Queue Weighting Load-Balancing Technique for Database Replication in Dynamic Content Web Sites

EBADA SARHAN* , ATIF GHALWASH*, MOHAMED KHAFAGY**

* Computer Science Department, Faculty of Computers & Information, Helwan University, Egypt
** Computer Science Department, Faculty of Information Systems & Computer Science, 6th October University, Egypt
M_h_khafagy@hotmail.com

Abstract: - There is an ever increasing need for database replication in dynamic web sites to improve availability. However, the main problem in replication is load balancing. This paper presents new load balance technique to increase the performance of database replication in dynamic web depending on the type and weight of database server queue. We attempt at evaluation various load distribution policies, taking in account their ability to achieve good load balancing by using LMB (Load Balance Metric), and also their impact on performance by measuring the throughput. The telecommunication benchmark is used to compare the different policies of load balancing.

The telecommunication benchmark, a powerful benchmarking tool, is used to test up to fifty database replicas that will play a great role in the evaluation process which could be performed through measurements on a web site that follows the TPC-W specifications. The results show that the Queue weighting Load Balancing has maximum LMB and best throughput.

Key-Words: - Replication, Availability, Fault Tolerance, Dynamic web site, Agent, Database

1 Introduction

Replication has become a central element in modern information systems. It significantly contributes to enhancing availability and performance. Yet, the main problem in replication is load balancing.

Load balancing is a mechanism where the server load is distributed to different nodes within the server cluster, based on a load balancing policy. Rather than executing an application on a single server, the system executes transactions on a dynamically selected server. When a client requests a transaction, one (or more) of the cooperating servers is chosen to execute the request. Load balancers act as single points of entry into the cluster and as traffic directors to individual web or application servers[1].

The load balancer receives each request and rewrites headers to point to other machines in the cluster. If any machine in the cluster is removed, the changes take effect immediately.

There are many different algorithms used to define the load distribution policy, ranging from a simple round robin algorithm to more sophisticated algorithms used to perform the load balancing. Some of the commonly used algorithms are:

Round-robin
Random
LARD
Weight-based

Load-balancing algorithms affect statistical variance, speed, and simplicity. For example, the weight-based algorithm has a longer computational time than other algorithms.

Queue weighting Load Balancing load balance technique enhance the performance of database replication in dynamic web, depending on the type and weight of database server queue. We use the DWT-B Telecommunication benchmark [2] in evaluation with the same specifications of TPC-W [3]. This benchmark specifies three workloads (Read, Write and Mix) with different percentages of writes in the workload. The evaluation uses simulation to confirm the performance effects of larger clusters.

The simulations show that the Queue weighting Load Balancing has maximum LMB and best throughput compared with different load balancing techniques.

The outline of this paper is presented as in the following: section 2 provides the Database Replication of Dynamic Content Web Sites, section 3 introduces other load balance technique introduced for comparison with the Queue weighting Load Balancing, section 4 describes our load balance technique and section 5 presents the Telecommunication Benchmark and its metrics. The results are outlined in section 6, and finally section 7 highlights the conclusion of the present paper.

2 Database Replication of Dynamic Content Web Sites

Today, dynamic content servers used by large Internet sites, such as Amazon and pc2call, employ a three-tier architecture that consists of a front-end web server tier, an application server tier that implements the business logic of the site and a back-end database tier that stores the content of the site. The first two tiers, the web and the application server, typically use nonpersistent data and are generally hosted on inexpensive clusters of machines. However, the database tier storing persistent data is centralized and hosted on a high-end multiprocessor [4].

Recently, several research prototypes have proposed using replicated databases built from commodity clusters as a more economical solution. These replicated databases, which have been used for running a single application, such as, an e-commerce benchmark , have shown good performance scaling with increasing replication. but the main problem in replication is load balancing[5].

3. Load balancing techniques

3.1 Round-Robin (ROUND-ROBIN):

In this technique the application server assigns the requests to the servers in a circular order [6].

3.2 RANDOM

Random Distribution (RANDOM):

The application server assigns the request to database server randomly[7].

3.3 Locality – Aware Request Distribution Scheme (LARD)

Locality-Aware Request Distribution (LARD) was improved and appeared to be successful for load balancing static content requests in a cluster [8]. The aim of LARD is to combine good load balance and high locality for increased hit rates in the data caches of each back-end. In LARD. When a new query arrives, accessing a certain set of tables, the scheduler computes the type of request and assigns it to a certain server[9].

Fig. 1 illustrates the principle of LARD in a cluster with two back-ends and a working set of three targets (A, B, and C) in the incoming request stream. The front-end directs all requests for A to back-end 1, and all requests for B and C to back-end 2. By doing so, there is an increased likelihood that the request finds the requested target in the cache at the back-end.

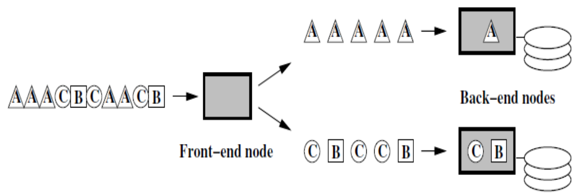

Fig. 1: LARD Load Balancing

3.4 Weighted Round Robin

Weighted round- robin is a common load balancing scheme in static- content cluster servers. The incoming requests are distributed in round- robin fashion. Weighted by an estimate of the load on the different back- ends[10].

3.4.1 Shortest Queue First (SQF)

The Shortest Queue First (SQF) uses the numbers of outstanding queries to a particular back- end as an estimate of the load on that back- end by determine the length of every database server queue [11].

We illustrate this technique using the example in Fig. 2. Assume that the SQF scheduler has placed queries Q1, Q2, Q3 and Q4 on the two database machines.

With respect to SQF, the two machines have optimal load balance (i.e., the same queue length). However, in this situation the total database engine load is not clearly balanced, as a result of the large differences query complexities. Even worse, all subsequent operations (i.e., Q5 and Q6) have to wait for the machine with the longest query times to finish.

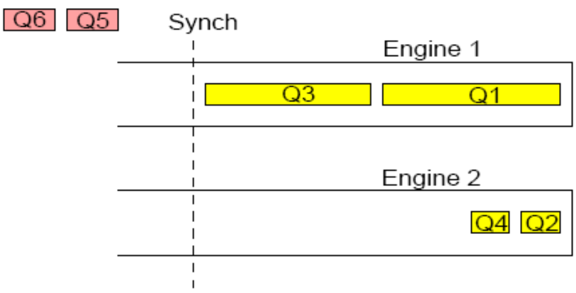

Fig. 2: SQF Load balancing

3.4.2 Shortest Execution Length first (SELF)

With Shortest Execution Length first (SELF) the execution time for calculating each query on an unloaded {idle} machine is measured off-line. The load on a particular back- end is estimated afterwards as the sum of the {measured} execution times for all queries outstanding to that back-end.

The execution time for each query on an unloaded (idle) machine [12] is measured off-line with shortest Execution Length First (SELF). At run-time, the load on a particular back-end is estimated as the sum of the (a priority measured) execution times for all queries outstanding at that back-end by the scheduler. As opposed to SQF, that treats each query as equal, SELF tries to take into account the widely varying execution times for different query types.

SELF makes a better load balancing strategy for e-commerce workloads through the wide range of query execution times.

4 Queue weighting Load Balancing

Assigning transactions to preferred servers is an optimization problem. It consists of distributing the transactions over the replicas S1, S2, ..., Sn. When assigning transactions to database servers.

The load balancing in each replica is measured by using Queue weighting. The weight for read transaction is 1, write transaction is 2 and update Transaction is 3[13]. The summation of all weights is calculated in every queue at database tier, and the queue length updates the weights every timeout. The Sequencer Agent chooses the minimum weight from the set of replicas that finish prior conflicting transactions [4].

1. Consider replicas S1, S2, ..., Sn. Transaction Ti belongs to Stk at time t, Ti € Stk, if at time t Ti is assigned to execute on server Sk.

2. Assign read-only transaction Ti to the replica Sk with the lowest Queue weight Qw(Sk, t) at time t, where Qw(Sk, t) =∑Tj € Stk wj where w =1 if T is read transaction, w=2 if T is write transaction and W=3 if T is update transaction.

A simple example

Consider a workload with 10 transactions, T1, T2, ..., T10, running in a system with 4 replicas S1, S2, S3, S4 and (T1,T4,T7,T10) are read transactions, (T2,T5,T8) are write transactions and (T3,T6,T9) are update transactions.

1. In the beginning of transactions all QW(Si)=0 then we assign T1 to S1 after this transaction the QW(s1,s2,s3,s4) =(1,0,0,0)

2. We assign T2 to S2 after this transaction the QW(s1,s2,s3,s4) =(1,2,0,0)

3. We assign T3 to S3 after this transaction the QW(s1,s2,s3,s4) =(1,2,3,0)

4. We assign T4 to S4 after this transaction the QW(s1,s2,s3,s4) =(1,2,3,1)

5. We assign T5 to S1 after this transaction the QW(s1,s2,s3,s4) =(3,2,3,1)

6. We assign T6 to S4 after this transaction the QW(s1,s2,s3,s4) =(3,2,3,4)

7. We assign T7 to S2 after this transaction the QW(s1,s2,s3,s4) =(3,3,3,4)

8. We assign T8 to S1 after this transaction the QW(s1,s2,s3,s4) =(5,3,3,4)

9. We assign T9 to S2 after this transaction the QW(s1,s2,s3,s4) =(5,6,3,4)

10. We assign T10 to S3 after this transaction the QW(s1,s2,s3,s4) =(5,6,4,4)

5 Benchmarks Platform

5.1 Telecommunication Benchmark

Telecommunication benchmark presents a benchmark used to evaluate database performance for telecommunication sites with dynamic content. The model covers most important features of dynamic website and telecommunication requirements. The services are modeled using simple transactions that represent the services.

The benchmark also uses a workload model from telecommunication website. In addition, the measurement of response time is added. The client emulator invokes oracle performance view that collects CPU, memory, I/O (input and output) and Response Time from the oracle Performance View.

5.2 Software Environment

The Benchmark use C# to make the site and use thread technique to implement clients' connection. We use Oracle 10g as a database server [14].

5.3 Hardware Platform

The Web server and the database server run on an Intel 2.2 GHz Dual core CPU with 2GB RAM, and a Maxtor 160 GB 5,400 rpm disk drive. A number of 2 GHz Intel machines run the client emulation software. There must be sufficient client emulation machines to guarantee that clients do not impede any of the experiments. All machines have to be connected through a switched 10/100 Mbps Ethernet LAN and the server connected with 10/1000 Ethernet LAN.

5.4 Workloads and Application Sizing

The Benchmark presents three different workloads, the browsing contains read-only scripts, the calling contains write scripts while the charging mix contains both read and update scripts. The database contains 10,000 accounts where every client can mix transaction from more than 10,000 mixed transactions randomly. For the telecommunication site, three workload mixes are used: a browsing call History mix made up of only read-only transactions, a site with a large user base in which 99.5% of accesses are reads [2] and a Calling mix that includes write interactions and charging made up of read-write transactions. There are always about 10,000 accounts, and a history of 500,000 calls in the History table is kept. It is assumed that users give feedback for call transactions. The (Delay Time) think time is used and the session time is specified by TPC-W.

5.5 Metrics

We study the load balancing effects of the proposed heuristics in both experiment settings. The Load Balance Metric (LMB) [15] is used as a performance metric for comparing results. To obtain the LMB value, the peak-to-mean ratio of server load is measured at different sampling points in the simulation.

The server load is defined as the utilization value of the server node. The LBM (1) value is obtained by calculating the weighted average of the peak-to-mean ratios measured, using the total server load as the weight for the sampling period in question. A smaller value indicates a better load balancing performance.

Load i,j – load of server i (of n servers) at the j th sampling point .
Peak_loadj – highest load on any server at the j th sampling point.

$$LMB= \frac{peak\_load_j}{(\sum_{i=1}^{n} load_{i,j})/n})$$

6 Results

Fig. 3 shows the throughput of the various algorithms for Load balance. The x-axis demonstrates the number of database machines, and the y-axis demonstrates the number of transactions per second. It is shown that the Queue weighting has a large throughput, and thus it may be concluded that the Queue weighting has best load balance. Moreover, it is also concluded that the Round robin algorithm has the lowest throughput.

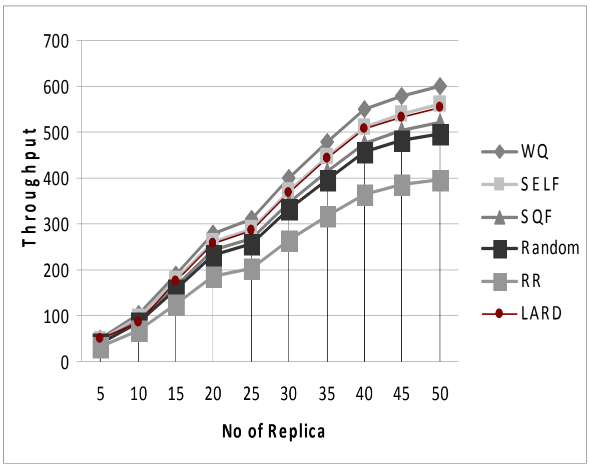

Fig. 3: Throughput

The LBM is used to compare the load balancing techniques. This comparison is illustrated in fig. 4. It is hence inferred that the Queue weighting technique has the best load balance with the increase of the number of transactions and that Queue weighting is the best technique in load balance compared with other techniques in the area of replication in dynamic content web sites.

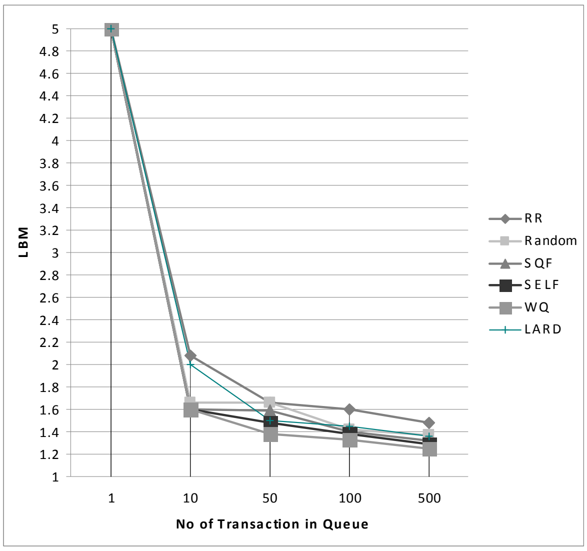

Fig. 4: LBM

7 Conclusion

We described the Queue weighting Load Balancing technique for database load balance serving as a backend database to a dynamic site. Queue weighting Load Balancing depends on the type and weight of database server queue. We evaluated Queue weighting Load Balancing by measuring simulation. Software platforms were employed commonly using: C# and Oracle 10g database. Various workload mixes of the TPC-W benchmark were used to evaluate and compare between all algorithms. Our simulations show that the Queue weighting Load Balancing has maximum LMB and best throughput compared with different load balancing techniques.

References

[1] Qi Zhang, Alma Riska, "Workload-Aware Load Balancing for Clustered Web Servers", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, VOL. 16, NO. 3, MARCH 2005.

[2] E. Sarhan, A. Ghalwash, M. Khafagy, "Specification and Implementation of Dynamic Web Site Benchmark In Telecommunication Area ", *12th WSEAS international Conference on COMPUTERS, Heraklion, Greece*, 2008

[3] Transaction Processing Council www.tpc.org,2009

[4] E. Sarhan, A. Ghalwash, M. Khafagy, "Agent-Based Replication for Scaling Back-end Databases of Dynamic Content Web Sites", *12th WSEAS international Conference on COMPUTERS, Heraklion, Greece*, 2008.

[5] Michael DePhillips, Jerome Lauret," Replication and load balancing strategy of STAR's Relational Database Management System (RDBM)", *International Conference on Computing in High Energy and Nuclear Physics*, 2008.

[6] Alissa Kaplunova, Atila Kaya, Ralf M¨oller," First Experiences with Load Balancing and Caching for Semantic Web Applications", *institute for software, technology and systems*, 2006.

[7] Brighten Godfrey, Karthik Lakshminarayanan, "Load Balancing in Dynamic Structured P2P Systems*", IEEE INFOCOM* 2004

[8] Vivek S. Pai, Mohit Aron, Gaurav Banga, Michael Svendsen, Peter Druschel, Willy Zwaenepoel, and Erich Nahum." Locality-aware request distribution in cluster-based network servers". *In Proceedings of the Eighth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 205–216, October 1998.

[9] Sameh Elnikety," Tashkent+: Memory-Aware Load Balancing and Update Filtering in Replicated Databases", *ACM EuroSys'07, Lisbon, Portugal*. 2007.

[10] DER-CHIANG , FENGMING M. CHANG, "An In–Out Combined Dynamic Weighted Round-Robin Method for Network Load Balancing*", Oxford University Press on behalf of The British Computer Society*, 2007.

[11]. C. Amza, A. Cox, and W. Zwaenepoel. Scaling and availability for dynamic content web

sites. *Technical Report TR02-395, Rice University*, 2002.

[12] F. Pedone, R. Guerraoui, and A. Schiper. Exploiting atomic broadcast in replicated databases. *In Proceedings of EuroPar* September 1998

[13] Oracle Database 10g: SQL Tuning 2006

[14] Oracle http://www.oracle.com,2009

[15] Richard B. Bunt, Derek L. Eager, "Achieving Load Balance and Effective Caching in Clustered Web Servers*", Proceedings of the Fourth International Web Caching Workshop, San Diego, California*, pages 159-169, 1999.