Performance Models for Perfect and Imperfect Clocks on 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

Abstract

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.

Keywords : Concurrency control performance, reordering, clock drifts, distributed databases

1 Introduction

Research in the area of concurrency control for distributed database systems has led to the development of many concurrency control algorithms. 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 [ACL87], [EEAF91], [Gal84], [Li87], [SA86], [Sin91], [Tay87], [WL86]. There are two critical points in analyzing the performance of timestamp ordering concurrency control algorithms in distributed database systems. The reordering phenomenon and the effect of clock drifts.

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 [KTZ88] among the clocks. More specifically, if t is the global time and 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 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 $LC(i, t_n)$ denotes the timestamp associated with T_n. In case of perfect clocks 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 reason(the reordering phenomenon), there is a probability of abort, PA, for each transaction T_n.

So, we have the following queueing problem:

Every transaction T_n, with generation time t_n, timestamp the transactions which will arrive after it, and there is some fixed conflict probability between them, p_c. 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, using combinatorial arguments, the probability of two transactions having at least one common data object is

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

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

3.2 Case I: Perfect Clocks

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

$$p_i = \frac{1}{2}(\frac{\lambda_c}{\mu + \lambda_c})^i \tag{3}$$

and the probability of abort, PA, for each transaction is

$$PA = \sum_{i=1}^{\infty} p_i \pi_i \tag{4}$$

So, probability of not rejected is PNR,

$$PNR = 1 - PA \tag{5}$$

Other interesting performance measures are $Throughput(THR)$

$$THR = \lambda PNR \tag{6}$$

and $AbortRatio(AR)$

$$AR = \lambda PA \tag{7}$$

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 or before than it.

The probability of order reverse, in case of future is

$$p^f_{i\epsilon} = p_i Prob\{E^i_1\} \tag{8}$$

Also, in case of past we have that

$$p^p_{i\epsilon} = p_i(1 - Prob\{E^i_1\}) \tag{9}$$

Where,

$$\begin{aligned} &Prob\{E^i_1\} = \\ &= A \\ &+ \frac{1-A}{8\epsilon^2} \int_{w=0}^{2\epsilon} \left[4w\epsilon - w^2 + 4\epsilon\right] \frac{\lambda_c^i w^{i-1} e^{-\lambda_c w}}{(n-1)!} \, dw \end{aligned} \tag{10}$$

with

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

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

$$PA_\epsilon = \sum_{i=1}^{\infty} (p^f_{i\epsilon}\pi_i + p^p_{i\epsilon}\pi_{i+1}) \quad i = 1, 3, 5... \tag{11}$$

The probability of not rejected, PNR_ϵ, is

$$PNR_\epsilon = 1 - PA_\epsilon \tag{12}$$

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

$$THR_\epsilon = \lambda PNR_\epsilon \tag{13}$$

and $AbortRatio(AR_\epsilon)$ is,

$$AR_\epsilon = \lambda PA_\epsilon \tag{14}$$

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.

We observe that our analysis has the better accuracy in all cases except, when network has big traffic,case in which $\lambda >> \mu$, our analysis suffers a little. Figure 1 shows graphical representations for BTO. This Figure represents results for both the analysis and simulation. In all graphs the horizontal axis indicates the network service rate, μ, and the vertical represents our performance measures, such as PNR, THR and AR.

In all cases we observe that the analysis is very accurate. The effect of ϵ is very small for values effect in all performance measures for $\epsilon \geq 0.5$. Also the effect of ϵ is unsignificant 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 1: BTO analytical and simulation results, $\lambda = 6$,$M = 4$

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

[EEAF91] El–Toweissy M., El–Makky, Abougabal M., Fouad S., “The mean value approach to performance evaluation of the time–stamp ordering algorithms”, ICCI’91,Ottawa Canada, pp. 276–281, Springer–Verlag Lectures Notes in Computer Science No 497

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

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

[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

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

[Sin91] Singal M. , “Performance analysis of the basic timestamp ordering algorithm via Markov modelling”, Performance Evaluation 12(1991), pp 17–41

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