ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 9, March 2013

Survey- Analyzing Performance of File Distribution and Search in Unstructured P2P Networks

N. Senthil Madasamy

Abstract*—All the peers in a peer to peer network form an overlay network and share resources such as storage, bandwidth and processing capacity have gained many interests recently. The definition of peer-to-peer networks allows centralized servers and super nodes. Peer-to-peer networks can be also classified into the five categories depending on their functionality, which include distributed computing, communication and collaboration, Internet service support, data base system, and file distribution. However, most of the peer-to-peer networks fall into the file distribution category, for example, Napster, Gnutella, Kazza and Bit Torrent. Peer-to-peer file distribution networks allow peers to publish, search and download files from each other. File need to be located first before download in an unstructured Network. A search is usually performed to find the desired file. In this survey, I study the performance analysis of peer-to-peer file distribution networks and studying the search performance in unstructured networks. I compare multicast approaches based on cooperative m-ary trees and cooperative mesh network achieve different content delivery delay when the participating peers are allocated far together in a wide area networks.

***Index Terms*— Peer-To-Peer, Query Efficiency, Search Responsiveness, TTL, Search Responsiveness**

I. INTRODUCTION

Peer-to-peer networks have been commonly used for tasks such as file sharing or file distribution. A class of cooperative files distribution systems where a file is broken up into many chunks that can be downloaded independently. The different peers cooperate by mutually exchanging the different chunks of the file, each peer being client and server at the same time. While such systems are already in widespread use, little is known about their performance and scaling behavior. I develop analytic models that provide insights into how long it takes to deliver a file to *N* clients. The Unstructured P2P centralized overlay model was first popularized by Napster. This model requires some managed infrastructure (the directory server) and show some scalability limits. A flooding requests model for decentralized P2P overlay systems such as Gnutella, whereby each peer keeps a user-driven neighbor table to locate data objects are quite effective to locate popular data objects, thanks to the power law property of user-driven characteristics. However, it can lead to excessive network bandwidth consumption, and remote or unpopular data objects may not be found due to the limit of lookup horizon, typically imposed by TTL. The argument is that DHT-based systems while more efficient at many tasks and have strong theoretical fundamentals to guarantee a key to be found if it exists, they are not well suited for mass-market file sharing. They do not capture the semantic object relationships between its name and its content or metadata. In particular, DHT-based ability to find exceedingly rare objects is not required in a mass-market file.

II. PERFORMANCE ANALYSIS OF SEARCH

Reference [1] and [2] both study the search performance in unstructured peer-to-peer networks. In [2], the search performance metrics in searching are given in both user aspects and load aspects. In user aspects, the metrics include Pr success, which is the probability of finding the queried object, and N hops, which is the delay in finding an object as measured in number of hops between peers. In load aspects, the metrics include average number of messages per node, number of nodes visited, and the maximum number of messages in the busiest node. In [1] , the search efficiency can be measured quantitatively by taking the product of query efficiency (QE) and search responsiveness (SR). In turn, QE is defined as the ratio of query hits to messages per node. SR evaluates responsiveness and reliability, which can be defined as: Search Responsiveness = Success Rate / Hops Number.

As I compare the defined search performance metrics, [1] provides a quantitative measure, while [2] provides the analysis from both the user and load aspects.

A. P2P network model

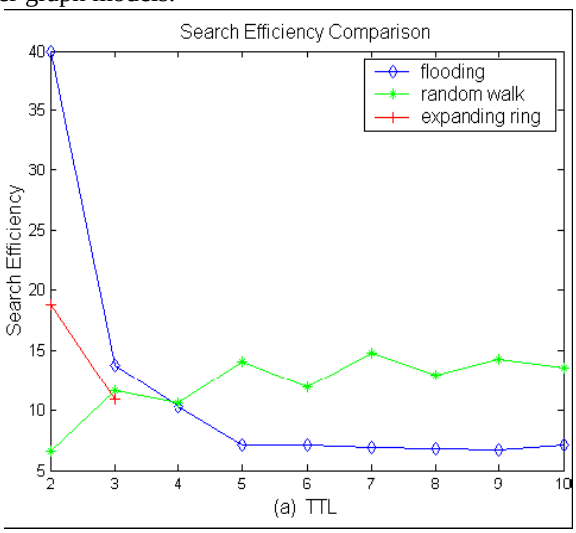
To model the unstructured peer-to-peer network, reference [2] uses four graph models: Power-law graph, Normal random graph, Gnutella graph, and two-dimensional graph. In Power law graph, the node degrees follows a power law distribution: if one ranks all nodes from the most connected to the least connected, then the i^{th} most connected node has w/ia neighbors, where w is a constant. Normal random graph is generated by a GT-ITM topology generator [3]. Gnutella graph is obtained from the Gnutella network in 2000. Finally, the two dimensional graph just is a two dimension grid. In [1], the graph used has the property of a two-stage power-law distribution, since the Gnutella network has this property according to their reference paper [4].

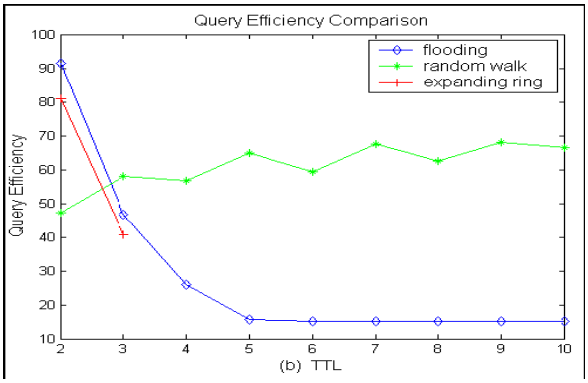
B. P2P Search algorithms

The search algorithms analyzed in both papers include flooding, expanding ring and random walks. Flooding is simply to send the query to all its neighbors, and the neighbors will do the same again by forwarding the query to all their neighbors. To limit the number of messages, TTL

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 9, March 2013

(Time-To-Live) is used to control the hops of a query. The main problem with flooding is the selection of TTL. A too small TTL may give few or no responses while a too large TTL results large amount of query messages. Expanding ring uses successive floods with increasing TTLs. A node starts to send with a small TTL, waits to see if the search is successful, if not, it increases the TTL linearly. In this way, the expanding ring algorithm solves the TTL selection problem. Random walk algorithm forwards query messages to a randomly chosen neighbor at each step until the object is found. Random walk cuts down the message overhead significantly compared to other two algorithms. However, the standard random walk algorithm may result long delay because the number of hops to the searched object may be long. To decrease the delay, in [2], it also evaluates random walk with a number of "walkers". By using *k* walkers simultaneously, it is expected to cut the delay at the cost of more query messages. Both of the papers[1] and [2] simulated the setup with different algorithms and graph models. The details of their simulation are however not given as in many of the research papers. In Fig. 1, the results from paper [1] including search efficiency, query efficiency, and search responsiveness of the three algorithms are shown. As can be seen in the figure, random walk has good query efficiency while flooding has good search responsiveness. The performance of expanding ring is also quite poor. As the expanding algorithm finds the queried object in 3 hops, so no performance metrics are show for TTL greater than 3. The conclusion made from the paper are that current search algorithm either overwhelms network bandwidth hoping to meet user's satisfactory requirement, or sacrifice responsive performance in order to produce scalable solutions. However, they have not studied the performance of random walk with *k* walkers. In addition, they only used one graph model. Even though their reference paper suggests that Gnutella network can be model using the chosen graph model, but it would be interesting to see how the algorithms work in other graph models.





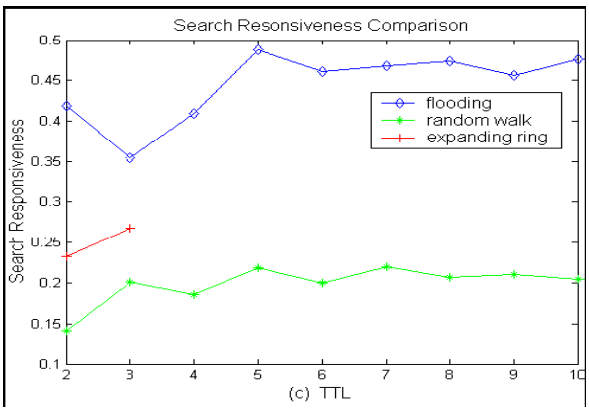


Fig. 1, Search Efficiency, Query Efficiency, and Search Responsiveness comparison of the three algorithms simulated in Gnutella network

Although there are plenty of results from paper [2] showing performance for different setups, their main conclusion is clear: A random walk with 32 walks resolves query as fast as the flooding algorithm, while reducing the network traffic by two orders of magnitude.

III. PERFORMANCE ANALYSIS OF FILE DISTRIBUTION

A. Service capacity of peer-to-peer systems

In a peer-to-peer system, if a single peer wants to distribute a popular file to many other peers, the service capacity is rather poor at the beginning due to the shortage of the "server". However, as soon as one of the peers finishes downloading, the peer becomes a server and starts serving other peers. It is obvious that this process causes the number of server in a peer-to-peer system grows exponentially at the beginning and the total service capacity of the system increases. This state in which the number of servers grows is called transient state. At some point, the system enters a steady state where the number of servers and service capacity becomes stable. The service capacity in these two states depends on four factors:

a. Data management: a file may be partitioned into various parts permitting concurrent downloading from multiple peers; the granularity and placement of these are critical.

b. Peer selection: the mechanism whereby a peer is selected as a server may take into account load balancing, bandwidth

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 9, March 2013

availability, and differentiate among peers who contribute more to the community.

c. Admission, scheduling and bandwidth allocation policy: limiting the number of concurrent downloads and priority scheduling of service or differentiation in the bandwidth allocated among peers.

d. Traffic dynamics: the request arrival processes and the dynamics of how peers leave the system and delete documents.

Reference [5] explores the interactions among some of these factors and their impacts on the service capacity. Transient state and stationary state are analyzed by using two models, deterministic model and Markov Chain model.

B. Deterministic model

Deterministic model is a simple model and is used to exhibit the basic transient dynamics for the service capacity and the performance of a peer-to-peer system. Assuming that there are *n* users expecting acquire a file which is initially available at a single peer; defining $n = 2k$. Each peer's uploading capacity is limited to *bbps*, and there is no constrain to the network capacity. The size of the file is s bits. A peer can serve the file once it finishes downloading –see figure 2.

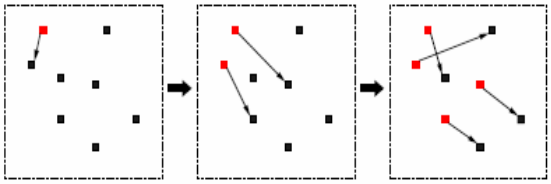


Fig. 2, File Sharing in Peer-To-Peer System

The strategy is the server first serves one user at rate *b*, which makes the service capacity grows to *2b*, then these two servers serve additional users. The process continues until *n* users are served. The service time for each peer to complete downloading is $\tau = s / b$ seconds. Since the number of peers that can serve the file doubles, it leads to an exponential growth of $2t/\tau$ in the "service capacity". With this proposed strategy, n peers will be served by time $\tau log2n = \tau k$. In this transient state, the average delay $\bar{d}$ is calculated as follow:

$$\bar{d} = \frac{1}{n}\sum_{j=1}^{n} d_j = \sum_{i=0}^{k-1} 2^{i-k}\tau(i+1) = k\tau - \frac{n-1}{n}\tau$$

$$= \tau\left(\log_2 n - \frac{n-1}{n}\right) \approx \tau \log_2 n \qquad (1)$$

d_j denotes the delay experienced by the *j*th peer to complete downloading. According to previous assumption, $n = 2k$; using $i+1$ to represent *j*, the j^{th} peer completes service at time $(i+1)\,\tau$. From equation (1), the conclusion is that although the system sees an initial burst of *n* requests, the average delay seen by peers scale as *log2n* which is relative to the linear scaling of *n*. From this point, the typical way to speed up peer-to-peer system is to enable multi-part download. The file is divided into *m* chunks with identical size, and the peers can serve such chunks immediately once they are downloaded. Intuitively by dividing the download process into smaller chunks, transfers can be pipelined among the peers. As a result, the performance should be improved by deducing the delay by a factor of *1/m*.

C. Markov chain model

Markov chain model is used for the steady state analysis of the service capacity. First, all the peers in a peer-to-peer system are downloading or serving a file. One assumption is that there is always at least one peer acting as a server to serve the system. Multi-part download is used. So, the peers in the process of downloading can serve the parts that have been finished to other peers. Both the serving only peers and the downloading peers can contribute to the system's service capacity. The difference is a downloading peer's contribution is a fraction η of that of a peer who has already finished downloading the whole file. μ is used to defined the service rate for a peer which has already finished downloading. So, the total service capacity is denoted by $\mu(\eta x + y)$. However, the peer finishes downloading may leave the system at rate γ. The evolution for the state of this system can be described by a continuous time Markov chain model by using a rate transition matrix Q:

$q((x, y), (x+1, y)) = \lambda$ *new request*
$q((x, y), (x-1, y+1)) = \mu(\eta x + y)$ *serve a peer*
$q((x, y), (x, y-1)) = \gamma y$ *exit system*

Reference [5] numerically computes the stationary distribution for this Markov chain model. Then, a real system experiment was deployed. The performance characteristics are compared with the numerical results provided by the Markov chain model. The conclusion is a peer-to-peer system in steady state will likely exhibit fairly good performance. When the rate at which peer leaves the system is slow, the documents with a high offered load may improves the average performance versus those with lower loads.

D. Approaches of file distribution

Reference [6] analyzes different approaches to distribute a file from one single server to a set of peers. In particular, three distribution models are analyzed:

1. A linear chain architecture, where peers are organized in a chain with server uploading chunks to
peer P1, which in turn uploads the chunks to P2 and so on.
2. A tree architecture, where peers are organized in a tree with certain out degree, all nodes that are not leaves in the tree uploads chunks to other peers.
3. A forest of trees consisting of different trees, which partition the files into several parts and constructs spanning-trees to distribute it to all peers. The file to distribute is partitioned into C chunks. If b is the download rate, then the time to download a complete file with C chunks is referred to one round or 1 unit of time. Thus, the time needed to download a single chunk at rate b is 1/C. First the authors studied analytically the performance of the linear chain architecture. At time 0, the server starts to serve the first peer. At time 1/C, the first peer receives the first chunk and starts to serve a second peer. Likewise, once the second peer receives the first chunk at 2/C, it starts to serve the third peer. As a result, the length of the peers increases by one each 1/C time unit. At time 1, the server finishes uploading the file to the first peer. If there are still peers left that have not received a single chunk, the server starts a new chain. So at time 2, the server starts a new chain, in turn this makes t+1 chain with in t

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 9, March 2013

rounds. The number of served peers at time t can be analytically derived and is given by

$$N_{linear}(C,N) = C * t^2 \quad (2)$$

So the number of served peers grows linear with the number of chunks and quadratically with number of rounds. From equation (2) it also could be derived that the time needed to completely serve N peers is:

$$T_{linear}(C,N) = 0.5 + \sqrt{0.25 + \frac{2N}{C}} \quad (3)$$

With a tree architecture with degree k, the server serves k peers in parallel at rate b/k. Each interior node uploads file to other k peers, each at rate b/k. The then number of peers served in t rounds is:

$$N_{tree}(C,k,N) = k^{(t-k)\frac{C}{k}+1} \quad (4)$$

And the time needed to completely serve N peers is:

$$T_{tree}(C,k,N) = k + \log_k\left(\frac{N}{k}\right) * \frac{k}{C} \quad (5)$$

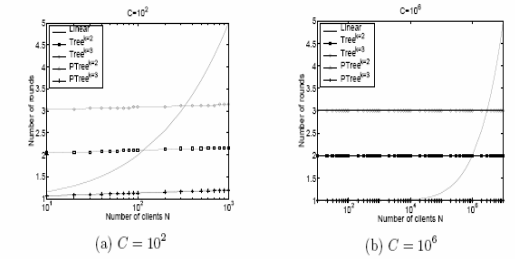
As the leaf nodes in a standard tree does not upload at all thus wasting its uploading capacity, so parallel trees can be used to improve the performance. In a parallel tree, a peer receives k parts in parallel from k different peers, each at rate b/k. While the peer helps distributing at most one part of the file to k peers. Therefore, the total amount of data a peer uploads corresponds exactly to the amount contained in the file. Regardless out degree k of trees. Thus, the then number of peers served in t rounds is:

$$N_{tree}(C,k,N) = k^{(t-1)\frac{C}{k}} \quad (6)$$

And the time needed to completely serve N peers is:

$$T_{tree}(C,k,N) = 1 + \log_k(N) * \frac{k}{C} \quad (7)$$

Fig. 3 shows number of rounds needed to serve different number peers. As can be seen in the figure, the parallel tree outperforms the ordinary tree. The linear architecture could be a good approach if the number of peers is small, but as the number of peers grows, the performance of the linear approach becomes very poor.



(a) $C = 10^2$ (b) $C = 10^6$

Fig. 3, Performance of Linear, Tree and P Tree architecture as a function of number of clients.

Reference [7] compare and evaluate two multicast approaches for delay sensitive media distribution in P2P networks. First approach is based on cooperative m-ary trees, while the second approach is based on a cooperative mesh network. Both schemes exploit the full upload capacity of the participating peers and their proximity relation. Delivery strategy involves the full cooperation between all participating peers, including the source. In both schemes, the source splits the content into several blocks and distributes them to all requesting peers via multiple trees or a meshed network. Every peer contributes its upload capacity by being a forwarding peer into the distribution infrastructure. It evaluates both multicast approaches in terms of end-to-end delay. Performance evaluation made in Planet Lab shows that both multicast approaches achieve different content delivery delay when the participating peers are allocated far together in a wide area networks.

IV. COMPARISON

Reference [5] focuses on analyzing the service capacity of the peer-to-peer systems in transient state and steady state. The simplest mechanism of a peer-to-peer file sharing system is taken into account. Only one peer serves the system at the beginning, and every peer can only serves a single peer at one time. The analysis of deterministic model during the transient state exhibits that the average delay time is an important factor of the service capacity. In order to improve the performance, the desired file is divided into multiple chunks for downloading, which is called multi-part download. Reference [6] continues analyzing different file distribution architectures for multi-part download mechanism. Three file distribution models are presented and analyzed. For the steady state, Markov chain model gives a good analysis regarding to the transmission between different states of servers and downloaders. Reference [7] analyzes and compares both multicast approaches based on cooperative m-ary trees and cooperative mesh network achieve different content delivery delay when the participating peers are allocated far together in a wide area networks.

V. CONCLUSION

An appropriate selection of the models and the mathematic approaches simplifies the analysis of the different performance metrics of peer-to-peer system, as well as the assumptions. The search performance, the service capacity and the file distribution architectures are important issues in the performance analysis of a peer-to-peer system. However, the realistic peer-to-peer systems are very complicated. Typically the number of simultaneous downloaders of the popular files could be a few hundreds while the total number of downloaders during the lifetime of a file could be several tens or even hundreds of thousands. A lot of research areas related to the peer-to-peer technique are ongoing. As a technique under development, the peer-to-peer applications have become immensely popular and the peer-to-peer traffic starts dominating the bandwidth in the certain segments of Internet.

REFERENCES

[1] T Lin and H Wang. “Search Performance Analysis in Peer-to-Peer Networks”, IEEE the Third Conference on Peer-to-Peer Computing, 2003.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 2, Issue 9, March 2013

[2] Q. Lv, P. Cao, E. Cohen, K. Li and S. Shenker. "Search and replication in unstructured peer-to-peer networks", International Conference on Supercomputing, 2002.

[3] Qiu, D., and Srikant, R., "Modeling and performance analysis of bittorrent-like peer-to-peer networks". In Proceedings of ACM Sigcomm (Portland, OR, Aug 2004).

[4] L. Adamic, R. Lukose, A. Puniyani, and B. Huberman, Search in Power-Law Networks. Phys. Rev. E, Vol. 64, pages 46135-46143, 2001.

[5] G. de Veciana and X. Yang. "Fairness, incentives and performance in peer-to-peer networks". Forty-first Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, Oct. 2003.

[6] E.W.Biersack, P. Rodriguez, and P. Felber, "Performance Analysis of Peer-to-Peer Networks for File Distribution", QOFIS'04, Barcelona, Sept 04.

[7] Francisco de_Asis Lopez-Fuentes, **Performance Comparison of Peer-To-Peer Content Distribution Schemes** BWCCA '12 Proceedings of the 2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications Pages 88-93 IEEE Computer Society Washington, DC, USA ©2012 Pages 88-93 .

AUTHOR'S PROFILE

N. Senthil Madasamy passed Bachelor degree of Computer science and Engineering in Government College of Engineering, Tirunelveli in the year 1998. He has passed M.E. degree of Computer science and Engineering in National Engineering College in the year 2007.He is presently working as an Assistant Professor and Head with the Department of information technology in Kamaraj College of Engineering and Technology, Virudhunagar, affiliated by Anna University Chennai India, Since june 2003. He is research scholar of Anna University Tamilnadu, India since 2009. His research interests include peer to peer networks, distributed system and component based system.