ISSN:2231-0711
www.ijcset.net

An Efficient Architecture for Consistency Maintenance and Peer Synchronization in Distributed Systems

Bollimuntha Kishore Babu[1] and Attaluri Rajesh[2]
M.Tech (CNS), Green Fields, K.L.Univerisity, Vijayawada, India.

bkishore002@hotmail.com[1]
cheguvera@techie.com[2]

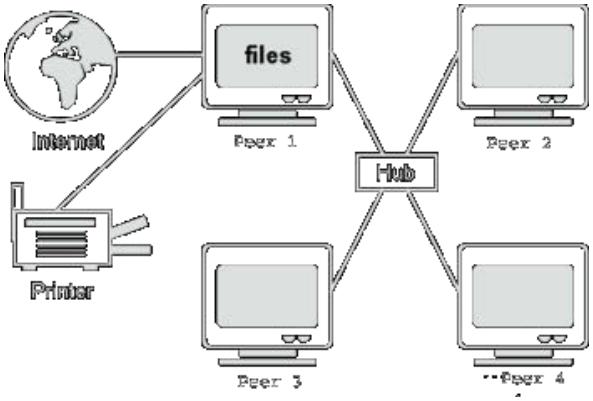
Abstract - **P2P is a popular technology used for file sharing. File replication and Consistency maintenance are the techniques used in P2P for high system performance. File replication methods specify replica nodes without considering consistency maintenance which may lead to high overhead for unnecessary file replications and consistency maintenance. Consistency maintenance methods update files without considering file replication dynamism which may not give the accuracy of replica consistency. Instead of passively accepting replicas and update messages, we develop a mechanism which combines both file replication and consistency maintenance. This mechanism is very efficient in file replication and consistency maintenance at low cost. In this mechanism, accepting replicas and updates is based on file query rate and update rate. Peer-to-Peer architectures for content and knowledge management foster the creation of communities of workers in which effective knowledge and information sharing takes place. In such communities, workers have similar capabilities in providing other workers with data and/or services and are autonomous in managing their own knowledge objects. Since objects are typically shared among a set of workers, problems regarding concurrent access to content, content consistency and synchronization of peers arise. This paper describes a hybrid architecture for the management of data consistency and peer synchronization. The designed framework combines centralized, yet dynamic, mechanisms for metadata management and peer-to-peer mechanisms for data transfer.**

Keywords— P2P, File replication, consistency maintenance.

1. 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, 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 presents a mechanism which integrates File Replication and Consistency Maintenance to achieve high efficiency in file replication and consistency maintenance at a lower cost.

Most of the current architectures for content sharing and knowledge management are typically based on client/server architectures in which one or more servers act as central entities. In these architectures, knowledge handled by workers must be managed according to organizational guidelines. However, these centralized approaches do not reflect the social nature of knowledge . As discussed in the seed of new knowledge is individual (tacit) knowledge, but the importance of the knowledge increases when it becomes available to the whole organization. Therefore, the externalization of tacit knowledge is a quintessential process to create new knowledge; this typically requires people to interact and collectively reflect on a problem or an idea. Such observations promote the demand for new technological architectures that place more emphasis on collaboration.

Moreover, since an increasingly number of workers very often operate outside of the traditional office, a virtual workplace where the physical workplace can be recreated is required. The Virtual Office (VO) solution complies with this requirement. A VO can be viewed as a work environment defined regardless of the geographic locality of the employees. This model is becoming essential since, even in conventional offices, today many business relationships are necessarily maintained across distributed environments. The VO is based on the concept of workspace. A workspace is a community of people that work together, as if they were in the same physical workplace, concurrently access shared content and accomplish common objectives. Such communities produce and exchange knowledge within workspaces through a set of shared tools.

The Virtual Office model cannot be effectively managed through a centralized entity in charge of object updating and dissemination, because this solution can hinder the autonomous interactions among workers and be poorly scalable. Conversely, the Peer-to-Peer (P2P) paradigm can be more appropriate and effective, because it fits both the requirements of collaboration (synchronous and asynchronous) and knowledge sharing that are raised by the adoption of VOs. In fact, P2P architectures naturally support the creation of communities (e.g., workspaces, peer groups) in which content and conveyed knowledge can be created, shared, exchanged and transformed.

We developed a P2P system, named K-link+, that implements the VO model and allows users to create flexible and collaborative P2P applications for knowledge management. In K-link+, peers can concurrently work on the same shared documents/files, in the following referred to as "knowledge objects" or simply "objects". To foster peer autonomy, different local replicas of an object can be created, so concurrent access can affect data consistency if adequate mechanisms are not provided. Moreover, peers can join or leave the system at any time, thus introducing the synchronization issue: synchronization is required by peers that reconnect to the network and need to be informed about recent updates made on objects by other peers. Object consistency is also a fundamental reliability requirement for a P2P system. Even if it is not possible, or convenient, to guarantee that all users are provided identical object replicas all the time, mechanisms must be provided to make users work without any actual limitations .

This paper describes an architecture designed for the management of knowledge in small/medium enterprises and actually adopted in the K-link+ system. This architecture adopts a hybrid model to cope with the content consistency and peer synchronization issues. It exploits the efficiency of centralized models but at the same time includes decentralized features, which assure scalability properties when the system size increases. This is accomplished by using: (i) a unique and stable server to maintain a limited amount of metadata information about shared objects, (ii) a number of interchangeable servers that maintain and manage the primary copies of shared objects, and (iii) a pure decentralized mechanism that allows P2P nodes to effectively exchange up-to-date object replicas.

To this aim different roles can be assumed by K-link+ Nodes (KLNs). In particular, a Rendezvous node maintains a common view about shared objects and their state. A set of Manager nodes are in charge of receiving object update requests from Worker nodes and possibly authorizing them. Finally, Broker nodes are used to speed up the propagation of updated objects over the network. The work presented here extends and refines the preliminary architecture description given and provides an analytical study through which the described hybrid architecture was assessed and important performance indices, such as computation load and response time, were evaluated.

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.

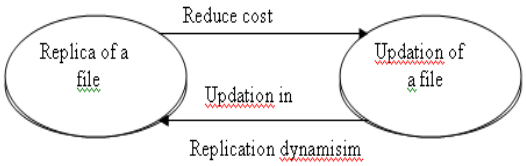


Fig. 1. Interrelationship between replica of a file and updation of a file

Replication dynamism deals with replica node generation, deletion and failures.

2. RELATED WORK

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 number of replicas are based on queries.

Replication of content is an important issue in P2P systems, especially if these are devoted to collaborative knowledge management. Replication mechanisms are usually classified into reactive and proactive mechanisms. In reactive replication, as objects are transferred from the home node to the requesting peer, intermediate nodes through which the data flows, determine independently whether or not to cache the content. Some researchers propose to cache pointers instead of real objects in order to yield better query search performance. In DiCAS , queries are forwarded to peers of a predefined group which passively cache the pointers in an unstructured P2P network. However, a large overhead is necessary to update the pointers when the object is moved or deleted, since the updated location information has to be flooded to the whole overlay network.

In proactive replication, content is pushed to selected peers by the node that stores the primary copy, in order to obtain better performance in terms of query latency, load balance etc. However, the cost of replicating objects to a large number of peers can be cumbersome in both disk space and bandwidth, particularly for systems that support applications with large objects (e.g., audio, video, software distribution). A replication strategy based on object popularity in unstructured P2P networks. Nevertheless, this strategy does not reduce the worst-case search latency for all the objects.

The strategy adopted in this paper borrows characteristics of both reactive and proactive approaches. A push-based mechanism is initiated by a peer when it generates or receives an updated version of an object, since it forwards this object to other workers, in a P2P fashion. This approach assures a quick dissemination of objects to the members of a community but, owing to its decentralized and unstructured nature, cannot guarantee that every worker is given the updated version of every shared object all the time. However, the updated version of an object is always maintained by the related Manager node. Therefore, whenever a worker cannot obtain the updated version of an object through the P2P mechanism, it can always request this object, with a pull modality, to the Manager.

An issue strictly related to replication is content consistency, which is, in fact, a fundamental reliability requirement for a P2P system. Current approaches differ according to the scale of P2P systems. In a large-scale and dynamic system, it is complex and cumbersome to guarantee full consistency among replicas, so researchers have designed algorithms to support consistency in a best-effort way. A hybrid push/pull algorithm is used to propagate updates, where flooding is substituted by rumor spreading to reduce communication overhead.. SCOPE is a P2P system that supports consistency among a large number of replicas, at the cost of maintaining a sophisticated data structure. By building a replica-partition-tree (RPT) for each key, SCOPE keeps track of the locations of replicas and then propagates update notifications.

Conversely, in a small- or medium-scale system, it is possible to adopt centralized schemes to guarantee a strong consistency model, which is often the sequential model. an algorithm for file consistency maintenance through virtual servers in unstructured and centralized P2P systems is proposed. Consistency of each dynamic file is maintained by a virtual server (VS). A file update can only be accepted through the VS to ensure the one-copy serializability.

The hybrid architecture described in this paper, and adopted in the K-link+ system, is specifically designed for knowledge management in small/medium enterprises. Its main purpose is to combine the efficiency of centralized models and the scalability and fault-tolerance characteristics of decentralized systems.

3. PROPOSED APPROACH

Our system 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.

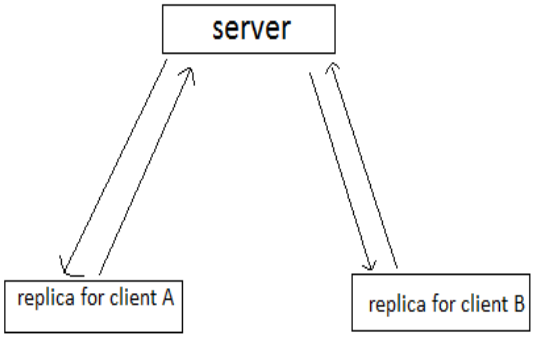


Fig. 2 Combined approach of file replication and consistency maintenance

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.

3.1 Adaptive 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.

3.1.1 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.

3.1.2 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

3.1.3 Adaptation of Replica node

A file might not be accessed frequently .so the replica node should frequently update their query passing rate, so that underutilized replicas can be removed.

3.2 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 node might be generated 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 reduce the polling operations to save cost and to maintain accuracy in consistency maintenance[1].

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. TTRquery and TTRpoll denotes the next time, where the file is updated.

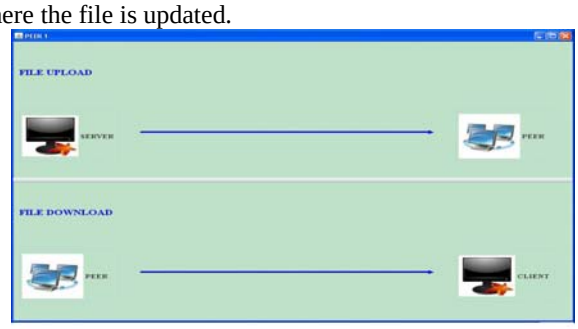


Fig 3. file upload and download using peer

4. ARCHITECTURE AND IMPLEMENTATION OF K-LINK+

K-link+ is a collaborative P2P system that provides users with a Virtual Office environment, in which content can be shared, to enable collaborative work, and replicated to foster peer autonomy. Different applications (document sharing, messaging, shared boards, agenda, etc.) can be integrated within a single environment (a K-link+ workspace) and new tools can be added as new components to support emerging requirements. In this section, the system architecture is briefly presented along with its implementation in JXTA .

4.1 THE K-LINK+ ARCHITECTURE

The K-link+ architecture is based on five layers including basic grouping and communication services, data handling services, semantics services, workspace management services and a set of high level tools for content sharing and user cooperation. Figure 4 shows the K-link+ architecture, whose layers are described in the following.

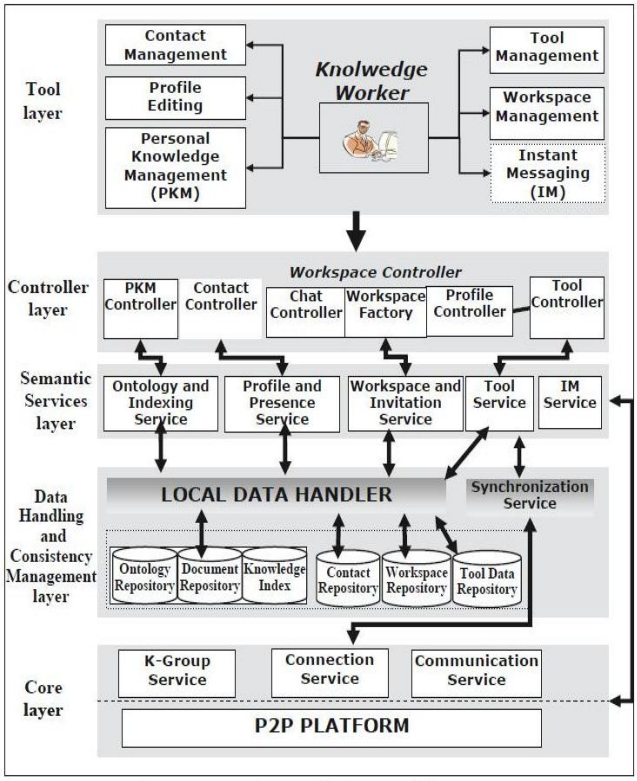


Fig 4. K-Link+ Architecture

K-link+ Core Layer:

This layer defines the K-link+ core services that are exploited by higher layers. In the current version of K-link+, we adopted the JXTA framework to implement these services, though any other P2P infrastructure can be used. Among the core services, the K-Group Service allows KLNs to create new K-Groups (e.g., communities or workspaces), while the Connection Service allows KLNs to join the K-link+ network at any time. Features used to send and receive messages are provided by the Communication Service.

Data Handling and Consistency Management Layer:

This layer is responsible for handling the problems introduced in the introductory section, that is, concurrent access to shared objects, object consistency and peer synchronization. The layer includes the Local Data Handler, which manages a set of local repositories to store information about contacts, workspaces, objects and so on.

K-link+ Semantic Services Layer:

The services of this layer manage local and remote operations performed by a K-link+ user. The Ontology and Indexing Service deals with operations involving ontologies (creation, update) that K-link+ exploits to describe and annotate resources in a semantic way. The Indexing Service copes with the indexing of documents for keyword based searches. The Profile and Presence Service manages state check operations and enables users to create and publish their profile within the K-link+ network. The Workspace and Invitation Service handles workspace set up and population by sending invitation messages to other KLNs. The Tool Service is used to add new tool instances to workspaces at run time. The Instant Message (IM) Service allows KLNs to communicate each other via a chat-like system.

K-link+ Controller Layer:

This layer contains a set of controllers through which the system interacts with the Data Handling and Consistency Management Layer. The Workspace Controller manages workspace settings through the creation of workspace profiles that contain information about workspace topics, sets of tools and workspace members. The Contact Controller enables workers to discover other peer workers in the network and add them to a personal Contact List. The PKM Controller is delegated to manage personal knowledge, that is, knowledge that is autonomously handled by workers. The Tool Controller allows workers to handle operations (add, update, remove) on knowledge management tools.

K-link+ Tool Layer:

This layer provides a basic set of K-link+ Tools (file sharing, shared calendar, contact manager, etc.) that can be embedded in a workspace. In addition, other tools can be developed and included in the system as modular components, with the only requirement that the K-link+ tool interface must be implemented. Tools are shared among workspace participants: when a new tool is added to a workspace, a local tool instance will be automatically created on the K-link+ client of each workspace member. Hence, objects created through workspace tools can be shared among workspace members.

4.2 THE K-LINK+ IMPLEMENTATION

While the K-link+ technology is independent of any particular P2P architecture, however the JXTA framework was used to implement it. As a consequence of this choice, the Core Layer of the K-link+ architecture maps each K-link+ peer, also called K-link+ Node (KLN), to a JXTA peer and each K-link+ group (K-group) to a JXTA peer group. JXTA peers can be divided into two categories: edge peers and rendezvous peers. Edge peers are transient peers provided with resource discovery and publishing capabilities. On the other hand, rendezvous peers are expected to be more stable. The aim of a rendezvous peer is to enable edge peers to discover and publish network resources. In JXTA each network resource (e.g., peers, peer groups, services) is published through XML documents called Advertisements. To participate in a P2P network, an edge peer needs to know how to connect to at least a rendezvous peer. Typically, an edge peer maintains a list of known rendezvous peers (called "seed" rendezvous) and participates in dynamic discovery of new rendezvous nodes. This allows edge peers to fail-over to an alternative rendezvous when needed, so as to enhance overall network reliability. Every JXTA peer can act both as an edge peer or a rendezvous peer. In fact, an edge peer can adaptively become a rendezvous peer if it cannot connect to any rendezvous for an extended period of time.

5. CONTENT CONSISTENCY AND PEER SYNCHRONIZATION

In K-link+, several users can work concurrently on shared objects.

To favor the autonomy of users, the system allows different replicas of the same object to be created, so that users can work on their local copies. As mentioned in the purpose of the Data Handling and Consistency Management layer is to ensure data persistence, consistency management and peer synchronization. In the context of K-link+, the sequential consistency model is adopted, which assures that all updates performed on an object are seen in the same sequence by all the peers.

The model is implemented by associating, to each object, a K-Link+ node (called Manager), which is responsible for authorizing object updates thus allowing the KLNs to view the updates in the same order. In particular, each object is assigned a Version Number (VN), which is incremented after each update. In more details, K-link+ defines the following set of roles that can be assumed by workspace nodes:

Creator. It is a KLN that creates a shared object and specifies the Manager List (ML), i.e. the list of KLNs that can assume the Manager role for this object. Managers are ordered on the basis of their responsibilities in managing the object.

Rendezvous. For each workspace, one rendezvous node maintains metadata about all the shared objects in a Consistency Table and provides such information to workspace members. The Rendezvous stores up-to-date information about objects, in particular the identity of the node which is currently in charge of each object (i.e., the Current Manager) and the current VN.

Manager. An object Manager is a KLN that manages the life cycle and is contacted by KLNs when they want to propose an object update. An object can be assigned to several Managers, but at a given time only the Current Manager, i.e., the first online Manager in the ML, is actually responsible for the object. The Current Manager can decide whether or not to authorize an object update, according to the specific set of semantic rules associated to the object. KLNs are informed about the identity of the Current Manager by the Rendezvous.

Broker. It is a KLN that maintains an updated copy of an object and can forward it to other KLNs. Whereas the Manager role is assigned at object creation time, the Broker role is dynamic, since it can be played by any node whenever it maintains an updated copy of an object.

Worker. It is an ordinary KLN that operates on an object and possibly issues update proposals to the Current Manager. Workers can obtain an updated copy of an object either by a Broker, in a P2P fashion, or by the Current Manager of the object, with a centralized approach.

6. 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. 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.

This paper focuses on two relevant issues in P2P systems, i.e., the consistency of data that arises, since users can work concurrently on multiple replicas of the same object, and the synchronization of peers that is needed when they disconnect from the platform and reconnect again. A model was designed to face these issues in K-link+: a small/medium scale collaborative system founded on the concepts of Virtual Office and workspace. A hybrid model was adopted that combines the efficiency of centralized interaction patterns, which are used for the management of metadata information about knowledge objects, with the scalability and adaptive features of decentralized interactions, which are adopted for the update and dissemination of actual data. An analytical performance evaluation, based on the theory of queue networks confirms the suitability of this approach.

ACKNOWLEDGEMENTS

We are greatly delighted to place my most profound appreciation to Dr. K. Satyanarayana Chancellor of K.L.University, Dr. K. RajaSekhara Rao Principal , Prof S.Venkateswarlu HOD CSE, Dr. K. Subramanyam coordinator for M.Tech under their guidance and encouragement and kindness in giving us the opportunity to carry out the paper. Their pleasure nature, directions, concerns towards us and their readiness to share ideas enthused us and rejuvenated our efforts towards our goal. We also thank the anonymous references of this paper for their valuable comments.

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.