that

$$Prob\{E_1 in\ Case\ 2\} = \frac{1 - e^{\lambda_j 2\epsilon}}{4\epsilon^2} \int_{x=0}^{2\epsilon} \left[2x(2\epsilon - x) + x^2 + \frac{(2\epsilon - x)^2}{2}\right] \lambda_j \epsilon^{-\lambda_j x}$$

After some calculations we get that

$$Prob\{event\ E_1\ in\ Case\ 2\} =$$
$$= \frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2} \left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right) e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)$$

Figure 7: The four subcases of Case 2

Appendix 2

First of all we shall examine, what happens for $Prob\{E_1\}$ when $\epsilon \to \infty$ We know that

$$Prob\{E_1\} = \left(e^{-\lambda_j 2\epsilon} + \frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right)e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)\right)$$

From the above equation we get that

$$lim_{\epsilon\to\infty} Prob\{E_1\} =$$
$$= \; lim_{\epsilon\to\infty}\left(e^{-\lambda_j 2\epsilon} + lim_{\epsilon\to\infty}\left[\frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right)e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)\right)\right]$$

The first term converges to 0. So

$$lim_{\epsilon\to\infty} Prob\{E_1\} =$$
$$= \; lim_{\epsilon\to\infty}\frac{1 - e^{-\lambda_j 2\epsilon}}{4\epsilon^2}\left(2\epsilon^2 + \frac{2\epsilon}{\lambda_j} + \left(\frac{1}{\lambda_j^2} - 4\epsilon^2\right)e^{-\lambda_j 2\epsilon} - \frac{1}{\lambda_j^2}\right)$$

It is obvious now that

$$\lim_{\epsilon\to\infty} Prob\{E_1\} = \frac{1}{2}$$

We have that

$$R = Prob\{E_1\} * \frac{\lambda}{2(\lambda+\mu)} + (1 - Prob\{E_1\}) * (1 - \frac{\lambda}{2(\lambda+\mu)})$$

We observe that

$$\lim_{\epsilon\to\infty} R = \lim_{\epsilon\to\infty} Prob\{E_1\} = \frac{1}{2}$$

Appendix 3

It is known that for small x

$$\epsilon^{-x} \approx 1 - x$$

So, in that case

$$Prob\{E_1\} \approx (1 - 3\lambda\epsilon + 4\lambda^2\epsilon^2)$$

Thus

$$R_{apr} = (1 - 3\lambda\epsilon + 4\epsilon^2\lambda^2)\frac{\lambda}{2(\mu+\lambda)} + (3\lambda\epsilon - 4\epsilon^2\lambda^2)\frac{2\mu+\lambda}{2(\mu+\lambda)}$$