International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 7, September - 2012

Establishing Replica Consistency Maintenance Using Adaptive Polling Method In P2P Systems

Mr. R. Rakesh Kumar Jadav
Mtech [computer sc. engineering]
Aurora's Technological And Research Institute
Hyderabad, India

Mr. I. Naveen
Associate professor
Aurora's Technological And Research Institute
Hyderabad, India

Abstract-This paper presents two techniques File Replication and Consistency Maintenance in P2P systems for high system performance. File replication methods rigidly specify replica nodes without consideration of consistency maintenance which may lead to unnecessary file replications and consistency maintenance overhead. On the other hand, consistency maintenance methods update files without considering file replication dynamism which may not give the accuracy of replica consistency. Passively accepting replicas, update messages and each node actively relying on polling file owners based on query rate and update rates still cannot guarantee that all file requesters receive up-to-date files. So, we develop a mechanism which combines both file replication and consistency maintenance using adaptive polling method to fully exploit file popularity and update rate for efficient and effective replica consistency maintenance.

Keywords: File replication, consistency maintenance, adaptive polling, peer to peer.

I. INTRODUCTION

The term P2P refers in networks 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, malware through file sharing applications. Some of the P2P file sharing programs available on internet are bit torrent, lime ware, 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 presents 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.

The past few years have seen a dramatic increase in the popularity and use of peer-to-peer (P2P) file sharing networks. Current P2P systems are specifically designed to share static content such as music and video files. The utility of P2P systems goes beyond sharing of static files—future P2P applications (e.g., collaborative P2P applications) can be expected to share dynamic files. In such applications, shared files are not necessarily static; files may be modified and updated during their lifetime. To handle such dynamic content, P2P networks must evolve from a predominantly read-only system to one where files can be both read and written. Since files may be widely replicated in a P2P system, handling dynamic files requires consistency techniques to ensure that all replicas of a file are temporally consistent with one another.

Peer-to-peer overlay networks are widely used in distributed systems. P2P networks can be divided into two categories: structured peer-to-peer networks in which peers are connected by a regular topology, and unstructured peer-to-peer networks in which the topology is arbitrary. The objective of this work is to design a hybrid peer-to-peer system for distributed data sharing which combines the advantages of both types of peer-to-peer networks and minimizes their disadvantages. Consistency maintenance is propagating the updates from a primary file to its replica. Adaptive consistency maintenance algorithm (ACMA) maintains that periodically polls the file owner to update the file due to minimum number of replicas consistency overhead is very low.

P2P systems are known to be highly dynamic—peers can dynamically join and leave the net-work at any time and the mean session duration of a peer is only a few hours. Since peers containing replicated content may not be part of the network when a file is modified, maintaining consistency is more challenging in these environments.

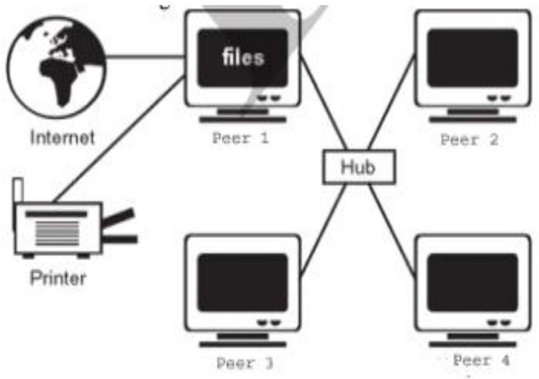

Fig .1. Inter connection of peers in a network

II. RELATED WORK

In the PAST[2] system, storage nodes and files are each assigned uniformly distributed identifiers, and replicas of a file are stored at nodes whose identifier matches most closely the file's identifier. This statistical assignment of files to storage nodes approximately balances the number of files stored on each node. However, non-uniform storage node capacities and file sizes require more explicit storage load balancing to permit graceful behavior under high global storage utilization; likewise, non-uniform popularity of files requires caching to minimize fetch distance and to balance the query load.

The Cooperative File System (CFS)[3] is a new peer-to-peer readonly storage system that provides provable guarantees for the efficiency, robustness, and load-balance of file storage and retrieval. CFS does this with a completely decentralized architecture that can scale to large systems. CFS servers provide a distributed hash table (DHash) for block storage. CFS clients interpret DHash blocks as a file system. DHash distributes and caches blocks at a fine granularity to achieve load balance, uses replication for robustness, and decreases latency with server selection. DHash finds blocks using the Chord location protocol, which operates in time logarithmic in the number of servers.

Backslash [4], a collaborative web mirroring system run by a collective of web sites that wish to protect themselves from flash crowds. Backslash is built on a distributed hash table overlay and uses the structure of the overlay to cache aggressively a resource that experiences an uncharacteristically high request load. By redirecting requests for that resource uniformly to the created caches.

While current Peer-to-Peer (P2P) systems facilitate static file sharing, newly-developed applications demand that P2P systems be able to manage dynamically-changing files. Maintaining consistency between frequently-updated files and their replicas is a fundamental reliability requirement for a P2P system. Scalable Consistency Maintenance in Structured P2P Systems [SCOPE], a structured P2P system supporting consistency among a large number of replicas. By building a replica-partition-tree (RPT) for each key, SCOPE keeps track of the locations of replicas and then propagates update notifications. Theoretical analyses and experimental results demonstrate that SCOPE can effectively maintain replica consistency while preventing hot-spot and node-failure problems. Its efficiency in maintenance and failure-recovery is particularly attractive to the deployment of large-scale P2P systems.

The file replication methods copies files near file owners , file requesters or along a query path from a requester to a owner. PAST [2], CFS [3], and Backslash [4] replicate each file on close nodes near the file's owner. In LAR [5] and Gnutella [6], 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 [8] , 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 [7] and message spreading [9].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. 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.

III. PROPOSED SYSTEM

A. *Adaptive polling:*

Basically, each node actively decides to create or delete a replica and to poll for update based on file query and update rates in a totally decentralized and autonomous manner. It replicates highly queried files and polls at a high frequency for frequently updated and queried files.

Dynamically vary the polling frequency based on the update rate for the file . A peer can observe that rate of updates to a file and poll more frequently when the file is being modified frequently and less frequently when it is not. Instead of using a history of modification times, a simpler approach is to vary the poll frequency based only on the result of the most recent poll: the frequency of polls is reduced if the file was not modified since the last poll and made more aggressive if the file was modified.

Due to this**,** it avoids unnecessary file replications and updates, it improves replica utilization, file query efficiency, and consistency fidelity, It enhances the guarantee of file consistency. It offers the flexibility to use different replica update rate to cater to different consistency requirements determined by the nature of files and user needs, it ensures high possibility of up-to-date file responses.

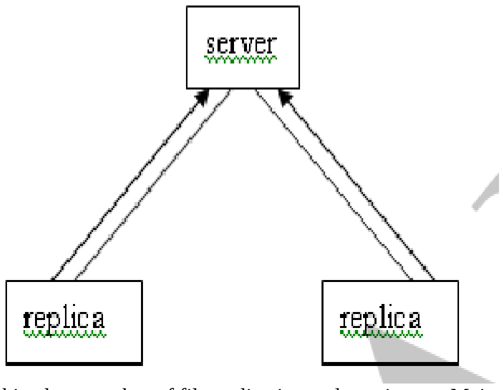

Fig2. Combined approaches of file replication and consistency Maintenance

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.

B. *File replication*

Combined approach of File Replication and Consistency maintenance mechanism is developed by using EAD [8] file replication algorithm. This algorithm achieves an optimized trade-off between query efficiency and overhead in file replication. File replication component addresses two main problems 1) The point at which the replicas should be generated , and are not underutilized. 2) To remove underutilized and unnecessary replicas so that the overhead for consistency maintenance is less.

a) *The need to determine replica nodes*

Whenever a popular file like audio and video is accessed continuously, the server will be degraded and will give delayed response .In that situation placing a replica for that file is suitable.

b) *Creation of replica for popular file*

Creation of a replica is based on file query rate. A replica is created when the requesters request continuously a file. If the query rate is less to that Replica, then the replica node is deleted

c) *Adaptation of Replica node*

A file might not be accessed frequently in a replica node, so the replica nodes should maintain and update their query and passing rate.Hence, underutilized replica nodes can be identified, this helps in removal of a replica.

C. *File Consistency maintenance*

To maintain consistency between frequently and not frequently updated files and its copies is an important aspect in peer-peer file sharing systems. Here the replica node might be created or failed or deleted. To be aware of this we employ a polling method, in which the replica node itself polls the server for update continuously. In our proposed system, the replica node frequently polls the file owner for update. It is based on two main problems 1) In what frequency, the replica node polls the sever for update. 2) To optimize the polling operations to reduce the cost of polling operations on server and to maintain accuracy in consistency maintenance.

D. *Algorithm for file consistency maintenance*

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.

```
1. If a query is requested for a file then
     Include an poll request within query of the file
      Else send the update request.
2. If the acceptance reply is given from the owner of file {
          / /check the conditions
3. If the file is a valid one then
          TTR = TTRold+ α.  // α is a constant
     If the file is a stale one then {
          TTR = TTRold + β.  // β is a constant. We need to
   update replica of a file. }
 4. If Time to refresh rate (TTR) is greater than maximum or
   less then minimum TTR then
          TTR = max (TTRmin, min(TTR max, TTR))
 5. If time to refresh rate is less than or equal to query
          Then  TTRpoll = Tquery
          Else
 6. TTRpoll = TTR }
```

when TTR > Tquery , that is, the file is queried at a higher rate than change rate, then the file should be updated timely based on TTR. As a result, TTRpoll should be calculated based on the following formula

TTRpoll ={Tquery   TTR <=Tquery;
TTR TTR>Tquery:}

Similar to file replication requests, replica nodes try to include all polling messages along with queries in order to save the polling overhead. Algorithm shows the pseudo-code of the adaptive file consistency maintenance algorithm. As a result, the number of polls and the overhead for file updates are reduced without compromising the file fidelity of consistency; that is, there is low possibility that a file is outdated when visited.

IV. CONCLUSION

This paper proposes the combined approach of file replication and consistency maintenance which is highly efficient at low cost. Instead of accepting replicas and updates, nodes determine the need for replicas based on file query rate and update rate. Hence this mechanism which enhances both file replication and consistency maintenance using adaptive polling method to fully exploit file popularity and update rate for efficient and effective replica consistency maintenance.

This approach guarantees high utilization of replicas, high query efficiency and accurate consistency maintenance. It reduces redundant file replicas, consistency maintenance overhead, and unnecessary file updates. Replica node polls the file owner , this might not be true that all the requesters of the file can have up-to-date files. Although its performance is better when compared to other file consistency maintenance algorithms.

V. 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] 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.
[3] 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.
[4] 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.
[5] 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.
[6] Gnutella Home Page, http://www.gnutella.com, 2008.
[7] S. Tewari and L. Kleinrock, "Analysis of Search and Replication in Unstructured Peer-to-Peer Networks," Proc. ACM SIGMETRICS, 2005.
[8] 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.
[9] 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, Dec. 2008
[10] M. Theimer and M. Jones, "Overlook: Scalable Name Service on an Overlay Network," Proc. 22nd Int'l Conf. Distributed Computing Systems (ICDCS), 2002.
[11] K. Huang, T. Huang, and J. Chou, "LessLog: A Logless File Replication Algorithm for Peer-to-Peer Distributed Systems," Proc. 18th Int'l Parallel and Distributed Processing Symp. (IPDPS), 2004.
[12] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "OceanStore: An Architecture for Global-Scale Persistent Storage," Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems, 2000.
[13] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and Replication in Unstructured Peer-to-Peer Networks," Proc. 16th Int'l Conf. Supercomputing (ICS), 2001.
[14] E. Cohen and S. Shenker, "Replication Strategies in Unstructured Peer-to-Peer Networks," Proc. ACM SIGCOMM, 2002.
[15] S. Tewari and L. Kleinrock, "Analysis of Search and Replication in Unstructured Peer-to-Peer Networks," Proc. ACM SIGMETRICS, 2005.
[16] S. Tewari and L. Kleinrock, "On Fairness, Optimal Download Performance and Proportional Replication in Peer-to-Peer Net-works," Proc. IFIP Networking, 2005.
[17] S. Tewari and L. Kleinrock, "Proportional Replication in Peer-to-Peer Network," Proc. IEEE INFOCOM, 2006.
[18] D. Rubenstein and S. Sahu, "Can Unstructured P2P Protocols Survive Flash Crowds?" IEEE/ACM Trans. Networking, vol. 13, no. 3, pp. 501-512, June 2005.
[19] Q. Yang, W. Xiao, and J. Ren, "PRINS: Optimizing Performance of Reliable Internet Storages," Proc. 26th Int'l Conf. Distributed Computing Systems (ICDCS), p. 32, 2006.
[20] D. Tsoumakos and N. Roussopoulos, "APRE: An Adaptive Replication Scheme for Unstructured Overlays," Proc. 14th Int'l Conf. Cooperative Information Systems (CoopIS), 2006.
[21] M. Raunak, P. Shenoy, B. Urgaonkar, A. Ninan, and K. Ramamritham, "Maintaining Mutual Consistency for Cached Web Objects," Proc. 21st Int'l Conf. Distributed Computing Systems (ICDCS), 2001.

VI. AUTHOR BIOGRAPHIES

Mr. R.Rakesh Kumar Jadav received his B.Tech in Computer science and Engineering from cvr college of engineering, JNTU, Hyderabad and Pursuing M.Tech in Computer science and Engineering from Aurora's Technological And Research Institute, JNTU, Hyderabad.

I.NAVEEN, B.TECH from JNTU, M.TECH from Visveswaraya Technological University, Karnataka. Registered for PhD in University of Hyderabad. Areas of interest: Computer Networks, Mobile Computing, Adhoc Networks and Autonomic Computing. my research topic is Trust Computing in Autonomic Systems.