The Perfect and Imperfect Clocks Approach to Performance Analysis of Basic Timestamp Ordering in Distributed Databases

C. J. Bouras P. G. Spirakis

Department of Computer Science and Engineering
University of Patras, Greece
and Computer Technology Institute, Greece
P.O. Box 1122, 26110 Patras, Greece

Abstract

Locking and timestamping are two popular approaches to concurrency control in databases systems. Although more than a dozen analytic performance studies of locking techniques have recently appeared in the literature, analytic performance study of timestamp–based concurrency algorithms largely remains an unexplored area. This work presents a model of a distributed database system which provides the framework to study the performance of timestamp ordering concurrency control. We exhibit an analytical solution, which has been tested with extensive simulation. The accuracy seems to be very high. We assume perfect and also imperfect clocks for synchronization and quantify the way in which local clock inaccuracies affect the phenomenon of transaction conflicts. In particular, we derive a lot of interesting performance measures such as probability of abort, throughput and others.

1 Introduction

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. Performance studies of concurrency control algorithms have been done using simulations as well as analytical methods [**?**], [**?**], [**?**], [**?**], [**?**], [**?**].

There are two critical points in analyzing the performance of timestamp ordering concurrency control algorithms in distributed database systems.

- The reordering phenomenon

  For obvious reasons, due to variable network delays, transactions do not necessarily arrive at the system sites in elaborate. From the point of view of queueing theory this phenomenon has analyzed in [**?**], [**?**], [**?**]. Our work extends these results in the field of performance analysis of concurrency control algorithms in distributed database systems.

- The effect of clock drifts

  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. The effect of clock drifts is an important point [**?**]. However, aside from relativity considerations, it usually holds that there is some bounded proportion between elapsed local time spans [**?**]. Techniques such as message passing can be used to keep local clocks almost synchronized [**?**]. In this work we do assume that local clocks suffer a small bounded drift.

This work studies the performance of Basic Timestamp Ordering(BTO) and compares the numerical results with results from simulation experiments.

2 The Model

We assume that the distributed database consists of K sites. The database is not replicated. This means 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 is N. A perfectly reliable network is assumed to connect the K 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.

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, which belong in the same Local Data Base. Each Local Data Base accepts an independent Poisson process of transactions with rate λ. 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).

In the case of clock drifts we assume an ϵ–bounded drift among the clocks. More specifically, if t is the global time and $LC(j,t)$ the indication 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 each transaction receives 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.

3 Performance Analysis

3.1 The Queueing Problem

Consider one of the sites of the $DDBMS$. It receives a sequence of transactions which affect the contents of the Local Data Base. We assume that each transaction is identified by a timestamp, and that each site of $DDBMS$ carries out the transactions in timestamp order.

Let $T_1, T_2, ...T_n$ denote a sequence of transactions which enter the system and which are directed to each of the $DDBMS$ sites via a communication network, and let $LC(i,t_n)$ denote the timestamp associated with T_n. In case of perfect clocks $LC(i,t_n)$ is equal to global time t_n, where t_n is the generation time for the transaction T_n. Each transaction T_n reaches the site where it must be executed after a communication delay y_n. Thus at the output of the communication network the transactions arrive at instants $t_n + y_n$, that do not necessarily respect the timestamp order. For this reason(the reordering phenomenon), there is a probability of abort, PA, for each transaction T_n.

So, we have the following queueinq problem:

Every transaction T_n, with generation time t_n, timestamp $LC(i,t_n)$ and network delay y_n, must finish before the transactions which will arrive after it, and there is some fixed conflict probability p_c between them. Which is the probability of abort, PA?

The transaction's conflict probability, p_c is the probability which measures the following event

- Two or more transactions may access the same data object and at least one of them is an attempt to write. Let this happen with probability p_c .

We have assumed that each transaction has a constant size M and access these M data objects, out of N, selected uniformly. Then the probability of two transactions having at least one common data object is

$$p_c = 1 - \frac{\begin{pmatrix} N-M \\ M \end{pmatrix}}{\begin{pmatrix} N \\ M \end{pmatrix}}$$

The above expression can be further simplified to:

$$p_c \approx \frac{M^2}{N} \tag{2}$$

There are two cases for analysis: the Case I, Perfect Clocks, and the Case II, Imperfect Clocks.

3.2 Case I: Perfect Clocks

Let us consider a particular transaction T generated at instant t which is then transmitted to one site. We assume that the transactions which conflict construct an independent Poisson process, with rate $\lambda_c = p_c\lambda$ [**?**]. It is obvious, that in case of perfect clocks, each transaction must be rejected from transactions which arrive later than it. So, from Fig. 1, transaction T arrives at instant t has a network delay d, must be rejected from each transaction T_i, which generated at t_i, $t < t_i$, and has a delay due to network d_i. Then an order reverse will be found between T and each other from T_i transactions if the following inequality holds:

$$t + d > t_i + d_i \tag{3}$$

Let

Event OR_i be the inequality $t + d > t_i + d_i$

Also note that event OR_i can be rewritten as

$$d > d_i + w \tag{4}$$

In the sequel we assume that d and d_i are exponentially distributed with mean μ. The factor $w = t_i - t$ is the time which we must wait until the generation of ith transaction after the transaction T. It is known [**?**] that w has a Gamma distribution with parameters i and λ_c,

$$f_w(w) = \frac{\lambda_c^i w^{i-1} e^{-\lambda_c w}}{(i-1)!}$$

Figure 1: The order reverse issue in case of Perfect Clocks

.

The probability of order reverse between transaction T and each one from T_i is then

$$\begin{aligned} p_i &= Prob\{OR_i\} \\ &= \frac{1}{2}(\frac{\lambda_c}{\mu+\lambda_c})^i \end{aligned} \quad (5)$$

It is straigthforward to observe that the probability of abort, PA, for each transaction is

$$\begin{aligned} PA &= \\ &= (1-p_1)p_1 + (1-p_1(1-p_1))p_2 + ... \\ &= \sum_{i=1}^{\infty} p_i\pi_i \end{aligned} \quad (6)$$

where

$$\pi_1 = (1-p_1)$$
$$\pi_2 = (1-p_1\pi_1)$$
$$\vdots$$
$$\pi_k = (1-p_{k-1}\pi_{k-1})$$

So, probability of not rejected is,

$$PNR = 1 - PA \quad (7)$$

Other interesting performance measures are $Throughput(THR)$

$$THR = \lambda PNR \quad (8)$$

and $AbortRatio(AR)$

$$AR = \lambda PA \quad (9)$$

3.3 Case II: Imperfect Clocks

In this case, due the clock drifts, there is the possibility that each transaction must be rejected from transactions which arrive later than it or before. Let us consider a particular transaction T generated at site j. Denote by w the time we have to wait until

Figure 2: The order reverse issue in case of Imperfect Clocks

we see the generation of i–th transaction T_i (say at site k). Let t and t_i be the actual generation times of T, T_i and $LC(j,t)$, $LC(k,t_i)$ the corresponding timestamps of T and T_i. If d and d_i denote the network delays (transmission plus pipeling) for T and T_i then an order reverse will be found if and only if one of the following two sets of inequalities hold:

Either, for future $t < t_i$ Fig. 1, T is generated earlier than T_i and T arrives later than T_i

$$LC(j,t) < LC(k,t_i) \ and \ t + d > t_i + d_i$$

or, for past $t > t_i$ Fig. 2, T is generated later than T_i, the clocks are out of order, but T arrives later than T_i

$$LC(j,t) < LC(k,t_i) \ and \ t + d > t_i + d_i \quad (10)$$

Let

Event E_1^i be the inequality $LC(j,t) < LC(k,t_i)$
Event E_2^i be the inequality $t + d > t_i + d_i$

In this case we define event of order reverse ,$OR_{i\epsilon}$, as

$$OR_{i\epsilon} = (E_1^i \wedge E_2^i) \quad (11)$$

Note that all literature up to now considered event E_1^i just to be $t < t_i$ (thus ignoring the clock synchronization issue). The probability of order reverse, in case of future, from (11) is then

$$\begin{aligned} p_{i\epsilon}^f &= Prob\{OR_{i\epsilon}\} \\ &= Prob\{E_2^i\}Prob\{E_1^i\} \\ &= Prob\{d > d_i + w\}Prob\{E_1^i\} \\ &= p_i Prob\{E_1^i\} \end{aligned} \quad (12)$$

Also, in case of past we have that

$$\begin{aligned} p_{i\epsilon}^p &= Prob\{OR_{i\epsilon}\} \\ &= p_i(1 - Prob\{E_1^i\}) \end{aligned} \quad (13)$$

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

Case A (Fig. 3)

Figure 3: Case A

Figure 4: Case B

If $t+\epsilon \leq t_i - \epsilon$, which means that $w = t_i - t \geq 2\epsilon$, then event E_1^i holds with conditional probability 1

In this case

$$Prob\{event\ E_1^i\ in\ Case\ A\} = \\ = \quad e^{-2\epsilon\lambda_c} \sum_{n=o}^{i-1} \frac{(2\epsilon\lambda_c)^n}{n!} \tag{14}$$

Case B (Fig. 4)

If $t+\epsilon > t_i - \epsilon$ then $2\epsilon > w = t_i - t$

In this case

$$Prob\{event\ E_1^i\ in\ Case\ B\} = \\ = \quad 1 - e^{-2\epsilon\lambda_c} \sum_{n=o}^{i-1} \frac{(2\epsilon\lambda_c)^n}{n!}$$

Also, Fig. 5, conditioning on $t_i - t = w$, $0 \leq w \leq 2\epsilon$

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

It is easy to see that

$$Prob\{E_1/A\} = 1$$

$$Prob\{E_1^i/B\} = 1$$

$$Prob\{E_1^i/C\} = \frac{1}{2}$$

$$Prob\{E_1^i/D\} = 1$$

Also, by counting areas in Fig. 5, we get that

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

Figure 5: The four subcases of Case B

$$Prob\{B\} = \frac{(t_i - t)^2}{4\epsilon^2}$$

$$Prob\{C\} = \frac{(t - t_i + 2\epsilon)^2}{4\epsilon^2}$$

$$Prob\{D\} = \frac{(t_i - t)(t - t_i + 2\epsilon)}{4\epsilon^2}$$

Thus, by conditioning on $t_i - t = w$ we have that

$$Prob\{E_1^i in\ Case\ B\} = \\ = \quad \frac{1-A}{8\epsilon^2} \int_{w=0}^{2\epsilon} \left[4w\epsilon - w^2 + 4\epsilon\right] \frac{\lambda_c^i w^{i-1} e^{-\lambda_c w}}{(i-1)!}\, dw$$

with

$$A = e^{-2\epsilon\lambda_c} \sum_{n=o}^{i-1} \frac{(2\epsilon\lambda_c)^n}{n!}$$

Thus, finally from Cases A and B we have that,

$$Prob\{E_1^i\} = \\ = \quad A + \frac{1-A}{8\epsilon^2} \int_{w=0}^{2\epsilon} \left[4w\epsilon - w^2 + 4\epsilon\right] \frac{\lambda_c^i w^{i-1} e^{-\lambda_c w}}{(i-1)!}\, dw \tag{15}$$

We have that

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

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

$$\lim_{i \to \infty} Prob\{E_1^i\} = 1$$

From the above, we conclude that,

$$\frac{1}{2} \leq Prob\{E_1^i\} \leq 1$$

It is easy to observe that probability of abort, PA_ϵ,in this case for each transaction is

$$\begin{aligned} PA_\epsilon &= \\ &= (1-p^f_{1\epsilon})p^f_{1\epsilon} + (1-(1-p^f_{1\epsilon})p^f_{1\epsilon})p^p_{1\epsilon} + ... \\ &= \sum_{i=1}^{\infty}(p^f_{i\epsilon}\pi_i + p^p_{i\epsilon}\pi_{i+1}) \quad i = 1,3,5... \end{aligned} \tag{16}$$

where

$$\pi_1 = (1-p^f_{1\epsilon})$$

$$\pi_2 = (1-\pi_1 p^f_{1\epsilon})$$

$$\vdots$$

$$\pi_k = (1-\pi_{k-1}p^f_{(k-1)\epsilon})$$

The probability of not rejected, PNR_ϵ, is

$$PNR_\epsilon = 1 - PA_\epsilon \tag{17}$$

Also the $Throughput(THR_\epsilon)$ is,

$$THR_\epsilon = \lambda PNR_\epsilon \tag{18}$$

and $AbortRatio(AR_\epsilon)$ is,

$$AR_\epsilon = \lambda PA_\epsilon \tag{19}$$

3.4 Numerical results and validation

In this section, we present numerical and simulation results for the performance analysis of BTO. We have compared our analytical results against the results of the simulation study to validate the accuracy of our analysis. In all cases the data base size, N, is equal to 250 and S, A means simulation and analysis respectively. In all cases we observe that:

- The analysis is very highly accurate.
- The effect of ϵ is very small for values 10^{-6}, 10^{-4}, and 10^{-2}. But there exists a significant effect in all performance measures for $\epsilon \geq 0.5$. Also the effect of ϵ is insignificant for small network service rate μ.

4 Conclusions and future work

We have presented a performance analysis of the timestamp ordering concurrency control algorithm for distributed database systems. A comparison against simulation studies shows that the proposed analytical solution has a very high accuracy for all performance measures. The analysis presented in this paper is the first in the literature, which studies the effect of imperfect clocks. 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 generalized 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.

Figure 6: BTO analytical and simulation results,$\lambda = 6$,$M = 4$

References

[1] R. Agrawal, M. Carey, M. Linvy, "Concurrency Control Performance Modelling: Alternatives and Implications", *ACM Transactions on Database Systems*, Vol. 12, No 4, pp. 609–654, 1987

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

[3] A. Bernstein, N. Goodman, "Concurrency Control in Distributed Database Systems", *Computing Survey*, Vol. 13, No. 2, 1981

[4] M. El–Toweissy, N. El–Makky, M. Abougabal, S. Fouad, "The mean value approach to performance evaluation of the time–stamp ordering algorithms", *ICCI*, pp. 276–281, 1991, Springer–Verlag Lectures Notes in Computer Science No 497

[5] W. Feller, *An Introduction to Probability Theory and its Applications*, Vol. 1–3, 2nd ed. Wiley, New York, 1971

[6] C. Fidge , "Logical Time in Distributed Computing Systems", *IEEE Computer*, pp. 28–33, 1991

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

[8] L. Lamport, "Time, Clocks, and the ordering of Events in Distributed Systems", *CACM*, Vol. 21, No. 7, 1978

[9] M. Singal, "Performance analysis of the basic timestamp ordering algorithm via Markov modelling", *Performance Evaluation*, 12, pp 17–41,1991

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

[11] A. Stafylopatis, E. Gelenbe, "Delay Analysis of Resequencing Systems with Partial Ordering", *PERFORMANCE*, pp. 433–445, 1987

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

[13] P. Vitanyi, "Distributed Algorithms in an Archimedean Ring of Processors", *ACM STOC*, pp. 542–547, 1984

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