Analysis of Database Replication Algorithm in Local and Global Networks

Sanjay Kumar Yadav
Dept. of Computer Science & IT
SHIATS-Deemed University
Allahabad, India

Gurmit Singh
Dept. of Computer Science & IT
SHIATS-Deemed University
Allahabad, India

Divakar Singh Yadav
Dept. of Computer Science & Engg
Institute of Engineering and Technology
Lucknow, India

ABSTRACT
In this paper, a database replication algorithm is presented. The main idea is to reduce the latency in database replication while maintaining very high throughput. The main points of the algorithm are detailed in this paper. Simulation results are presented to evaluate throughput and average delay. For the in-depth analysis of the algorithm, various cases are considered and it has been found that if the numbers of servers that can serve requests are larger in number then throughput is very high with very less average delay. The overall, throughput and average delay also depend heavily on the load and if load is comparatively less (< 0.8) then the throughput is very high and average delay is nearly zero.

General Terms
Distributed Database Replication Algorithm

Keywords
Database Replication, Throughput, Average Delay

1. INTRODUCTION
The process of copying and maintaining database objects, like tables, in multiple databases that generates a distributed database system form one location to another is called Data replication [1]. In database replication, changes made at one particular site, anywhere in the world are captured and stored locally before they are being forwarded and applied at each of the remote locations [2]. The centralized or distributed database technology is being used by the replication in order to share data between multiple sites. It must be remembered that there is difference between a distributed database and a replicated database. As far as a distributed database is concerned, data is generally available at several locations, but the location of a particular table is unique i.e. it resides at only one location [3].

Some of the most common but vital reasons for the usage of replication are described as follows: Replication provides fast, local access to shared data as it balances activity over multiple sites. As some users can access one server while the other users access different servers, the result will be the reduction of the load at all servers. In addition, use of the replication site for accessing data from the users in turn results in low access cost. Furthermore, this site is usually the geographically closest site to the users.

1.1 Centralized and Distributed Database Systems
In the traditional enterprise computing model, the control of a centralized corporate database system is done by an Information Systems department [4]. The required performance level is provided by the mainframe computers which are generally located at corporate headquarters. With the use of applications provided by the department of Information Systems (IS), the corporate database is accessed by remote sites through (WANs) [4]. Due to the corporate environment changes toward decentralized operations, the organizations are encouraged to move toward distributed database systems that complement the new decentralized organization [4].

At present time, global enterprise may have many (LANs) joined with a (WAN), as well as additional data servers and applications on the local-area networks. At the sites, client applications need to access data locally or remotely through the local-area networks through the wide-area network respectively. For example, a client in Delhi might locally access a table stored on the Delhi data server or remotely access a table stored on the New York data server [4].

Mainframe computers may be needed at regional headquarters or corporate headquarters in a distributed database environment in order to maintain sensitive corporate data. On the other hand, minicomputers and server-class workstations are used by the remote sites clients for local processing. The centralized as well as distributed database systems must deal with remote access problems:

- When WAN traffic is heavy, network response slows and leads to huge amount of data access delay.
- A centralized data server can become bottleneck as a huge user community contends for data server access.
- No Data is available on the network when a failure occurs.

1.2 Advantages of Replicating Data
The problems related with remote database access such as performance and availability problems can be solved by replicating the data from its source database to a local database. A cost-effective and fault-tolerant system is provided by Replication Server for replicating data.

The data is kept up to date in multiple databases by the Replication Server so that clients can access local data in place of remote, centralized databases. An improved data availability, system performance and reduced communication overhead is provided by a replication data system in comparison to a centralized data system. As it transfers transactions, not rows, the integrity of replicated data is maintained by Replication Server across the system, along with increasing data availability. Replication Server also allows the replication of stored procedure invocations, and so further improving performance [4].

.

1.3 Data Distribution with Replication Server

Replication Server performs to distribute data over a network by [5]:

- It provides application developers and system administrators with a flexible publish-and-subscribe model for making data and stored procedures to be replicated.
- The Management of replicated transactions across the network while retaining transaction integrity.

Since Replication Server replicates transactions in incremental form instead of data copies and it stores procedure invocations, not the stored procedures themselves, it provides an environment of high-performance and easily accessible distributed data while maintaining data integrity.
A fundamental challenge in database replication is maintaining a low cost of updates while assuring global system consistency. The problem is magnified for wide area replication due to the high latency and the increased likelihood of network partitions in wide area settings [5].

Therefore, in database replication, the location of nodes and their availability is important. In recent past, much work is done on the database replication in wide area networks [6-11]. In the similar context Y. Amir, presented the idea of portioning of the networks to reduce the load and thus to achieve better performance [12, 13]. In our previous work, PDDRA (Pre-fetching Based Dynamic Data Replication Algorithm [14]) have been modified to reduce the network based latency and a mathematical model has been presented to evaluate the throughput and average delay [15]. In this paper, we further investigated the proposed scheme, and simulation results are presented to evaluate the throughput and average delay.

The rest of the paper is organized as follows: In section 2, the M-PDDRA algorithm is described. The simulation results are presented in section 3. The major conclusions of the paper are discussed in section 4 of the paper.

2. M-PDDRA ALGORITHM

In [14], A PDDRA (Pre-fetching Based Dynamic Data Replication Algorithm) is presented. The main idea is to pre-fetch some data using the heuristic algorithm before actual replication start to reduce latency. In our previous work, modifications in PDDRA (M-PDDRA) are suggested to further reduce the latency. For more detail please refer to [14]. The main points of the algorithm are summarized as follows:

1. In M-PDDRA scheme the internet cloud will be considered as master node as it can be assumed that the data is available in the internet for the replication (Fig. 1).

2. If any replication request is generated by a node then via edge node it will be searched in local network, and a simultaneous request will also be send to the global network.

3. There is a possibility that the data may not be available at any local node or waiting time is too large, as simultaneous request is sent to both to a local node and master node, if access of master node is in queue for let's say time t_q then local search will only be done for time $t_s < t_q$. These simultaneous requests to both local and global network will reduce latency in comparison to first request send to local network then thereafter to global network.

In the simulation all the three possible cases will be considered and results in terms of throughput and average delay will be presented.

3. SIMULATION FRAMEWORK AND RESULTS

The replicated data is either available locally or it is available globally i.e., at the internet. Therefore, when requests are generated, some of the generated requests will be full-filled locally and leftover requests will be fetched from internet (master node). In this section simulation framework is developed to estimate the average response time of all the transactions. In nutshell there are four processes in database replication:

- Request generation at the local node
- Request serving at the local node
- Network propagation
- Request serving at the remote site in the global network

The important network parameters are:

Network Throughput: It refers to the volume of data that can flow through a network, or in other words, the fraction of the generated request which can be served.

Network Load: In networking, load refers to the amount of data (traffic) being carried by the network.

Network Delay: It is an important design and performance characteristic of data network. The delay of a network specifies how long it takes for a request to travel across the network from one node or endpoint to another. Delay may differ slightly, depending on the location of the specific pair of communicating nodes.

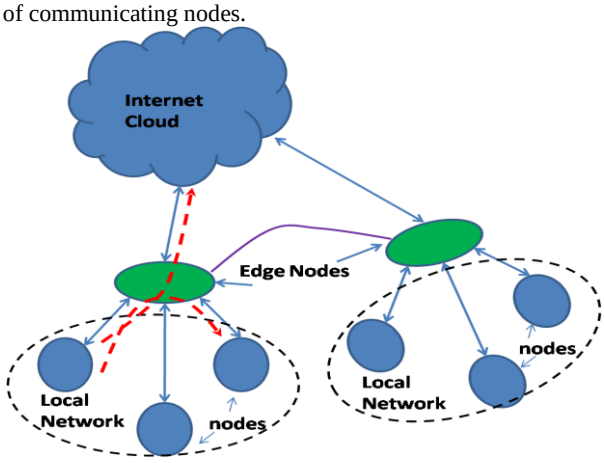

Fig. 1 Schematic of the database replication in network

Let in the local network, r requests are generated with probability p_r (it is assumed that every requests are generated with equal probability), out of which l requests are served locally with probability p, and then $(r-l)$ requests will be transmitted to the outer world (global network). Hence, transaction arrival rate is represented by λ

$$\lambda_i = rp_r \qquad (1)$$

Then the mean value of the requests served locally is

$$\lambda_{av}^{L} = \lambda_i p \quad (2)$$

And the requests served globally are

$$\lambda_{av}^{G} = \lambda_i (1-p) \quad (3)$$

Total requests served are

$$\lambda_i = \lambda_{av}^{L} + \lambda_{av}^{G} \quad (4)$$

$$\lambda_i = r p_r p + r(1-p) p_r \quad (5)$$

The throughput can be calculated as

$$T = \frac{a}{100} T_{av}^{L} + \frac{(1-a)}{100} T_{av}^{G} \quad (6)$$

The total delay can be evaluated as

$$D = D_{av}^{L} + D_{av}^{G} \quad (7)$$

The simulation is done in MATLAB. The simulation pattern is based on random number generation and well known as Monte Carlo simulation. In the simulation random traffic model is considered. In the simulation we have assumed:

1. Request can be generated at any of the input with probability p_r .
2. With probability p the generated request served locally.
3. Each request is equally likely to go to any of the servers (locally or globally) with probability $\frac{1}{N}$ where *N* is the number of servers available.

The probability that K requests arrive at the particular server is given by

$$P[K] = {}^{N}C_{K} \left(\frac{p}{N}\right)^{K} \left(1-\frac{p}{N}\right)^{N-K} \quad (8)$$

In the simulation synchronous network is considered hence time is divided into slots. Therefore following assumptions are made:

1. The requests can be generated at the slot boundary only.
2. Updates can be made at the slot boundary only.
3. Once replication is started, then further updation is not allowed till replication complete.

As a network perspective, there is no distinction in local and global network and same queuing structure is applicable. Considering the figure 2, a request generated in cluster 'A' will treat cluster 'A' as LAN (Local Area Network) and Clusters 'B', 'C' and 'D' will be a part of global network. Similarly, a request generated in cluster 'D' will treat cluster 'D' as local network while 'A', 'B' and 'C' will be treated as global network.

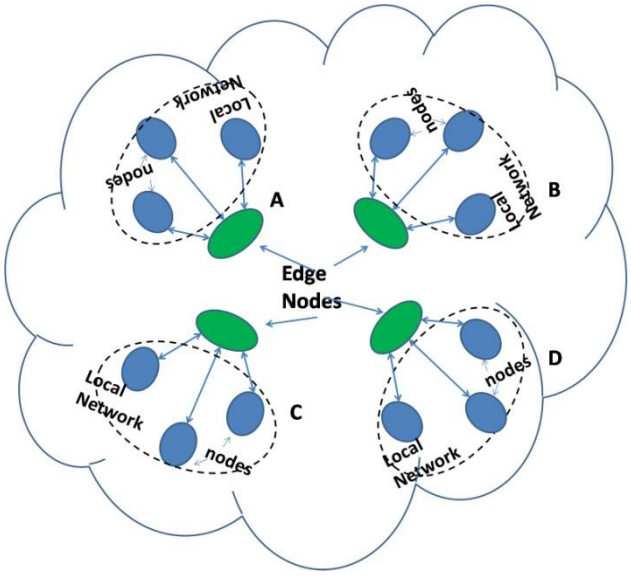

Fig. 2: The network structure for local and global network

In the simulation, various cases are considered.

Case I:

In the first case, we assumed that at the local network No request can be full-filled and therefore all the generated requests will be sent to the global network.

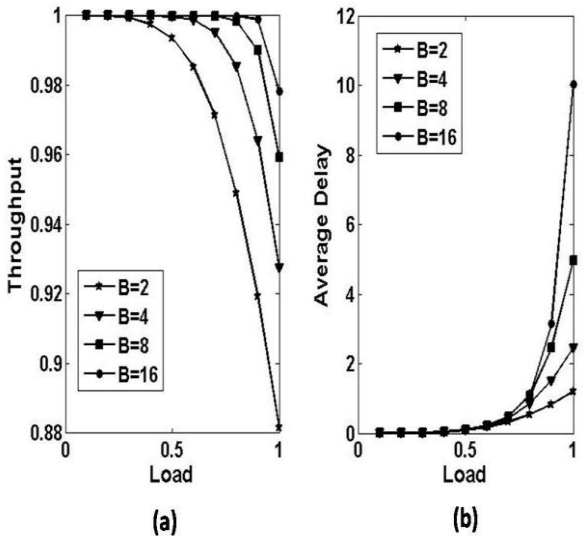

Fig. 3: (a) throughput vs. load and (b) delay (in slots) vs. load with varying buffering capacity while considering request generating nodes and servers (N) are 4.

In figure 3, the throughput (a) and average delay (b) vs. average load is plotted with varying number of storage capacity '***B***' (in terms of number of requests that can be stored at a server node). While considering that the request generating nodes are four and server nodes are also four. It is evident form the figure 3(a), to get at least 90% throughput, we need buffering capacity of 4 requests. It is also noticeable that as the storage capacity is increased, the throughput increases. It can be observed from the figure that, even at the higher load > 0.8 the throughput can be achieved up to 98%. In figure 3(b): average delay vs. load with varying buffering capacity while considering request generating nodes (***N***) as 4.

As expected as buffer space increases, the average delay also increases. It is also observed from the figure, that below the load of 0.5, the average delay is nearly zero, and thereafter it rises exponentially. This is also very obvious that as the load increases more number of requests arrive and to sustain the throughput buffer space has to be increased and thus the average delay also increases.

Case II:

In the second case, we assumed that at the local network some requests can be full-filled and therefore all the generated requests will be shared between the local and global networks.

In figure 4(a), throughput vs. load is plotted while; **'*a*'** denotes the percentage of traffic that is served globally, for example *a*=80 denotes that 20% of the total generated requests are served locally and 80% of the generated traffic is served globally. It is observed from the figure; if 80% of the total traffic is served globally then 100% throughput is possible. However, as traffic crosses 80% limit the throughput decreases and attain a value of 96%. For the similar values as in figure 4(a), the average delay vs. load is plotted in figure 4(b).

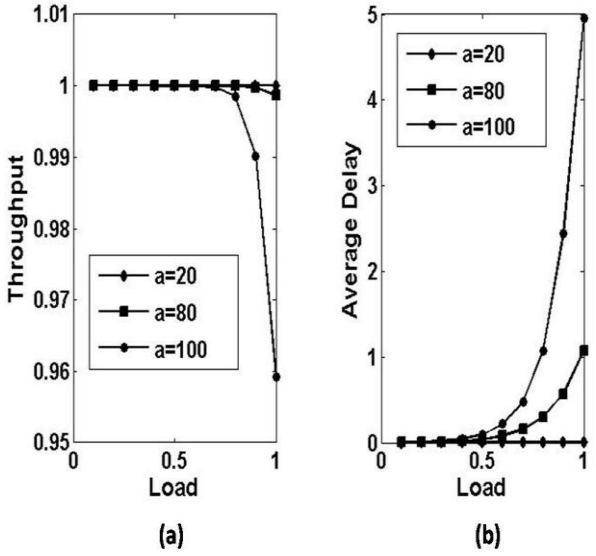

Fig. 4: (a) throughput vs. load and (b) delay vs. load with fixed buffering capacity (B) of 8 while considering request generating nodes and servers (N) are 4 while considering both local and global servers.

It can be observed form the figure, that when only 20% of the total traffic is served globally then the average delay is nearly zero. For 80% of the traffic the average delay is zero below load 0.6. Above the load 0.6, the average delay increases and attains a value 1 at the load 1. However, if all the traffic is served globally the delay is significant and at the load 1 it attains a value of 5 slots. It must be remembered that, if we assume same set of values for the local network, then the queuing structure will remain the same for local and global network. In view of this, considering figure 4, then results for *a*=20% is for local database and *a*=80% are for global database. Using the equation 6, the total throughput at the load of 0.9 or below is

$$T = \frac{20}{100} T_{av}^{L} + \frac{80}{100} T_{av}^{G} = 0.2 \times 1 + 0.8 \times 1 = 1.0$$

However, the total delay will be evaluated as

$$D = 0 + 0.5 = 0.5 \text{ slot.}$$

Even at the load '1' the throughput is

$$T = \frac{20}{100} \times 1 + \frac{80}{100} \times 0.998 = 0.9904$$

The total delay will be evaluated as

$$D = 0 + 1.0 = 1.0 \text{ slot.}$$

Considering the case, when 80% of the requests are served locally then only 20% of the requests will be served globally. Even in such a case the throughput will remain same as in the above case. However, the average delay at the local network now will be of 1 slot and delay at the global network will be nearly zero. Therefore, $D = 1.0 + 0 = 1.0$ slot. This clearly suggests that it may possible that sometimes data may available locally but the access time of the local network may be higher than the global network. Hence, a simultaneous request to both local and global network is necessary to minimize the latency (point 3 of the algorithm).

Case III:

In the third case, we retain our assumption that at the local network some requests can be full-filled and therefore all the generated requests will be shared between the local and global network. But here we assume that at the server request cannot be buffered. This is possible when data has to be replicated with any nodal machine which dose not have the functionally of server.

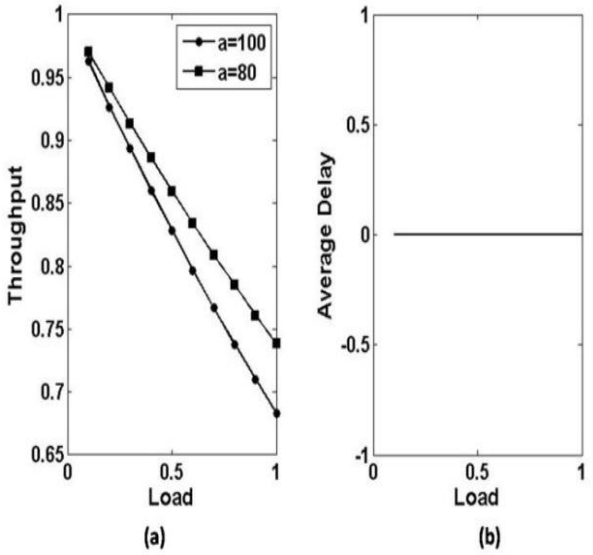

Fig.5: (a) throughput vs. load (b) average delay vs. load with request generating nodes and servers (N) are 4 while considering storage capacity (B=0) is nil.

In figure 5(a), throughput vs. load with request generating nodes are 4 while considering storage capacity (*B*=0) is nil, is plotted. It is observed from the figure, as the buffer decreases, the throughput also decreases. Comparing this figure with figure 3(a), it is clearly observed that on the overall throughput, buffering has deep impact. As in figure, with buffering capacity nil and with *a*=100%, the throughput value

is below 0.68. In the above cases, there is no need to calculate average delay as buffering capacity is zero; the average delay will be zero (Figure 5(b)).

Case IV:

In the fourth case, we assumed that the numbers of servers that can serve the request are either 1 or 4. We also assume that (I*)* at the local network no request can be full-filled; (II) some of the request can be full-filled locally.

In Figure 6, the number of requests are assumed to be 4 while assuming that the number of server are *S*=4 and *S*=1 with *a*=100% and buffering of 4 requests. It is observed form the figure, up to load 0.8, the throughput is nearly one in case of *S*=4, while with *S*=1 it is continuously decreasing and it touches a value of 0.5 at the load of 0.5 which is very less.

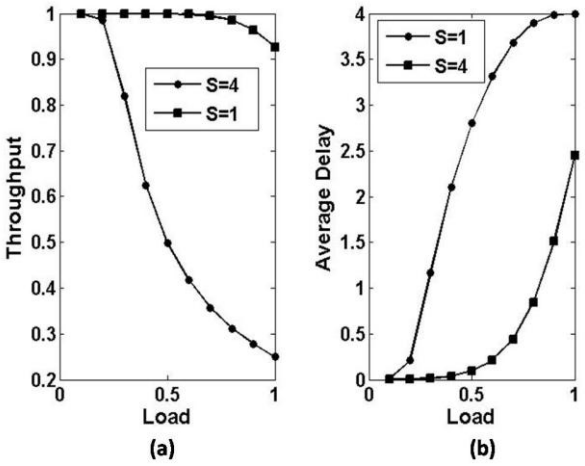

Fig. 6: (a) throughput vs. load and (b) delay vs. load with fixed buffering capacity (B) of 4 while considering request generating nodes (N) are 4 and number of servers as 1 and 4.

As the buffer of only 4 request is allowed with *S*=1 the average delay reaches to 4 even at 0.9, load while with *S*=4 it reaches to a value of 2.5.

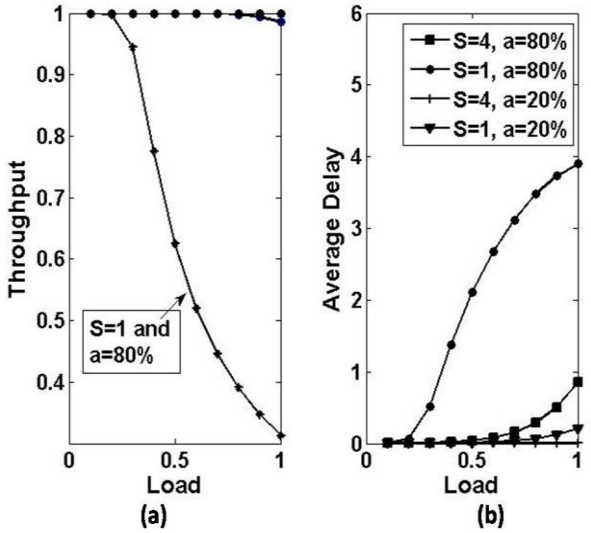

Fig. 7: (a) throughput vs. load and (b) delay vs. load with fixed buffering capacity (B) of 4 while considering request generating nodes (N) are 4 while considering number of servers as 1 and 4 and assuming both local and global services of the requests.

In the figure 7, the numbers of servers are assumed to be 1 and 4 while assuming that the request generating nodes are four with buffering capacity of four requests. In the simulation we considered that the locally serve data is 20% and 80%. It is clear form the figure that with *S*=1 and *a*=80% the throughput is very less, and at the higher load it is nearly zero, while delay is maximum of 4 slots.

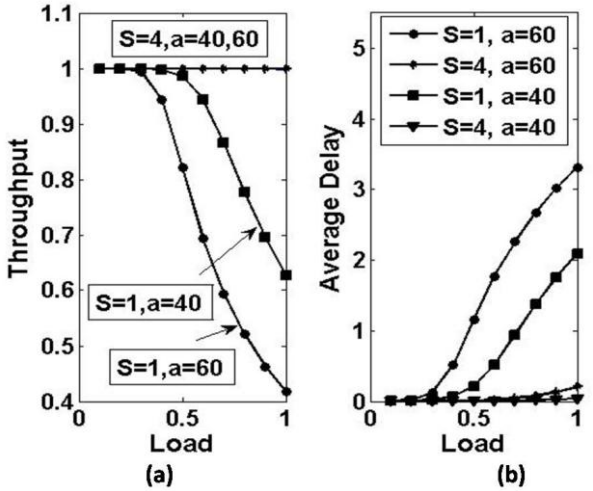

Fig. 8: (a) throughput vs. load and (b) delay vs. load with fixed buffering capacity (B) of 4 while considering request generating nodes and servers (N) are 4 while considering number of servers as 1 and 4 and assuming *a*=40 and 60%.

As the number of servers increases or load on server decreases the throughput increases and average delay also decreases. Comparing the case of *S*=1 and *S*=4 with *a*=80%, the delay in case of *S*=1 is 4 while for *S*=4 it is of 1 slot.

In the figure 8, results are generated with varying number of servers. The numbers of servers are assumed to be 1 and 4 while assuming that the request generating nodes are four. In the simulation we considered that the local serve data is 40% and 60% of the total generated data. It is evident form the figure, as the number of server that can serve a particular request decreases throughput decreases and average delay increases. Considering the case of (*S*=1 and *a*=40%) and (*S*=1 and *a*=60%), it is evident that throughput decreases form 0.6 to 0.4 while average delay increases by one unit. While considering that the available servers are four in numbers the throughput is one at all the loading conditions and average delay is nearly zero. This clearly indicate that to keep throughput at very higher level of load (~ 1) and average delay at zero level the replicated data should be available to comparatively large number of servers.

Case V:

When data request traverse through the network, the network delay may be significant, and it becomes an important delay parameter. If we incorporate the network delay ($D_{N/W}$), then the total delay will be formulated as in other words the total delay can be evaluated as

$$D = D_{av}^{L} + D_{av}^{G} + D_{N/W} \qquad (9)$$

Here, $D_{N/W}$ is the round trip delay.

Now as the more data centric applications are coming up, the whole computer network system is slowly transferring in fiber optic network. Consider the fiber optical network the latency in the network would be

$$D_{N/W} = \frac{L}{v} = \frac{Ln}{c} = \frac{100 \times 1000 \times 2}{3 \times 10^8}$$

$$D_{N/W} = 0.67 \times 10^{-3} \approx 1ms$$

Let the global node is 1000 Km away from the request generating node, then there will be a network delay of 10 ms and thus the round trip delay would be 20 ms. Similarly, if a local node is only 100 m away from the request generating node then the round trip network delay would be $2\mu s$ .It introduced latency that is proportional to the size of the frame being transmitted and inversely proportional to the bit rate as follows:

$$D_F = \frac{F_S}{B_R}$$

In the above equation, F_S is the frame size and B_R is the bit rate. For a frame of 64 bytes and data rate of 100 Mbps the delay is $0.5\mu s$. As average queuing delay is in terms of frame size, for example a delay of 4 slots will be equal to $0.5 \times 4 = 2.0\mu s$.

In case of global network, the main contribution in the delay is due to the propagation delay. Considering the case, when all the generated requests are transferred to global network (refer figure 4) is

$$D = D_{av}^{G} + D_{N/W} = 0.5 \times 5\mu s + 20ms \approx 20ms.$$

Again considering the case, when all the generated requests are served at local network (refer figure 4) is

$$D = D_{av}^{L} + D_{N/W} = 0.5 \times 5\mu s + 2\mu s = 4.5\mu s .$$

Considering, the case, when data servers are employed separately in the network for data replication. As these data server will be fixed in numbers and they will have large buffering capacity. Considering the case of buffering of 100 requests, here the throughput will be very high and delay will also be larger. Assuming a delay of 50 slots, then the total delay would be

$$D = 0.5\mu s \times 50 + 20ms = 20.025ms.$$

Overall, it can be observed that, on the overall delay the network delay plays very significant role in case of global network. However, in case of local network it can't be neglected as it still plays a role in overall delay.

4. CONCLUSIONS

In this paper, a database replication based algorithm is presented. The main idea behind the algorithm is to reduce the network latency in WAN. Simulation results are presented to obtain the mean waiting time and throughput for a database replication algorithm. For the in-depth analysis of the algorithm various cases are considered and it has been found that the storage capacity has deep impact on the throughput and average delay. If server load is below 80% then, nearly 100% throughput is possible with very small average delay (in slots), even at the higher load the throughput is very acceptable. It is also found that if the numbers of servers that can serve the request are larger, then throughput is very high and average delay is very less. The overall, throughput and average delay also depends heavily on the load and if load is comparatively less (0.8) then the throughput is very high and average delay is nearly zero.

5. REFERENCES

[1] Database Replication - *Oracle Documentatio* docs.oracle.com/cd/F49540_01/DOC/server.815/a67781/c31repli.htm

[2] Oracle Database 11g: Oracle Streams (Technical White Paper)www.oracle.com/technetwork/database/twp-*streams-11gr1-134658.pdf*

[3] Introduction to Advanced Replication – Oracle Documentationdocs. Oracle.com/cd/B19306_01/server.102/b14226/repoverview.htm

[4] Design Guide – Sybase infocenter.sybase.com/help/topic/com.sybase.infocenter.../design.pdf

[5] Practical Wide-Area Database Replication1 1. Introduction Y. Amir ww.cnds.jhu.edu/pub/papers/cnds-2002-1.ps

[6] Bettina Kemme, Gustavo Alonso. 2010. Database Replication: A Tale of Research across Communities. VLDB, Vol.3, No.1.

[7] A. Dogan, 2009. A study on performance of dynamic file replication algorithms for real-time file access in data grids, Future Generation Computer Systems 25 (8): 829–839 .

[8] R. S. Chang, P. H. Chen, 2007. Complete and Fragmented selection and retrieval in data grids, Future Generation Computer Systems, 23 : 536–546.

[9] Marius Cristian MAZILU,2010. "Database Replication", Database Systems Journal , 1(2), 33-38.

[10] Ratnasamy, S.; Karp, B.; Yin, L.; Yu, F.; Estrin, D.; Govindan, R.; Shenker, S. GHT.2002. A Geographic Hash Table for Data-centric Storage. In Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, GA, USA, 28; pp. 78–87.

[11] Wiesmann, Pedone, Schiper, Kemme, Alonso.2000. Understanding Replication in Databases and Distributed Systems. Proceedings of 20th International Conference on Distributed Computing Systems.

[12] Yair Amir, Claudiu Danilov, Michal Miskin-Amir, Jonathan Stanton and Ciprian Tutu. 2002. Practical Wide-Area Database Replication. Technical Report Johns Hopkins University, http://www.cnds.jhu.edu/publications.

[13] Y. Amir. Replication Using Group Communication Over a Partitioned Network. 1995. Ph.D. thesis, The Hebrew University of Jerusalem, Israel. www.cs.jhu.edu/~yairamir

[14] N. Saadat and A.M. Rahmani. 2012. PDDRA: A new pre-fetching based dynamic data replication algorithm in

data grids. Springer: Future Generation Computer Systems, vol 28, pp. 666-681.

[15] Sanjay Kumar Yadav, Gurmit Singh, Divakar Singh Yadav.2013. Mathematical Framework for A Novel Database Replication Algorithm. International Journal of Modern Education and Computer Science (IJMECS), vol.5, no.9, pp.1-10, DOI: 10.5815.

6. AUTHORS'PROFILES

Sanjay Kumar Yadav: Is Assistant Professor of Computer Science in Dept. of Computer Science & Information Technology at Sam Higginbottom Institute Of Agriculture, Technology & Sciences" (Formerly Allahabad Agricultural Institute), (Deemed-to-be-University) Allahabad. He obtained batchelor degree in B.Sc.(Maths) from University of Allahabad, MCA degree from Institute of Engineering and Technology, Lucknow. M.Tech. in Software Engineering from Motilal Nehru National Institute of Technology Allahabad and pursuing his Ph.D. in Computer Science & IT at Sam Higginbottom Institute Of Agriculture, Technology & Sciences" (Formerly Allahabad Agricultural Institute), (Deemed-to-be-University) Allahabad. His research interest includes distributed system and mobile ad-hoc network.

Prof. Gurmit Singh: is Emeritus Professor of Computer Science in Dept. of Computer Science & Information Technology at Sam Higginbottom Institute Of Agriculture, Technology & Sciences" (Formerly Allahabad Agricultural Institute), (Deemed-to-be-University) (In short form SHIATS)Allahabad. He served the department as professor and Head for several years and retired in year 2012. He was also served the University as Dean, Shepherd School of Engineering & Technology and is on the program committees of the University. He is the author/co-author of several publications in technical journals and conferences. Presently he is serving in the Dept. of Computer Science & Information Technology as Emeritus Professor; his research interest includes distributed system and mobile ad-hoc network, wireless sensor network and evolutionary computing.

Prof. Divakar Singh Yadav: is Professor of Computer Science at Institute of Engineering and Technology, Lucknow. He obtained B.Tech in Computer Science& Engineering, M.Tech in Computer Science from IIT, Kharagpur and Ph.D from University of Southampton, U.K. Before joining Gautam Buddh Technical University, Lucknow as Pro-Vice Chancellor, he was at South Asian University, New Delhi, an international university established by South Asian Association for Regional Cooperation (SAARC) nations, where he was Chairperson of Department of Computer Science at Faculty of Mathematics and Computer Science.

Dr. Yadav possesses more than 20 years of experience in academics/research in India and Abroad. He has long standing academic interests in database systems and distributed computing. His primary research interests are in formal methods, refinement of distributed systems using Event-B, verification of critical properties of business critical systems and reasoning about distributed database systems. He has also participated in prestigious Daghtuhl seminar at Schloss Dagstuhl-Leibniz Center for Informatics, Germany in 2006, in addition to invitation at Commonwealth Scholarship Commission, U.K. seminar held at the University of the West England, Bristol in 2007. Dr. Yadav is author of four (04) books in the area of computers and information technology including best seller 'Foundations of Information Technology' published in 2001. His research contributions in the area of computer science and information technologies appeared in the international journals and refereed conference proceedings published by Springer-Verlag, Elsevier and IEEE.