ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

Improving P2P Performance by Applying Proper Replica Placement Strategies

Thara R J, Kala Karun A, Shine S

***Abstract*—One among the several emerging distributed computing systems is Peer-to-Peer (P2P) systems that provide a platform for large internet scale distributed applications, with no dedicated infrastructure, develops distributed applications by dividing them across individual systems termed peers which takes the role of both client as well as servers. Popularity of objects in such a system can change rapidly, which demands the need for a rapid and light weight content replication strategy that takes this dynamic popularity changes into consideration. While considering P2P in distributed file sharing applications, data availability in a P2P system, which is highly related to how users select peers to replicate data, has significant impact on system's performance. In addition to optimized availability, the proposed solution should guarantee reduced search and data access latency in worst case. The proposed solution should dynamically adaptable to instantaneous query arrival rate as well as dynamic membership of individual peers. It should provide good performance with reduced number of control messages. Thus an intelligent placement of replicas considering various factors of the system always out perform a random placement of replicas on random peers. This paper discuss various such RPA proposed for structured as well as unstructured P2P and finally compare them and summarized the major outcomes. This paper can enable the readers to choose appropriate replication strategy for their applications.**

***Index Terms*— Peer-to-Peer (P2P) systems, structured P2P, unstructured P2P, replica placement algorithms**

I. INTRODUCTION

Distributed computing is an emerging area in the computer world, which can handle data intensive as well as processing intensive internet scale applications in an efficient manner. While modeling a problem based on distributed computing system, a number of challenges have to be discussed. One such challenge is fault tolerance, i.e. the ability of the system to tolerate independent failures. Replication is key, to effectiveness of a distributed system in that it provides enhanced performance, fault tolerance and high availability. The system may suffer from server failures, network partitioning and disconnected operations; irrespective of all such problems the system should be able to provide services to its clients, i.e. it needs to be highly available as well as fault tolerant, so that the over performance get increased. Efficient replication plays a major role in achieving such a performance. Now to get maximum benefit out of replication, a strategic placement of replicas considering various factors is needed. The replica placement problem discussing here has been solved by a number of techniques, each aims at different goals, optimizes various aspects of the system. The problem is treated differently in each of the distributed computing systems like grid networks, peer-to-peer (P2P) systems, cloud network, etc. Out of them some popular proposals for the P2P systems have been discussed and compared throughout this paper.

P2P systems as its name implies represent a network of peers connected by communication channel, with each peer takes the role of client as well as servers of a client-server model. As that of any other distributed computing systems, they follow a divide and conquer approach for solving distributed applications, where large problem is divided into small sub problems and assigned to different peers so that they can solve the sub problems independently and finally combines the results back to obtain the solution to original problem. P2P systems have proved their efficiency on large scale decentralized file sharing applications over internet. The scaling of the P2P file sharing system without any central control is achieved through their overlay network structuring, which makes the file sharing applications to scale up automatically with increasing number of peers. They are able to adapt the arrival and departure of nodes with relatively low cost, and hence they are robust and self organizing.

The attractive features of these systems are there highly available structuring as well as reduced query latency towards user requests, which are achieved because of the inherent redundancy in the system through replication, where peers replicate each other's data so that when one peer is offline the other can serve the requested data. Most of the researches in P2P efficiency improvement, are concentrating on replica placement issue, i.e. how to place replicas at proper locations so that the overall performance of the system can be increased.

The goal of replica placement algorithms (RPA) also varies from one to another. The goal may be on maximizing the availability of peers, making the popular objects highly available or improving the QoS of the system. Such an intelligent RPA achieves much more performance than simply placing replicas randomly. There are static as well as dynamic RPAs proposed for P2P networks. Static RPA, replicate objects statically into certain peers which can be accessed by other peers. A RPA that considers the dynamic aspects of the system seems to be more efficient.

***Manuscript received Feb, 2013*.**

***Thara R J**, Department of Computer Science, College of Engineering Trivandrum , Trivandrum, India.*
Kala Karun A, Department of Computer Science, Sree Chitra Thirunal College of Engineering, Trivandrum , Trivandrum, India.
***Shine S**, Department of Computer Science, College of Engineering Trivandrum, Trivandrum, India.*

The replica placement problem in P2P is found to be NP Complete as that of any other distributed computing systems. Thus no polynomial time algorithm exists for the problem that yield optimal solution; i.e. optimal placement of replicas will always leads to exponential time algorithms. Rather heuristic based techniques called greedy technique are proposed by various papers which run to give approximate solution for the problem.

One problem with P2P is its overhead associated with their need on massive amount of control message to locate particular resource over the internet. To locate objects efficiently some P2P models propose centralized server which tracks the pointers of all the data in the network and some other models tracks pointers in a decentralized fashion. Replicating these pointers rather replicating the objects itself will also improve the system efficiency.

To make P2P highly efficient, a slight variation to its pure architecture is applied and yields P2P-CDN architecture. This hybrid architecture combines the complimentary benefits of P2P as well as Content delivery networks (CDN). Pure CDN networks, to improve the performance of content delivery, replicate popular objects at deployed surrogate servers at the edge of internet, which has the disadvantages of increased deployment cost. On the other hand pure P2P suffers from degraded QoS guarantees at the worst case. Now the combined P2P-CDN architecture solves problem with CDN as well as pureP2P network and results into a better architecture. Researchers are also looking for improved RPA for such a combined architecture for making it highly efficient. Such RPAs are also selected for this survey.

II. Literature Review

If you An Gyuwon Song et al.[1] proposed an efficient algorithm for improving the data availability in a peer-to peer storage system. Building a highly available P2P system, especially a P2P storage system is difficult as well as challenging as the peers can join and leave the system at any time without any notice. To meet this challenge a replica placement algorithm which exploits the availability pattern of each individual peer is being put forwarded here. The availability pattern of individual peers has been traced through a probabilistic model referred as Peer Availability Table (PAT), which simply covers short term as well as long term availability of peers under consideration. The replication algorithm estimates the similarity between PAT's and based on a heuristic approach, algorithm placed replicas at proper peers so as to make the system highly available.

Jian Zhou et al.[2] put forwarded an effective pointer replication algorithm that can reduce the worst case query latency in P2P networks. The degree of replication achieved here is dynamically adaptable to the instantaneous query arrival rate and churn characteristics of the system in order to reduce total control traffic. In an environment where the popularity of objects changes dynamically, the algorithm provides a rapid and light weight content replication strategy to reduce search and data access latencies. The procedure intelligently replicate roots in structured P2P networks, where pointers to objects are kept distributive, based on a greedy, 2-approximation algorithm.

Hai Jiang et al.[3] discussed an efficient replica placement algorithm for the Hybrid CDN-P2P(Content distribution networks – P2P) architecture, or HCDN, which combines the complementary advantages of CDN and P2P networks. Such a network can reduce the deployment cost faced by the pure CDN networks and can improve the quality of service in file sharing and video streaming applications compared to pure P2P networks. The CDN-P2P hybrid network is thus an efficient architecture for large-scale content distribution. A heuristic replica placement algorithm that takes into account the effects of P2P distribution at P2P level as well as the deployed surrogate servers at CDN level has been discussed. Compared to replica placement algorithms for pure CDN networks, the approach taken here can reduce the placement cost (i.e. the deployment cost), as the peer contribution is taken into account in this approach. In [5] they proposed an alternate and better strategy for placing replicas strategically in HCDN with assumed ring topology, where each surrogate server makes the replica placement in terms of the content replicas on k closer surrogate servers, which enhances the system scalability compared to the centralized replica placement heuristics. Their approach tries to obtain maximum gain out of replications.

Jian Zhou et al.[4] proposed a static, distributed replica placement algorithm in P2P networks. The algorithm considers the replica placement in real world P2P networks as a clustered K center problem which is NP-complete. It is an efficient approximation algorithm which can work with several orders of magnitude faster than optimal solutions with minimum access latency. The clustered k center problem is similar to classic k center problem in graph theory. The k center problem tries to find a set of k centers in an arbitrary graph such that the maximum shortest distance of all nodes to the nearest center is minimized. The proposed algorithm differs from the k center in that instead of choosing a set of k nodes to place the replicas, every node can act as center. The system consists of a set of n peers in a distributed P2P network and queries can originate in any node and are forwarded to the host node where the needed file is present. The objective of the algorithm is, given the replica count k , find the k nodes such that the maximum access latency is minimized, i.e. the target is to make the network a forest of k search trees with each tree rooted at one replication node.

Rzadca et al.[6] discussed replica placement strategies that tries to optimize availability and the number of replicas. The selfishness behavior of peers while decentralized replica placement, leads to performance inequalities that results into unreliable systems. In such a system, highly available peers tend to replicate data among themselves, and exclude peers with low availability from replication, so that highly available peers have their data availability increased at the expense of decreased data availability for less available peers. Thus a socially-optimal solution which can increase the availability of all the peers is needed. Centralized procedures can achieve this goal easily, but is complex as well as infeasible, rather some decentralized approach for optimizing availability, which can limit this selfishness among peers is needed. System wide cooperation rules are set to reach such an optimization. The procedure assumes two models for

proposing its solution: deterministic (time slot) model and probabilistic model. In probabilistic model, availability of a peer is the probability that the peer is available with respect to its expected life time. The goal is to maximize data availability given the constraints on the storage size. In contrast, in the time slot model peer availability is measured to be a function of time, thus availability is a set of time slots in which the peer is available with certainty. The goal is to minimize the number of replicas such that the sum of their availability periods covers the whole prediction time.

Yu-Chih Tung et al.[7] proposed a new approach for replica placement where it considers the available bandwidth of individual peers while placing replicas. In addition to bandwidth, it considers the online characteristics of individual users as well as popularity of the elements to be replicated during replica placement. The solution achieves increased data availability as well as improved data access probability for individual data items. The replication system assumed is group reciprocity based, where the users are arranged into different groups and each user in a particular group replicates data items of all other members of the group. In such a group the availability of a particular data item can be calculated to be the probability that at least one member in the group is online. The proposed algorithm divide the users into different groups by considering the popularity of the data they replicate as well as the available bandwidth they have so as to get a mutual balance between them. The major goals of the replication strategy are:

• To classify the users into different reciprocity based groups so that data availability of the worst group can be maximized.

• Improve the group such that when the group is available, i.e. any member of the group is online, the available members should have enough bandwidth to serve the entire request for the data it replicate.

III. Critical Evaluation

Throughout this study we have seen the replica placement problem in distributed computing environments especially in P2P environments. The problem is handled differently by different environments. Depending on our application as well as the platform used to develop it, appropriate replica placement algorithm can be taken. After review the following findings can be summarized.

There are three key phases in all the data replication algorithms which are replica selection, placement and management. Replica selection is the process of selecting replica among other copies that are spread across the networks. Replica placement is the process of selecting a network site to place the replica. Finally replica management phase has the major role of providing replication consistency and transparency. Replica consistency means how the system makes the replicated copies up-to-date, i.e. mechanisms for propagating updates efficiently across replicated copies of the objects should be there and the schemes should impose the required level of consistency across the system. Depending on the application we can either choose strict or eventual consistency models. The maintenance of replication transparency is another issue, i.e. to make the system behave as if there is no replication within the system; i.e. the complexity of replication should be hidden from users.

Replication strategies can majorly classify into centralized or distributed replication. In centralized replication, the replica placement decisions were taken in a central node that coordinates the entire networks. The drawback is evident – single point of failure of the coordinating node. On other hand in distributed replication all the nodes in the system participate in replica placement decision making. So it's a good idea to choose a distributed replica placement strategy over centralized schemes for a distributed environment like P2P or grid environments.

Replication algorithms can be static or dynamic. In static approaches the created replica will exist in the same place till user deletes it manually or its duration is expired. Such a scheme is not good to choose as when the client access pattern changes, the benefits brought by replication will decrease. On the contrary, dynamic replication takes into account the changes in environment and automatically creates new replicas for the referenced data files or moves replicas to other sites as needed to improve performance.

The replication algorithms may be popularity based, were the objects with highest popularity is given maximum number of replicas. The popularity of certain objects is evaluated based on access pattern of the similar objects obtained from historical data. Some replication algorithms are threshold based, where a data set with an access rate higher than a fixed threshold is deemed as popular and gets replicated from a heavily loaded site to another site. It is a big challenge to set the popular data set threshold right. Too high a threshold will lead to insufficient data replications and result in a still unbalanced system. Alternatively, too low a threshold will trigger many unnecessary replications, which not only waste network bandwidth but may even lead to over-reactions that cause performance degradations. Moreover, the threshold needs to be adjusted for different workloads. A fixed threshold is inappropriate for dynamic environments.

The replica placement algorithms may assume different topologies for the placement environment. A number of early algorithms address placement of data replicas in parallel and distributed systems with regular network topologies such as hypercube, torus, and rings. These networks posses many attractive mathematical properties that enable the design of simple and robust placement algorithms. Most of the replica placement algorithms in grid assume tree topology, in which requests can only be forwarded upwards towards the root node site and is satisfied by the nearest replica. In a real-world data grid, the requests served on a site can be generated anywhere.

The algorithm design techniques used may also vary for different algorithms. Some algorithms propose a heuristic solution for the replica placement problem. However, the algorithm does not guarantee to find the optimal solution. They use approximation algorithms to find the near optimal placement. There are replica placement algorithms which employ dynamic programming, that yield optimal solution. In general the replica placement problem is found to be NP

complete and an approximated solution that runs in polynomial time is proposed by most of the papers.

Another important issue is choosing the optimal number of replicas. The denser the distribution of replicas is, the shorter the distance a client site needs to travel to access a data copy. However, maintaining multiple copies of data in network are expensive in terms of update cost, and therefore, the number of replicas should be bounded. Clearly, optimizing access cost of data requests and reducing the cost of replication are two conflicting goals. Finding a good balance between them is a challenging task.

The quality of service (QoS) treatment is an important goal behind the success of any replica placement schemes. The QoS requirement may be from user perspective (eg :- Hop count) or from system perspective (e.g. :- Work load constraints, Link bandwidth constraints etc). A good replication algorithm should try to impose QoS requirements.

A replication scheme that jointly considers online characteristic of users who access data, popularity of the accessed data and bandwidth capability of individual peers is needed, which can improve not only data availability, but also access probability for each data item. The proposed solutions should take the non uniform skewed distribution of data popularities and tries to attain a balance between while replicating objects. The RPA should optimize data availability such that the highly available as well as less available peers should be able to improve their availability. The basic unit of replication varies from one system to another, like the replicas of popular web objects in CDN-P2P networks, pointers or links in unstructured P2P etc.

The proposed solution should considers the major challenge of any P2P system, i.e. individual peers can join and leave the system at any time without any prior notification. This dynamic membership change handling is crucial and the RPA should intelligently adapt to such changes.

RPA for P2P systems as that of any other distributed environment is found to be NP complete, hence the researchers in this field always tries to formulate their solutions using a heuristic methodology. The algorithm design technique they adopt is greedy, which always yield an approximated version of the original solution. A scheme based on dynamic programming technique will definitely out perform the current replica placement scenarios; this can be proposed as a future work in this field.

The current RPAs can be extended by adding mechanisms for propagating updates efficiently across replicated copies of the updated objects; i.e. required consistency rules has to be proposed; depending on the application we can either choose strict or eventual consistency models. The maintenance of replication transparency is another issue, i.e. to make the system behave as if there is no replication within the system; i.e. the complexity of replication should be hidden from users.

IV. CONCLUSIONS

The paper gives an overview of replication and replica placement problems in the emerging P2P distributed computing environment. Replica Placement Problem aims to place replicas strategically yielding maximum benefits from replication. The focus of this paper is to give an overview of various centralized/distributed and static/dynamic replica placement algorithms in structured as well as unstructured P2Ps. The paper compared 7 popular P2P replica placement algorithms and extracted their main features. Each of the replica placement strategies considered in this paper aims at different goals and optimize various aspects of the system, making P2P highly efficient. The paper finally extracted and summarized major features behind each of the RPAs proposed by various authors.

REFERENCES

[1] Gyuwon Song; Suhyun Kim; Daeil Seo, "Replica Placement Algorithm for Highly Available Peer-to-Peer Storage Systems", Advances in P2P Systems, 2009. AP2PS '09. First International Conference, 2009 , Page(s): 160 – 167

[2] Jian Zhou; Bhuyan, L.N.; Banerjee, A. "An effective pointer replication algorithm in P2P networks". Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium. 2008 - Page(s): 1 – 11.

[3] Hai Jiang; Zhan Wang; Wong, A.K.; Jun Li; Zhongcheng Li. "A Replica Placement Algorithm for Hybrid CDN-P2P Architecture". Parallel and Distributed Systems (ICPADS), 2009 15th International Conference, Page(s): 758 – 763

[4] Jian Zhou, Xin Zhang, Laxmi N. Bhuyan, Bin Liu. "Clustered K-Center: Effective Replica Placement in Peer-to-Peer Systems." In proceeding of the Global Communications Conference, 2007. GLOBECOM '07, Washington, DC, USA, 26-30 November 2007

[5] Zhan Wang; Hai Jiang; Yi Sun; Jun Li; Jing Liu; Dutkiewicz, Eryk. "A k-coordinated decentralized replica placement algorithm for the ring-based CDN-P2P architecture." Computers and Communications (ISCC), 2010 IEEE Symposium – Page(s): 811 - 816. 2010

[6] Rzadca, Krzysztof; Datta, Anwitaman; Buchegger, Sonja. "Replica Placement in P2P Storage: Complexity and Game Theoretic Analyses". Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference, 2010 , Page(s): 599 – 609.

[7] Yu-Chih Tung; Lin, K.C.; Cheng-Fu Chou. "Bandwidth-Aware Replica Placement for Peer-to-Peer Storage Systems". Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, Page(s): 1 - 5

[8] Mahesh Maurya, Ketan Shah, "A Review on File replication Algorithms", Techno-Path Journal of Sci. Engg. & Tech. Mgt. Vol 3(2), July 2011.