Predictive Models for Consistency Index of a Data Object in a Replicated Distributed Database System

SHRADDHA PHANSALKAR[1], A. R. DANI[2]
Symbiosis International University
Pune,
India
shraddhap@sitpune.edu.in

Abstract: - Consistency is a qualitative measure of database performance. Consistency Index (CI) is a quantification of consistency of a data unit in terms of percentage of correct reads to the total reads observed on the data unit in given time. The consistency guarantee of a replicated database logically depends on the number of reads, updates, number of replicas, and workload distribution over time. The objective of our work is to establish this dependency and finding their level of interactions with consistency guarantees to develop a predictor model for CI. We have implemented Transaction Processing Council-C (TPCC) online transactions benchmark on Amazon SimpleDB which is used as big-data storage. We have controlled the database design parameters and measured CI with 100 samples of workload and database design. The findings helped us to implement a prototype of CI based consistency predictor using statistical predictive techniques like a) Regression model and b) Multiple Perceptron neural network model c) Hidden Markov model. The data statistics show that the neural network based CI predictor causes less error and results in better coefficient of determination R^2 and mean square error (MSE).The Hidden Markov model based CI predictor is capable of modelling the effect of sequence of the input workload on the probability of obtaining a desired CI.

Key-Words: - Consistency Index (CI), predictive models, regression, neural network, Hidden Markov model

1 Introduction

Data consistency in replicated databases refers to the agreement of the data throughout multiple copies (replicas) in the system. It is always perceived as a functional requirement of database to guarantee consistency. Traditionally a database applies consistency constraints to all the data items without semantic discrimination. Data stores (SQL/No SQL) can tune the consistency with transactions. It is however observed that consistency should be perceived as a characteristic of data than a transaction. Also it should be selectively applied to different data objects in database so that the applications guarantee performance. A system should also be able to adjust the level of consistency of data on fly as opposed to predefined level of consistency for all the objects throughout. Consistency Index (CI) is defined as ratio of number of correct reads user observes to the total number of reads. CI can be considered as a measure of correctness of read transactions on a data item and can also be used to represent an expected value of consistency for a data item.

In this work we have modeled CI as a dependent variable and predicted it with input workload and the database replication policy. The input workload characteristics are number of reads, number of updates and average time gap elapsed between update and consecutive read. The other important determinant is the replication policy and number of replicas. This predictor predicts the value of CI for a given workload and the replicated system. This value can be then compared to the desired value and the determinants can be controlled accordingly. Thus development of CI predictor is the first step towards tunable consistency guarantees.

In this work we have built the predictor using a) Multiple Regression Model to validate our logical assumptions of dependencies and b) Artificial Neural Network to enhance the accuracy with improved values of R^2 and MSE. CI predictor with regression as well as neural network can represent the transactional workload (read and write operation) with the number of reads and updates only. CI also depends on the sequence of the arrival (schedule) of these operations. Hence we build the predictor with Markov-Chain model where the probability of occurrence of an operation (read, update) is assumed to follow Poisson distribution. Markov chain model finds the probability of obtaining a given value of CI for all possible sequences of workload in a given database design.

The article is organized as follows. Section 3 briefly describes the notion of consistency index. Section 4 briefs the experimental set up built for measuring CI. Section 5 briefs about the predictive model theory. Section 6 shows the statistical modeling of CI with multiple linear regression and section 7 shows the same with multiple perceptron models of neural networks. Section 8 discusses the hidden Markov chain model. Section 9 concludes our work with future scope.

2 Related Work

A study of research reveals several approaches [1, 2, 3, 4, 5] to detect consistency anomalies for different web-based applications in transactional or non-transactional context. *Probabilistically bounded staleness (PBS)*[1] provides probability approach using partial quorums to find expected bounds on *staleness* with respect to both versions of the replicas and wall clock time. Thus the consistency guarantees are predicted using probability. In [2], Zhang et al. proposes a window based quantitative model to describe consistency guarantee as number of missed updates. In TACT model [3], the authors develop a *conit-based* continuous consistency model to capture the consistency spectrum using three application-independent metrics numerical error, order error, and staleness for the replicated data where they also present the design and implementation of TACT, a middleware layer that enforces arbitrary consistency bounds among replicas using these metrics. Thus the consistency guarantees will be in the bounds of these metrics. In [4] authors proposes a transaction paradigm, that allows designers to define the consistency guarantees on the data and inconsistency is measured as number of conflicting updates on the data. In [5] authors propose TRAPP replicated systems, which uses a combination of locally cached bounds and exact master data stored remotely to deliver a bounded answer consisting of a range that is no wider than the specified precision constraint, that is guaranteed to contain the precise answer. In [6] a comparative study of various consistency models especially relaxed level of consistency and their impact on the performance like latency and throughput is studied. Thus the work encourages preferring relax level over strict levels of consistencies. In [7] Brewer states that compromising on weaker levels of consistency help us achieve the performance [CAP2000]. Hence consistency is now treated as an optimization problem [8]. Consistency optimization is effectively proposed in [9] by offering consistency as a service in Amazon Web services. A thesis report in [10] propose causal consistency model to be more effective than eventual consistency model.

All of the contributions present consistency as a database characteristic with a rigid, constant value. We briefly describe CI metric in section 3. A metric becomes more effective with the development of a predictive model which allows us to predict its value with some interacting factors. Tuning these factors, can help us achieve a desired level of the metric. A predictive model requires identification of major independent factors on which the outcome (dependent variable) depends. Many articles like [11] support the application of multiple linear regressions (MLR) for causal analysis of an independent variable on dependent variable. The factors which significantly affect CI were found with multiple regression analysis as discussed in [12]. Artificial neural networks (ANN) models are used to model the nonlinearity or interaction effects amongst the parameters in the model better than regression models as discussed in[13]. Hence ANN is used for predictive model. The Markov chain model [14] is a memory less random process where the next state depends only on the current state and not on the sequence of events that preceded it. The use of Markov chain is suggested as it predicts the probability of the correctness of a read operation in a workload sequence after the occurrence of preceding update operation on the data item.

3 Consistency Index

In a truly replicated distributed system, a read and update can occur simultaneously on any replica. This is absolutely essential for high availability and low response time. When a replica is updated, the update is to be conveyed to all other replicas in the system, which is c*onvergence* of data. This convergence may be carried immediately (strong consistency) or with some delay (eventual consistency). When a replica is not updated with a recent value, it is said to be *stale*. Any read on stale replica is an *incorrect read.* The period between the update on one replica and the convergence of all the replicas is *unsafe period.* A read in the unsafe period is highly probable to be an incorrect read.

Consistency Index CI) of a replicated data object is a fraction of correct reads on the object to the total number of reads on all its replicas in the system for an observed period. This definition of formula consistency index suggests that its value will lie in interval [0, 1]. Its value closer to 1, will indicate large number of correct reads. Whereas the value

close to 0 will indicate larger number of incorrect reads.

The frequency of correct reads would depend on how many reads fall in the unsafe period. In N-replicated system, the consensus period (agreement) between the replicas plays a vital role in deciding the time required to achieve the consensus amongst the replicas. Consensus period is directly proportional to the number of replicas. Higher the number of replicas, higher the period of consensus and thus higher is the probability of incorrect reads. Secondly the occurrence of read request and its relative occurrence after the update in real time decide the number of incorrect reads. This suggests that following factors affect CI.

1. Workload (number of reads, updates)
2. Average time gap between an update and read in real time.
3. Number of replicas.

This is statistically proved using multiple linear regressions in section 6.

4 Experimental Set up for measuring CI

In order to verify the effects of the aforesaid factors we built an experimental set up to collect the data. Hence the TPCC [15] schema was implemented on Amazon SimpleDB[16]. The document based No-SQL data store was selected because of its simplicity and flexibility. This schema consists of all normalized tables with data attributes. Some of these data elements are critical to the domain as incorrectness in their value would incur cost in terms of customer dissatisfaction or losses due to overselling. CI can be considered as data characteristic. The data can be a data aggregate, data object or a data attribute. The application developer is free to decide the granularity of control. For experimental reasons, we have chosen an attribute level of data granularity.

We observe the CI on the *stock_qty* attribute of a *stock _item* domain in the TPCC schema. The *stock_qty* undergoes many semantic transactions like *new_order*, *stock_level* etc. which are actually read and update operations in database. The tables in the schema were implemented as SimpleDB domains and the domains were replicated at the application level. The input to the system was a randomly generated workload which was in the form of the sequence of reads, updates and NOP (no operation). The application server would take the input stream of operations and delegate the same to the replicas following a load balancing policy. The load was balanced using the round robin allocation policy to isolate effects of a load balancing policy on CI. The system implements release level of consistency where the consistency was left to the programmer. The data access was done after the synchronization accesses which are broken into *acquire* and *release* locks. The synchronization accesses were not globally consistent. This exposed the reads to occur in the consensus period. We then observed the number of reads that fell in the consensus period, and effect of the number of replicas, number of updates, and number of NOPs in the input workload. The proportion of correct reads (reads that did not occur in the consensus period) to the total reads in the observed time of simulation gave us an experimental value of CI.

5 Statistical Predictive models of CI

Predictive modelling is a name given to a collection of mathematical techniques having in common the goal of finding a mathematical relationship between a “dependent” variable and various “independent” variables to measure or predict future values of the dependent variable. Regression techniques are most simple methods to build linear and general predictive models. The ANN models however are also used to model the nonlinearity or interaction effects amongst the parameters in the model. The Markov chain model is a random process usually characterized as memory less state where the next state depends only on the current state and not on the sequence of events that preceded it. The multiple linear regression based CI predictor (MLR-CI) in section 6 helps us infer the relationship between CI with the independent variables like number of reads(R), updates(U), replicas(R_p) and the time between read and update(T_g). The artificial neural network based CI predictor (ANN-CI) model in section 7 results in a better predictor of CI. The Hidden Markov based CI predictor (HMM-CI) in section 8 gives the probability of obtaining a desired value of CI if the general characteristic of the workload and replicated database design is known.

6 The MLR-CI predictor

Regression is defined as a statistical technique to predict an outcome of a response variable with model of the relationship between the explanatory variables and dependent variable. The explanatory variables are also known as independent variables. It is known as linear regression in case the relationship

Table.1 Model Summary of CI Using Regression Model

R	R^2	adj R^2	Std. Error	Change Statistics			
				F Change	df1	df2	Sig. F Change
.811	.658	.640	.1597	36.563	4	76	0

Table 2. Significance of the determinants and Coefficients of the variables

Model	**Un-standardized Coefficients**		**T**	**Sig** (p-value)
	B	**Std. Error**		
Constant(K)	1.277	.062	20.466	0
Number of Reads(R)	-.003	.002	-1.764	.082
Number of Updates(U)	-.012	.002	-5.970	0
Number of replicas(R_p)	-.181	.024	-7.414	0
Average Time gap (T_g) between reads and update (ms)	1.558E-005	0	2.085	.040

between dependent variables and independent variable is linear. If there is only one independent variable then linear regression is known as simple linear regression. It is known as multiple linear regressions in case the number of independent variable is more than 1. In this case number of independent variable is more than one. We build the predictive model for CI with independent variables such as number of reads (R), number of updates (U), number of replicas (R_p), average time gap between reads and updates(T_g) (number of NOPs). In the regression model the independent variables were entered simultaneously and the coefficient of determination (R^2) was found out. The F test was also carried to validate the null hypothesis about the relationship between variables. All the independent variables are *scale* in the nature. The value of R^2 is 0.658 and R^2_{adj} = 0.64 indicates that 65 % variability in the CI is accounted by all the predictors put together. The *Sig* in the table indicates that the hypothesis (Ho= independent variable has causal effect on the dependent variable) is supported by p-value of confidence. The p-value for U, R_p, T_g is less than the level of significance (α =0.05)and hence proved significant.The coefficients of the predictors obtained from table 2, indicates the following prediction equation for CI where K is a constant.

$$CI = K - 0.003 \times R - 0.012 \times U - 0.181 \times R_p + 1.558 \times 10^{-5} \times T_g \quad (2)$$

Table 3. MLR-CI: $R^2$and MSE

Parameter	R^2	R^2_{adj}	MSE
CI	0.658	0.640	0.075

7 ANN-CI Predictive Model

ANN can be used to transform inputs into meaningful outputs. ANNs have been widely used for classification and predictive tasks. ANNs are known to model complex nonlinear relationship between dependent and independent variables. We use the multilayer perceptron model to model the relationship of CI with the independent variables stated above. The independent variables are treated as *covariates*. The input data is partitioned as standard 70% data as training data set and 30 % testing data set. We have chosen the number of hidden layers to be one. The learning paradigm used was supervised learning. The commonly used performance metric is *mean-squared error* which tries to minimize the average squared error between the predicted value and observed value. Our halt function was consecutive steps with no significant improvement in the error. The ANN results are tabulated in tables 4 and 5.

Table 4. Model Summary of ANN-CI model

Training	Sum of Squares Error	9.986
	Relative Error	.350
	Stopping Rule Used	1 consecutive step(s) with no decrease in error
	Training Time	0:00:00.04
Testing	Sum of Squares Error	1.993
	Relative Error	.185

ANN does not provide the relative impact of a predictor on the response. The factor *updates* and *number of replicas* seem to be important as shown by the MLR model. However the number of reads and the read write time (T_g) seem to be less significant as contrary to the regression model. Table 2 and table 4 indicate that the MLR predictive model determines 65 % of variance in CI explained by the parameters whereas ANN explains 82.55% of the variance in CI by its covariates. The MSE with ANN is 0.025 as compared to 0.075 in MLR which proves that ANN-CI predictive model performs better.

Table 4.ANN-CI: $R^2$and MSE

Parameter	R^2	R^2_{adj}	MSE
CI	0.8336	0.8255	0.025

8 HMM-CI Predictive Model

HMM is suitably used to predict structure, function for sequences as in [17].The input workload to our system is a sequence of read and writes operations distributed on real time. As discussed, the input sequences follow Poisson's distribution [18] where the occurrence of a read or write operation is equi-probable. Logically the number of read and write operations as well as their schedule affects the occurrence of stale reads in the consensus period thereby affecting the value of CI. Modeling the input sequence requires use of Markov chain model which models the probabilities of state transition depending on its current state. This was precisely our requirement where the occurrence of a correct (incorrect) read depends on whether the current operation is idle/read/update. We built a Markov Model for predicting CI which represented the probability of occurrence of next operation (read, update, idle) in sequence thereby obtaining probability of an observed sequence. The summation of the probabilities of occurrence of the observed sequences leading to a desired number of correct reads (or more) yields the probability of obtaining a desired value of CI.

A data object can undergo one of the operations like U(update), I(idle), R(read) at a particular instance. Depending on the sequence of the input and their relative placement on time axis reveals that the read could lead to either of the two observed states of Correct read (CR), Incorrect read (ICR).

Hidden Markov Model (HMM) is a Markov chain where the state is only partially observable. In our problem, "read" is a hidden state and "correct read (CR)", "incorrect read (ICR)" are consequent observed states whose occurrence would depend on the previous state. Figure 1 shows the resultant HMM-CI model. I,U, R are equi-probable states and hence the probability of their occurrence is shown as 0.33. The occurrence of the observed states CR and ICR would depend on the number of replicas explained later. The time between the reads and updates is modeled using the idle state (I).

Figure 1. HMM-CI Model

8.1 HMM and sequences

HMMs allow you to estimate probabilities of unobserved events. In our problem we observe a sequence of updates (U), reads(R), idle (I) states i.e A = ('U', 'R', 'I'….) and we want to determine the probability of a sequence of observed states (O='U','CR'….'ICR'…..) i.e P (O|A).

The parameters of a HMM are:

a) States: S={ I,U,R,CR,ICR}
b) Transition probabilities: A= $a_{1,1}$,$a_{1,2}$,…,$a_{n,n}$ Each $a_{i,j}$ represents the probability of transitioning from state S_i to S_j.
c) Emission probabilities: A set B of functions of the form $b_i(o_t)$ which is the probability of observation o_t being emitted by S_i . O ={CR,ICR}
   P(CR|R) = $1/R_p$ (event that read occurs on the same replica where previous update occurred)
   P(ICR|R) = $(R_p-1)/R_p$ (event that read occurs on the different replica where previous update occurred)
   R_p number of replicas

d) Initial state distribution is the probability that S_i is a start state. P(I) = 1/3,P(U) = 1/3, P(R) = 1/3 as we assume that all the operations are equi probable.

By Markov chain property, probability of state sequence can be found by the formula:

$$P(s_{i1}, s_{i2}..s_{ik}) = P(s_{ik}|s_{i1}, s_{i2}..s_{ik-1})P(s_{i1}, s_{i2}..s_{ik-1}) = P(s_{ik}|s_{ik-1})P(s_{ik-1}|s_{ik-2})\ldots..\ (s_{i2}|s_{i1})P(s_{i1}) \quad (3)$$

Suppose we want to calculate a probability of an observed sequence of states in our example,
O= {'CR','U','CR','U', 'ICR'}.
P(O)= P({'CR','U','CR','U', 'ICR'})
= P('ICR'|'U') × P('U'|'CR') × P('CR'|'U') × P('U'|'CR') × P('CR')

$$= \frac{R_p - 1}{R_p} \times 0.33 \times 0.33 \times \left(\frac{1}{R_p} \times 0.33\right) \times 0.33 \times 0.33$$

8.2 Probability of getting CI = X

We apply the equation 3 to find the probability of an observed sequence from the HMM-CI model. However, the observed sequence is one of the desired sequences which yield CI greater than or equal to (≥) desired value.

$$P(CI = X) = \Sigma\ P(o \subset O \mid o \text{ leads to } CI >= X) \quad (4)$$

Table 5 shows the working of equation 3 and 4 on the example. We work on the workload (read, update): (R, U) = (2, 2).The number of replicas is 2. The desired level of CI = 0.5. Hence all the sequences which generate CI greater than or equal to 0.5 are desired observed sequences. For every desired observed sequence, we find the probability of the occurrence using equation 3. By equation 4, we sum the probabilities and obtain the probabilities of getting the desired CI. Referring to table 5, the probability of
CI ≥0.5 is found out as follows:

$\Sigma P(CI \geq 0.5)$=(0.003 +0.003 + 0.002+ 0.009 +0.006+0.006+0.009 + 0.006 +0.006 +0.006 + 0.0046)
= 0.0726

9 Conclusion

Our work uses a novel quantitative measure of data consistency with CI and predicts the value of CI using database design parameters and workload characteristics. This work majorly contributes in developing a predictive model of CI using statistical methods like regression, neural networks and Markov chain model. Regression model predicts CI with the suggested design parameters. ANN further enhances the prediction of CI with improved values of coefficient of regression (R^2) and mean square error (MSE). The HMM-CI model also includes the effect of input sequence and predicts the probability of obtaining desired CI with given input sequence. The work does not refer to the validation of these predictive models which can be further explored. Our predictive models can be definitely used to tune the interacting factors and to obtain a desired value of CI. Thus the work lays the foundation of tunable guarantees of data consistency achieving desired levels of CI with minimal loss in the performance.

Table 6. Computing P (CI ≥ 0.5) Using HMM-CI predictive model for (R, U) = (2, 2)

Input sequence	**Observed sequence set(O)**	**o ⊂ O such that CI(o) ≥0.5**	**P(a, b, c, d) = P(a) × P(b|a) × P(c|b) × P(d|c)**
U R U R	U CR U CR UCR U ICR U ICR U CR U ICR U ICR	U CR U CR U CR U ICR U ICR U CR	0.003 0.003 0.002
U R R U	U CR ICR U U ICR CR U U CR CR U U ICR ICR U	U CR ICR U U ICR CR U U CR CR U	0 0.009 0.006
U U R R	U U CR CR U U ICR ICR U U CR ICR U U ICR CR	U U CR CR U U CR ICR U U ICR CR	0.006 0 0.009
R U U R	CR U U CR ICR U U ICR ICR U U CR CR U U ICR	ICR U U CR CR U U ICR CR U U CR	0 0.006 0.006
R U R U	CR U CR U ICR U ICR U ICR U CR U CR U ICR U	CR U CR U ICR U CR U CR U ICR U	0.006 0 0.0046
R R U U	CR ICR U U ICR CR U U CICR CR U U ICR ICR U U	CR ICR U U ICR CR U U CR CR U U	0 0 0.012

References:

[1] Bailis P., Venkataraman S., Franklin M. J., Hellerstein J. M., and Stoica I. "Probabilistically Bounded Staleness For Practical Partial Quorums". Proc. VLDB Endow. 5, 8 (Apr.2012), 776–787.

[2] Zhang C., Zhang Z., "Trading Replication Consistency for Performance and Availability:

an Adaptive Approach", Proceedings of the 23rd International Conference on Distributed Computing Systems, May 19-22, 2003, p.687

[3] Haifeng Y., Vahdat A., "Design and Evaluation of a Conit-Based Continuous Consistency Model for Replicated Services", ACM Transactions on Computer Systems, Vol. 20, No. 3, August 2002.

[4] Kraska T., Hentschel M., Alonso G.,Kossmann D. "Consistency Rationing in the Cloud: Pay Only when it Matters". In Proc. of VLDB, volume 2, 2009, pages 253–264

[5]Olston C., Widom J., "Offering a Precision-Performance Tradeoff for Aggregation Queries over Replicated Data, Proceedings of the 26th International Conference on Very Large Data Bases, September 10-14, 2000, p.144-155,

[6] Kourosh G., "Memory Consistency Models for Shared memory Multiprocessors", PhD thesis, Tech. Report CSL-TR-95-685, Stanford Univ., 1995.

[7] Brewer E., "Towards Robust Distributed Systems", Proceedings of 19th ACM Symposium on Principles of Distributed Computing *2000,* pp. 7-10.

[8] Redding D., Florescu D., Kossmann D., "Rethinking Cost and Performance of Database Systems", ACM SIGMOD Record, Vol 38, Issue 1, March 2009 pgs. 43-48

[9] Liu Q., Wang G., and Wu J., "Consistency as a Service: Auditing Cloud Consistency", *IEEE Transactions on Network and Service Management,* Vol. 11, No. 1, March 2014.

[10] Padhye V., "Transaction and data consistency models for cloud applications", Thesis submitted to University of Minnesota, Feb 2014.

[11] Allison P., "Prediction vs. Causation in Regression Analysis" blog July 8, 2014 http://www.statisticalhorizons.com/prediction-vs-causation-in-regression-analysis (Accessed June 2014)

[12] Leona S. Aiken and Stephen G., "Multiple Regression Model: Testing and Interpreting Interactions", West. Newbury Park, CA: Sage, 1991, 212 pp

[13] Dreiseitl S., Ohno-Machado L., "Logistic regression and artificial neural network classification models: a methodology review", J. Biomed. Inform. 35 (2002) 352—359

[14] Sung-Jung Cho, "Introduction to Hidden Markov Model and Its Application", Samsung Advanced Institute of Technology (SAIT), April 16, 2005.

[15] TPCC Benchmark Standard specification revision5.11, www.tpc.org/tpcc/ February 2010

[16] Amazon SimpleDB Documentation, http://aws.amazon.com/simpledb/ 2010

[17] Tsoumakos D., Konstantinou I., Boumpouka C., Sioutas S., & Koziris N. (2013, May). "Automated, elastic resource provisioning for NoSql clusters using Tiramola", 13th IEEE/ACM International Symposium in Cluster, Cloud and Grid Computing (CCGrid) 2013, pp. 34-41.

[18] Guenni L., Lovric M. *(Ed.)* Poisson distribution and Its Application in Statistics *International Encyclopedia of Statistical Science,* Springer Berlin Heidelberg, 2014.

[19] Field A., "Discovering Statistics using IBM SPSS Statistics" Sage Publications, 2013.