ISSN:2231-0711
www.ijcset.net

Excogitating File Replication and Consistency maintenance strategies intended for Providing High Performance at low Cost in Peer-to-Peer Networks

Bollimuntha Kishore Babu[#1], Divya Vadlamudi[#2], Movva N.V Kiran Babu[#3] ,M.Hema Madhuri[#4]

[#,1#2,#4]*Department of CSE, K L University, Andhrapradesh.*
[#3]*Department of CSE, Mother Theresa Institute of Science &Technology, Sathupally,AP.*

[#1]bkishore002@hotmail.com,
,[#2]divya.movva@kluniversity.in,
[#3]kiranbabuonline@yahoo.co.in,
[#4]metlamadhuri@gmail.com

***Abstract*— P2P is a trendy technology used for file sharing. File replication and Consistency maintenance are the methods used in P2P for elevated system performance. File replication methods indicate replica nodes without thinking about consistency maintenance which may lead to high overhead for redundant file replications and consistency maintenance. Consistency maintenance methods update files without considering file replication dynamism which may not give the accuracy of replica consistency. Hence there is a need to think about consistency maintenance while file replication to achieve high performance and high availability. When data files are replicated at many nodes, consistency must be maintained among the nodes. In this paper we point out different replication strategies that are applied P2P systems, followed by consistency maintenance techniques intended for high performance and high availability of data. Finally we explore a combined method of file replication and consistency maintenance.**

***Keywords*— peer to peer system, file replication, Consistency maintenance.**

I. Introduction

The term P2P refers to "peer-to-peer" networking. A peer-to peer network allows computer hardware and software to function without the need for special server devices. P2P is an alternative to client-server network design. In client-server network, each computer or process on the network is either a client or a server. Servers are powerful computers or processes dedicated to managing disk drives (file servers), printers (print servers), or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients rely on servers for resources, such as files, devices, and even processing power. In peer-to-peer network, each workstation has equivalent capabilities and responsibilities. This differs from client/server architectures, in which some computers are dedicated to serving the others. With increase in popularity of Peer to Peer (P2P) networks it has also become one of the medium for spreading of viruses, spywares, ad ware, and malware through file sharing applications. Some of the P2P file sharing programs available on internet are bittorrent, limeware, kazaa, shareaza, imesh, bearshare lite, kceasy, ares galaxy, emule, soulseek, winmx, piolet etc Most of the people download audio and video files by using P2P file sharing. Whenever a file is requested frequently, the capacity of the node degrades and gives delayed response. File replication is very useful in this situation. In this method, the load is distributed over replica nodes. File consistency maintenance is to maintain consistency between file and its replica nodes. This paper discuss about different strategies to achieve high efficiency in file replication and consistency maintenance at a lower cost. A sample P2P network is shown in Figure 1.

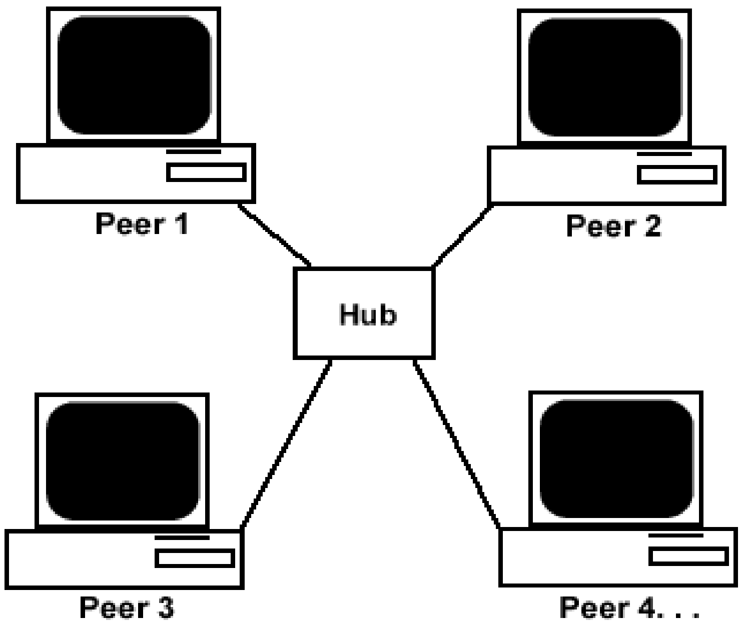

Figure 1: A sample p2p network

In file replication and consistency maintenance methods, nodes accept replicas and update messages. They are unable to keep track the utilization of replicas to determine the need of file replicas and replica updates. Minimization of the number of replicas helps to reduce unnecessary updates in consistency maintenance. Here the numbers of replicas are based on queries. In the next section we discuss different replication strategies that are applied to P2P systems.

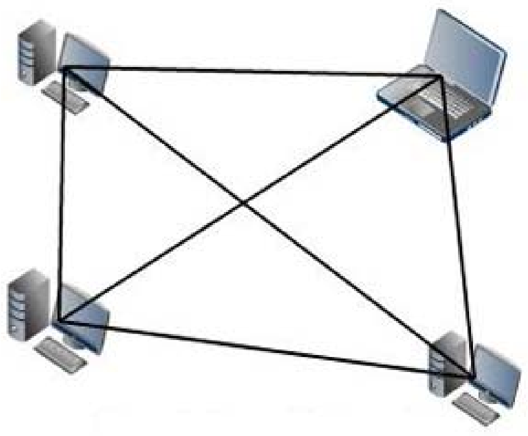

Figure 2: P2P network functional diagram

II. Replication Strategies

One way to improve the performance of a system is to replicate data files on several nodes, before a query is resolved. In this section, we survey various replication techniques that are applied to p2p systems. Initially, specific replication strategies that indicate how replicas are distributed across the nodes are proposed in [2]. Uniform, proportional and square-root replications are examples of the above strategies. In uniform replication strategy all data files are replicated at the same number of nodes, even though some data files are more frequently requested than others. Using this technique, the required maximum search cost is minimized. An alternative strategy is proportional replication. Here, the number of replicas for a specific data file is proportional to the query probability of the data file. So, if nodes store only the data files that are requested for, the replication distribution is almost proportional to the query distribution. Although queries for popular data files are satisfied efficiently because there are many replicas for the requested data files across the network, queries for unpopular data files require higher search cost. Between uniform and proportional replication is square-root replication. In this strategy, the replicas of a specific data file are proportional to the square root of its query probability. Square-root replication provides a balance for searching popular and unpopular data files.

A. REPLICATION ON STRUCTURED SYSTEMS

Additionally, structured p2p systems use specific methods to improve their performance and to increase their availability. When data files are replicated, the load of the system is balanced and usually there are copies nearby the requestor. Also, the availability is higher since we can use replicas in the case of failures and nodes departures. On the other hand, the amount of storage increases and we must maintain the consistency of the replicated data. In this section, we introduce such methods that are applied to Chord[3] and CAN[4].

Chord: A strategy for metadata replication that is used in Chord is based on successor lists. With a successor list a node maintains information about the next nearest successors on the ring. This list guarantees the correctness of a search.

CAN: In Content-Addressable Network (CAN) a replication technique is based on realities, which are independent coordinate spaces. Each node is assigned to a zone in each reality. Thus, if CAN has r realities, a node is assigned to r zones, one for each reality. Replicas of the hash table are stored in each reality. In this way, when there are multiple realities, a pointer for a specific data file is stored at more than one different node. In order to improve data availability, we can also use k different hash functions to map a key onto k points in the coordinate space, and so replicas of the (key, value) pair are placed to k different nodes in the system. In this case, the (key, value) pair is not available only when all k replicas are not available at the same time. Furthermore, replication is also used in the overloading coordinate zones technique. According to this technique, multiple nodes may share a zone. So, replicas of the hash table are placed to all nodes that have been assigned to the same zone, ensuring higher availability. In general, when a node conceives that receives many requests for a specific data key, it may replicate this data key at each of its neighbors. A node that holds a replica can be used to satisfy related requests, reducing the load of the node that holds the 'original' data. A particular kind of replication is caching. A node can maintain a cache of data keys that are recently accessed. So, it first checks its own cache in order to find the requested data key. Only if the data key is not found, the request is forwarded to other nodes**.**

B. **REPLICATION ON UNSTRUCTURED SYSTEMS**

Moreover in [2],[5] authors present replication techniques for unstructured p2p systems. The first one is called owner replication. When a search is successful, the desirable data file is replicated to the node that requests for it. This technique is used in Gnutella. Alternatively, in path replication, when a search is successful, the desirable data file is replicated to all nodes along the query path, i.e. the path from the node that asks for the data file to the node that provides it. This technique is used in Freenet [6] and in specific circumstances may decrease the system's performance. In a different approach, the idea of random walks is used. So, in random replication we count the number of nodes on a query path, say p, and we select randomly p of the nodes that the walks visited to replicate the data file. This technique seems to be harder to be implemented.

III. CONSISTENCY MAINTENANCE

When data files are replicated at many nodes, consistency must be maintained among the nodes. In general, we can separate replication into eager and lazy methods. Eager replication keeps all replicas synchronized at all nodes, by updating all replicas in a single transaction. In reverse, lazy replication propagates asynchronously replicas' updates to other nodes after replicating transaction commits. Most times, p2p systems use lazy replication because of its lower cost.

Furthermore, in [7], several strategies for spreading updates are proposed. The following strategies are typical examples of epidemic algorithms.

Direct mail: When an update occurs, it is immediately mailed from its originating node, i.e. the node where the update occurs, to all other nodes. The main advantage of this strategy is that updates are propagated very quickly.

Anti-entropy: Periodically, every node selects randomly another node and resolves any differences between them, by exchanging content. There are three ways to execute anti-entropy, called push, pull and push-pull. In push method, when an update occurs, the originating node propagates an update message to all nodes that hold replicas of the updated data File. In pull method, all nodes that hold replicas, ask the 'primary' node for updates. The last method, named push-pull, is a combination of the others. The anti-entropy strategy is reliable, but quite slow.

Rumor mongering: When a node receives a new update, it periodically selects randomly another node and checks if this node has seen the update, in order to send it to it. A node stops to send the update to other nodes, when many other nodes have seen it.

Moreover in [8], is proposed an update strategy, which is based on a hybrid push/pull rumor spreading algorithm. Nodes are many times offline. When these nodes are connected again, they must be informed about the updates that they have missed. This update scheme has two phases: the push and the pull one. The node where the update occurred, initiates the push phase. The node propagates the new update to a subset of nodes that hold a corresponding replica.They propagates, in turn, the update to another subset of nodes that they have not been updated yet, and so on. This process is similar to flooding method with constrains, because it is executed for a specific number of steps. Furthermore, it avoids many duplicate messages, while propagating the rumor. On the other hand, the pull phase is initiated either by a node that has been offline and then gets connected and needs to update its replicas or by a node that does not receive updates for some time or by a node that receives a pull request and is not sure that it has the freshest replica. The above hybrid spreading algorithm provides probabilistic guarantees for acceptable results for queries and results no strict consistency.

The file replication methods copies files near file owners, file requesters or along a query path from a requester to a owner. PAST [11], CFS [12], and Backslash [13] replicate each file on close nodes near the file's owner. In LAR [14] and Gnutella [15], overloaded nodes replicate a file at requesters. In these methods, file owners rigidly determine replica nodes and nodes accept replicas. They are unable to keep track replica utilization to reduce underutilized replicas and ensure high utilization of existing replicas. In efficient and adaptive decentralized file replication algorithm in P2P file sharing systems called EAD [9], traffic hubs that carry more query load are chosen as replica nodes. The nodes continuously check their query load in order to create copy for the file and remove low utilized replicas. Replication in a structured P2P system is to decrease file query time, while replication in an unstructured P2P system is to decrease the search time. File consistency methods are based on structure [16] and message spreading [17].In structure based methods, stable replica nodes are used but it is not true in practice because of file replication dynamism. In message spreading, unnecessary and redundant messages are generated and is not sure that all replicas receive update messages. Therefore the methods lead to unnecessary file replications and overhead in consistency maintenance.

IV. COMBINED APPROACH FOR REPLICATION AND CONSISTENCY MAINTENANCE

In this section we present an analysis of a combined approach [10] for File replication and consistency maintenance. This approach is a combination of both file replication and consistency maintenance. Both are dependent on each other. Instead of accepting replicas and update messages, it integrates file replication and consistency maintenance by letting each node autonomously determine the need for file replication and update based on file query rate and update rates. File replication places replicas in frequently visited nodes to guarantee high utilization of replicas, and meanwhile reduce underutilized replicas and overhead of consistency maintenance. It was illustrated in Figure 3.

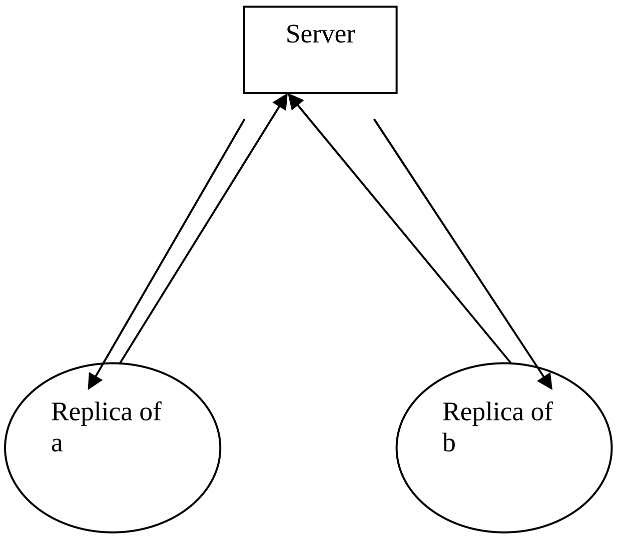

Figure 3: Illustration of combined approach

In the above figure, the straight line represents the link between replica node and server and the arrow mark represents that the replica polls the server for update, to make sure that an update file is available to the client. Consistency maintenance aims to guarantee file fidelity of consistency at a low cost with file replication dynamism consideration. Using adaptive polling, this ensures timely update operation and avoids unnecessary updates. The basic idea of this approach is to use file query and update rate to direct file replication and consistency maintenance. Combined approach of File Replication and Consistency maintenance mechanism is developed by using EAD [9] file replication algorithm. This algorithm achieves an optimized trade-off between query efficiency and overhead in file replication. The combined approach has a time-to-refresh (TTR) value with each replica node of a file. It denotes at what time the replica should poll the file owner to keep its replica updated. a node should poll the owner to keep its replica updated. The TTR value is changed frequently based on the results of each polling. It takes file query rate for poll time determination. TTR query and TTR poll denotes the next time, where the file is updated. IRM polling algorithm uses Time To Refresh value(TTR) to represent file change frequency. When TTR <= TTRquery, that is, when the file change rate is higher than the file query rate, there is no need to update the replica at the rate of file change rate. This is because the ultimate goal of consistency maintenance is to guarantee the received file is up to state. If a replica is updated soon after its original file is changed but there is no query for this replica until after the next update, it is a waste to update the file this time. For example, a file changes for every 1 second but it is visited for every 2 seconds by client, then updating replica once every 2 seconds can guarantee that the response file from replica node is the updated file. The replication dynamism is shown in Figure 4.

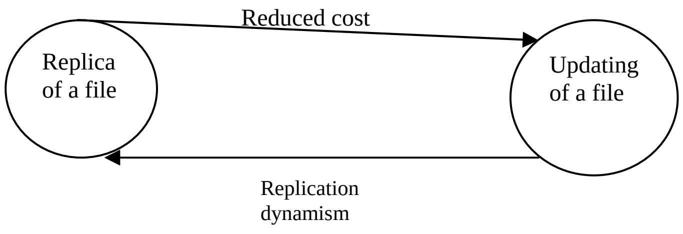

Figure 4: Interrelationship between replication and consistency maintenance

This section presented a mechanism which integrates File Replication and Consistency Maintenance to achieve high efficiency in file replication and consistency maintenance at a lower cost. Replication dynamism deals with replica node generation, deletion and failures.

CONCLUSIONS

In this paper, we addressed large-scale P2P collaborative applications in which shared data are distributed across peers in the network. Since these peers can join and leave at any time, data replication is required to provide high availability and we analyzed a combined approach for file replication and consistency maintenance which is highly efficient at low cost. Finally we conclude that the replication solution must satisfy the requirements like, data type independency, high level of autonomy and eventual consistency.

ACKNOWLEDGMENT

We like to express our gratitude to all those who gave us the possibility to carry out the paper. We would like to thank Mr.K.Satyanarayana, chancellor of K.L.University, Dr.K.Raja Sekhara Rao, Dean, and K.L.University for stimulating suggestions and encouragement. We have further more to thank Prof.S.Venkateswarlu, Dr.K.Subrahmanyam, who encouraged us to go ahead with this paper.

REFERENCES

[1] Haiying (Helen) Shen,"IRM: Integrated File Replication and Consistency Maintenance in P2P Systems", Parallel and Distributed Systems, IEEE Transactions on Jan 2010

[2] E. Cohen and S. Shenker. Replication Strategies in Unstructured Peer to-Peer Networks. In Proceedings of the ACM SIGCOMM'02 Conference, 2002.

[3] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In Proceedings of the ACM SIGCOMM'01 Conference, 2001.

[4] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In Proceedings of the ACM SIGCOMM'01 Conference, 2001.

[5] Q. Lv, P.Cao, E.Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In Proceedings of the ACM ICS'02 Conference, 2002.

[6] Freenet. http://freenetproject.org.

[7] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic Algorithms for Replicated Database Maintenance. *PODC*, 1987.

[8] A. Datta, M. Hauswirth, and K. Aberer. Updates in Highly Unreliable, Replicated Peer-to-Peer Systems. In Proceedings of the 23rd International Conference on Distributed Computing Systems, 2003.

[9] H. Shen, "EAD: An Efficient and Adaptive Decentralized File Replication Algorithm in P2P File Sharing Systems," Proc. Eighth Int'l Conf. Peer-to-Peer Computing (P2P '08), 2008.

[10] K. Shalini, Y. Surekha," An Algorithm for Consistency Maintenance in P2P systems ", (IJAEST) International Journal of Advanced Engineering Sciences and Technologies Vol No. 9, Issue No. 1, 097 – 100.

[11] A. Rowstron and P. Druschel, "Storage Management and Caching in PAST, a Large-Scale, Persistent Peer-to-Peer Storage Utility," Proc. ACM Symp. Operating Systems Principles (SOSP), 2001.

[12] F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, and I. Stocia,"Wide Area Cooperative Storage with CFS," Proc. ACM Symp. Operating Systems Principles (SOSP), 2001.

[13] T. Stading, P. Maniatis, and M. Baker, "Peer-to-Peer Caching Schemes to Address Flash Crowds," Proc.

First Int’l Workshop Peerto- Peer Systems (IPTPS), 2002.

[14] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, and P. Keleher, “Adaptive Replication in Peer-to-Peer Systems,” Proc. 24th Int’l Conf. Distributed Computing Systems (ICDCS), 2004.

[15] Gnutella Home Page, http://www.gnutella.com, 2008.

[16] S. Tewari and L. Kleinrock, “Analysis of Search and Replication in Unstructured Peer-to-Peer Networks,” Proc. ACM SIGMETRICS, 2005.

[17] G. Xie, Z. Li, and Z. Li, “Efficient and Scalable Consistency Maintenance for Heterogeneous Peer-to-Peer Systems,” IEEE Trans. Parallel and Distributed Systems, vol. 19, no. 12, pp. 1695-1708, Dec2008