An Efficient JSD-Based Search on Interest-Based Hierarchical Clustering of Overlay Networks

Hasan Bulut*, Asil Yardimci*, Sercan Demirci**, Yagiz Kaymak†, Muge Fesci-Sayit**, E. Turhan Tunali†
{hasan.bulut, asil.yardimci, sercan.demirci, muge.fesci}@ege.edu.tr, {yagiz.kaymak, turhan.tunali}@ieu.edu.tr
* Department of Computer Engineering, Ege University, Bornova, Izmir 35100 Turkey
** International Computer Institute, Ege University, Bornova, Izmir 35100 Turkey
† Faculty of Engineering and Computer Sciences, Izmir University of Economics, Balcova, Izmir 35330 Turkey

***Abstract*— In P2P networks, peers share contents, especially video files, which represent their interests. However, the underlying P2P topology may not represent this interest distribution. Thus, one important aspect of constructing an efficient P2P network is to exploit the interest similarity among peers. In this paper, we propose a hierarchical clustering mechanism for constructing an overlay network that takes interest similarity among peers into account. By measuring the similarity among interests of peers and clusters, interest-based hierarchical clusters are formed by using Jensen-Shannon Divergence metric. The clustering performance metrics, accuracy and correctness, are reported on PlanetLab. For limited keyword collections, a novel Jensen-Shannon Divergence-based search mechanism is implemented. It has been observed that the integrated mechanism provides an efficient method and better performance as compared to classical keyword-based search.**

Keywords- peer-to-peer; clustering; interest; search; Jensen-Shannon Divergence

I. INTRODUCTION

Peer-to-peer (P2P) systems and applications provide an environment for sharing contents, instant messaging, videoconferencing, distributed computation and so forth. While some of them are unstructured, others are loosely or highly structured [1]. One of the challenging problems in P2P systems is to locate the shared content. Gnutella's [2] file query method is based on, flooding while FreeNet's [3] is based on random-walk technique. Peers in structured P2P networks exchange information through the overlay network. Structured P2P networks organize peers in some way to enhance the search performance. The most common type of structured P2P networks is the Distributed Hash Table (DHT)-based systems [4][5][6].

Content shared among peers may include a wide range of file types from documents, images to video files. These contents represent peers' interests. The term interest can be considered as metadata name or description of files such as movies, videos, music or contents of files such as documents.

Clustering peers with similar interest can enhance the search performance and message complexity of the query. However, it is difficult to characterize the interest profile of a peer or categorize the shared resource, i.e., a video file.

Constructing an efficient P2P network will depend on representing the interest of the peer and exploiting the interest similarity among peers. A P2P topology that exploits this interest similarity will form interest-based clusters over the overlay network and will gather nodes with similar interests into the same cluster or neighboring clusters. This will have effect on the search mechanism implemented within the system and will improve the search performance, as number of queries, number of messages per query or false-positive rates.

In this paper, we propose a hierarchical clustering mechanism for constructing an overlay network that takes interest similarity among peers into account, and a novel JSD-based search mechanism is implemented for limited keyword collections within the hierarchical system. Jensen-Shannon Divergence (JSD) [7] metric is used to measure the interest distance between two nodes. The JSD is used in information theory to measure the divergence between two probability distributions. In our case the distribution of the keywords provided by the peer or by the description of the video file is used. Peers join the system by measuring its JSD distance to clusters starting from the top of the hierarchy. The hierarchical architecture is then exploited to direct the search using the JSD distance between the query and the clusters. Although the architecture can be used for any type of content, we emphasize using the architecture for file types such as video and music which contain very limited number of keyword rather than documents with large keyword sets. The JSD-based search will exploit the hierarchical structure of P2P network to improve the search performance in terms of number of messages per query or false-positive rates.

The clustering performance metrics, accuracy and correctness, are reported on PlanetLab [8]. It has been observed that the integrated mechanism provides an efficient method and better performance as compared to classical keyword-based search.

The paper is organized as follows: In Section 2, related work is summarized together with their pros and cons. Section 3, presents the system design developed in this study. Performance results are reported in Section 4. Finally, concluding remarks are made in Section 5.

II. RELATED WORK

There have been studies in clustering of peers in a P2P network. In these clustering approaches, clusters are formed by using metrics such as delay [9], interest [10] or both [11], [12]. In delay-based clustering, nearby peers are clustered together to decrease the delivery time. Especially delay, jitter and packet loss ratio are among major performance parameters in video streaming. These parameters are affected from the underlying physical topology. Delay-based clustering is expected to reduce the delivery time, hence positively affect the above parameters. However in delay-based clustering, peers with similar interest profile may be placed in different clusters and this will lengthen the search time. Since interest set of generated queries are expected to be parallel with peers' interest profiles, clustering peers with similar interests is expected to shorten the search time. In [10] and [12] the shared content is hierarchically classified according to some predefined classification. However, it is difficult to classify all contents. Also, a peer may hold many shared content with different interest profile from each other, which makes the clustering of peers difficult if same approach is followed to cluster. Using JSD as an interest distance metric will enable us to distinct peers relative to each other. The study presented in [11] is closer to our approach in the sense that they use JSD to characterize interest/similarity among peers. However, the overlay architecture proposed in [11] displays a flat structure and it uses Dynamic Interest Landmarks (DIL) to place the peer to an interest region or to form a new interest region. A hierarchical structured network can be more advantageous in terms of scalability and message complexity if the nearby peers are clustered together. In [13], peers with similar interests create shortcut to one another. Peers use the underlying overlay network and only create these shortcuts when they issue a query. In [14], the concept of semantic overlay clusters (SOC) for super-peer networks is introduced. The approach is based on predefined policies defined by human experts. Peers join the clusters if their metadata model matches with the cluster's policy. In [15], peers with some common properties are interconnected with a super-peer. The super-peer tries to find the target file on behalf of the peer, by forwarding the request to other super-peers by using the charge-based flooding (CBF) algorithm, a look-up protocol for distributed multimedia objects at the super-peer layer. PAIS [16] classifies contents as books, images, music, etc., which are considered as content categories or subcategories. The architecture presented in [17] is founded on interest-based superpeer paradigm, in which nodes that have a common interest will form a neighborhood relationship and elect super-peers among themselves. The super-peers resolve queries on behalf of those clients. In [17], interest is considered as generic names such as movies, music, etc. However, it is difficult to strictly classify the content or the peer which holds many semantically different contents.

There are various search techniques used in P2P systems. Centralized indexing systems such as Napster [18] has performance bottleneck at the index server. Flooding-based systems such as Gnutella [1], send query every node in the system and consumes a lot of bandwidth and peer resources. Systems that use random walk technique, such as Freenet [3], reduce flooding messages to some extent, but do not prevent wasting bandwidth with excessive messages or duplicate messages. There are variations of random walk technique such as k-walker random walk [1]. Besides random walk, there are many techniques proposed to overcome the disadvantages of flooding, which can be classified as BFS-based techniques; such as iterative deepening [19], intelligent search [20]. In these methods, file names or IDs are queried within the system.

DHT-based systems scale well, but they use (key, value) pair to route queries and retrieve files. Moreover, DHT-based systems have strictly controlled topologies and are efficient, and effective for name-based searches. pSearch [21] describes two algorithms, pVSM and pLSI, where document information, which are represented as vector of terms, is stored in DHT-based overlay networks.

In this paper, we propose a novel JSD-based search mechanism which is explained in subsection B of Section III. In this search mechanism, the query is forwarded to a cluster based on JSD measure between the request interest set and the video interest set of that cluster. So the query will be forwarded to clusters semantically closer and extra messaging will be avoided. This method gives lesser number of messages and false-positive ratio per query with a very closer hit ratio compared to flooding and keyword-based search.

III. System Design

A. *Interest-based Hierarchical Clustering*

We propose a hierarchical clustering architecture for constructing a P2P overlay network that exploits the interest similarity among peers. Clusters are formed based on peers' interest proximity. Cluster leaders are elevated to the next higher level in the hierarchy to form another cluster together with other cluster leaders. Clusters except level 0 clusters are formed by cluster leaders from the clusters in previous level.

1) *System Architecture*

In this paper, we use the term node both for peer and cluster. In our architecture, a cluster at the lowest level (level 0) is formed from peers and clusters at level 1 and above are formed from leader peers of clusters from the previous level.

Each cluster consists of at most N nodes and one of the nodes in the cluster acts as the cluster leader. N can be determined based on the message complexity within the cluster. M ($M \leq N$) neighboring clusters form another cluster at the next higher level in the hierarchy and one of the nodes in that cluster again acts as the cluster leader of the newly formed cluster. In Figure 1, a 2-level hierarchy is depicted.

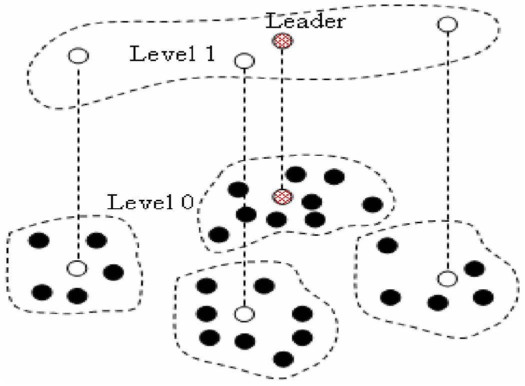

Figure 1. A 2-layer hierarchical structure

2) *Overview of JSD and Cluster Interest Set*

Let V be the set of all words in the vocabulary of all peers, $P_i(V)$ denote word frequency histogram in peer i, $v \in V$ be a word in the vocabulary and $p_i(v)$ be the percentage of the word in $P_i(V)$. Then, the Kullback-Leibler Divergence (D_{KL}) between peer i and peer j can be expressed as in [11]:

$$D_{KL} \overset{def}{=} (P_i(V) \,\|\, P_j(V)) \equiv \sum_v p_i(v) \log \frac{p_i(v)}{p_j(v)} \qquad \text{(Eq. 1)}$$

The Kullback-Leibler Divergence (D_{KL}) given in (Eq. 1) requires a workaround to prevent division by zero. For this reason, similar to [11], we have also used Jensen-Shannon Divergence (JSD) given below.

$$JSD(P_i(V), P_j(V)) = \frac{1}{2}(D_{KL}(P_i(V) \,\|\, (\frac{P_i(V)+P_j(V)}{2})) \qquad \text{(Eq. 2)}$$
$$+ D_{KL}(P_j(V) \,\|\, (\frac{P_i(V)+P_j(V)}{2})))$$

Peers' interests consist of a number of keywords where each keyword has weight associated with it, which represents the frequency of the keyword or the relative importance of the keyword from peer's view. Let I_i be the interest set of node i and w_k^i is the weight of k th word in interest set I_i.
$I_i = \{w_1^i, w_2^i, ..., w_{ni}^i\}$ set of words in *node i*

We have normalized the interest set as follows: Let $P_i(w_k^i)$ be the normalized histogram value of w_k in node *i*. Clearly, $P_i(w_k^i) = c_k^i \,/\, \sum_{k=1}^{ni} c_k^i$ where c_k^i is the weight of $w_k^i \in I_i$ and $\sum_{k=1}^{ni} P_i(w_k^i) = 1$.

Normalized histogram distribution is computed for each peer interest to be used with JSD. JSD distance between peer i and j is computed as JSD($P_i(w^i)$, $P_j(w^j)$) as given in (Eq. 2).

Let the cluster contain the nodes $i = 1, ..., n_c$. Then $I_c = \bigcup_{i=1}^{nc} I_i = \{ w_1^c, w_2^c, ..., w_{nc}^c \}$ and $P_c(w_k^c) = (1/n_c)\sum_{i=1}^{nc} P_i(w_k^c)$ where w_i^c is the cumulative of weights of the same word in each peer interest set in the cluster.

JSD distance between a peer and a cluster is also computed as JSD($P_i(w^i)$, $P_c(w^c)$), where $P_c(w^c)$ denotes the cluster interest distribution. Note that it is similar to JSD computation between two peers. In our design, a peer and a cluster is considered as a node. The cumulative interest set of peers within a cluster is normalized to represent the cluster's interest distribution.

Similar to peer-to-peer and peer-to-cluster JSD computations, cluster-to-cluster JSD computations is performed as JSD($P_{ci}(w^{ci})$, $P_{cj}(w^{cj})$), where $P_{ci}(w^{ci})$ is the interest distribution of cluster c_i and $P_{cj}(w^{cj})$ is the interest distribution of cluster c_j.

3) *Process of a New Peer Joining System*

Each peer and cluster in the system has a unique identifier. Similar to other P2P systems [9] [11], there exists a rendezvous point (RP) required for bootstrapping mechanism. A new peer A first communicates with RP and requests to join. If, currently, there is no cluster in the system, RP asks peer A to form a cluster (let ID of the cluster be C00) at level 0 and assign itself (peer A) as the cluster leader of cluster C00. Peer A forms a cluster at level 0, and another cluster (cluster ID: C10) at level 1, whose members are cluster leaders of clusters from level 0.

If there is already a hierarchical structure available in the system, RP sends peer A, the list of nodes of the cluster at the highest level. Peer A, then, measures its interest (JSD) distance between itself and other nodes in that cluster. If peer A finds a node whose JSD distance is below a threshold, peer A joins the cluster. If peer A finds a node at each level until level 0, then it joins to the cluster it finds at level 0. If peer A cannot find a node below the threshold, it forms new clusters starting from that level to level 0, and it joins the cluster where it last satisfied the threshold criteria.

4) *Cluster Splitting*

Cluster splitting is started by the cluster leader if the cluster size exceeds a certain value. The cluster leader maintains nodes' interest sets within the cluster. It first finds two farthest nodes to each other within the cluster. From these two nodes, the farther node to the cluster leader is chosen as the temporary leader of the new cluster. The other node is taken as a reference node in the current cluster. Then the leader measures every node's JSD distance to the new cluster's temporary leader and the reference node in the current cluster. If a node is closer to the new cluster's temporary leader, then it is placed into the new cluster.

Leader election algorithm is run in the new cluster. The leader of the newly formed cluster also joins the cluster at the next higher level. Cluster split algorithm is run recursively for the cluster at the higher level if the cluster size of that cluster also exceeds the threshold. The pseudo code of split algorithm is shown in Algorithm 1.

Algorithm 1: Split Cluster

```
Input: C: Current cluster
Output: A new cluster (C') or null
Vars: C': New cluster; L: Leader of C; n1, n2: nodes
(clusters)
  1: if (sizeOf(C) < upperSizeThreshold)
  2:       return null
  3: Find two farthest nodes within the cluster
  4: Assign the node closer to L to n1
  5: Add n2 to C'
  6: foreach node x ϵ C do
  7:       d1 = JSD(x, n1)
  8:       d2 = JSD(x, n2)
  9:       if (d2 < d1)  then assign x to C'
 10: end
 11: return C'
```

5) *Cluster Merging*

Cluster merging is performed if cluster size drops below a threshold. If a cluster's size drops below the threshold, it notifies its parent. The parent cluster knows the number of nodes in each of its child clusters. The cluster which requests for merge, measures its JSD distance with other cluster nodes in the cluster according to the order of list provided by its parent. If it finds a node (cluster) below a threshold it checks whether the sum of cluster sizes is also below a limit. If it is, then merging is performed. Otherwise, it continues its search until it finds one or until the end of the list. The pseudo code of cluster merging is shown in Algorithm 2.

Algorithm 2: Merging Clusters

```
Input: C: parent cluster of Ck
Output: A new cluster (C') or null
Vars: mergeList: list of clusters to be merged
newSize = sizeOf(Ck): size after merge
sortedCList: list of nodes (clusters) within C sorted
according to their size
  1: foreach Ci in C (excluding Ck)
  2:       if (newSize+sizeOf(Ci) < upperSizeThreshold)
  3:           add Ci to mergeList
  4:       newSize += sizeOf(Ci)
  5:       else break
  6: end
  7: Merge clusters in mergeList into C'
  8: return C'
```

6) Leader Election

Leader election algorithm is run within the cluster if the number of joins and leaves after the last election exceeds for a specified number. Cluster leader is the one with the minimum total distance to other nodes in the cluster. Since the cluster leader maintains nodes' interest sets within the cluster, it computes every node's total JSD distance within the cluster. The node with the lowest total distance value is chosen as the cluster leader. If the new leader is same with the current leader, nothing further is done. Otherwise, the new leader is updated in the parent cluster (cluster at the next higher level).

B. Search Mechanism

Our goal is to provide a search mechanism for items represented with a small amount of keywords (we call interest set) such as metadata of video files. Our search mechanism is based on JSD measure between the request interest set and the video interest domain. The search starts by initiating a query from a peer in a cluster at level 0. Note that level 1 and higher level clusters are virtual clusters which help nodes to place itself in an appropriate region. Search algorithm is explained next:

The query is first submitted to the peer's cluster head (CH). CH has the entire video list. So it first makes a simple search on the list to find the video. If the video file is located, then the information of which peer(s) keep(s) the video file is returned to the requestor peer. If it cannot find the video file, then it forwards the query to its parent, which is the cluster head of the parent cluster. Cluster heads, at level 1 and higher levels, do not keep any video list, because it is not feasible and manageable to keep such a big list. Instead of keeping the video list, it keeps the interest set of videos for each of its child clusters. When the cluster head receives a query from one of its child clusters, it measures the JSD distance between the query's interest set and the video interest set of the child clusters. The query is forwarded to the child cluster head with which the measured JSD distance is below a threshold. Otherwise, the query is not forwarded. Hence, excessive messaging is decreased. This reduces extra messaging and number of false-positives.

Parent cluster head waits for the query results forwarded to the child cluster heads. If the cluster head receives fail result from each of its child cluster head to which the query is forwarded, it forwards the query to its parent cluster head. If it is at the highest level of the hierarchy, search terminates as a failed search. The result is sent to the requestor peer.

If the video file is found in one of the clusters at level 0, the result is sent to the requestor peer and also its parent is notified to terminate the search as a successful search. The pseudo code of search algorithm is shown in Algorithm 3.

Algorithm 3: Search Algorithm

```
Input: q: query
Output: -
Vars: p: peer initiating the query, Cp: peer's cluster, Cq:
cluster from which the query is received, Vci: video
interest set of cluster Ci, Clist: list of the clusters to which
the query is forwarded
  1: if ( (level == 0) and (video exists) )
  2:       send the location of the video file to peer p
  3:       if  (Cp != Cq)
  4:           send success result to parentOf(C)
  5:       return
  6: end
  7: foreach Ci in C (excluding Cq) do
  8:       if ( JSD(q, Vci) < jsdThreshold)
  9:           add Ci to Clist
  10:          forward q to Ci
               // each Ci will run this algorithm upon
               // receiving q
  11: if (fail received from all clusters in Clist)
  12:      if  (parentOf(C) is not null)
  13:          forward q to parentOf(C)
  14:      else send fail to peer p
  15:      return
  16: end
  17: if (parentOf(C) is not null)
  18:      send success to parentOf(C)
  19: return
```

IV. PERFORMANCE

We have analyzed the clustering performance; how accurately the peer is located at the cluster and search performance in terms of the number of messages generated, search time and false-positives produced per query.

In our tests, peers have interest sets and may keep many video files. A video file is also represented with an interest set, which is constructed from its title, category, and some descriptive keywords from its summary. Peers' interest set is formed from similar keywords. The number of keywords is expected to be in the order of 100s. So, the interest set forwarded from one peer to another one is in the order of 100s keyword which means around KBs.

In our tests, we have set the JSD threshold value to 0.6 which is determined empirically. So a node can join a cluster if the JSD distance between the joining node and the cluster is below the JSD threshold.

A. *Clustering Performance*

We have analyzed level 0 clusters. The goal of the clustering is to locate the nearby peers (based on interest closeness) into the same cluster as much as possible. That is, the goal of this clustering performance measurement is to analyze the accuracy of this placement.

Correctness and accuracy metrics are provided in [22] to measure clustering performance. We need to adapt these metrics in order to use them to analyze our clustering performance. In addition to these two metrics, we have also measured the diameter of clusters, average distance to clusters and its standard deviation. When we measure the node's distance to its cluster, we first excluded that node from its cluster and then measured the JSD distance to the cluster. We call node's cluster *the reference cluster of the node*. We consider a selection is correct if the JSD distance between a node and its reference cluster is within γ times the distance between the node and the nearest cluster leader, where $\gamma = \dfrac{JSD\ distance\ to\ the\ reference\ cluster}{JSD\ distance\ to\ the\ nearest\ cluster}$.

In this study, we use $\gamma = 1.0$, to determine the closer cluster. $\gamma \le 1.0$ means that the node is correctly placed into the cluster. Let $x \in G_i$, then we define reference cluster as

$$\overline{G_i} = G_i - x \qquad \text{(Eq. 3)}$$

Then, $$\gamma = \frac{JSD(x,\overline{G_i})}{JSD(x,G_j)} \qquad \text{(Eq. 4)}$$

where G_j is the nearest cluster to x. Let L_i and L_j be leaders of i th and j th clusters. Similar to [22] we also define the accuracy metric as follows:

$$acc = 1 - \frac{1}{N'} \sum_{\substack{i,j\in[1\ldots k]\\ i<j}} \sum_{\substack{x\in G_i\\ y\in G_j}} \frac{|JSD(x,y) - JSD(L_i,L_j)|}{JSD(x,y)} \qquad \text{(Eq. 5)}$$

where $N' = \sum_{\substack{i,j\in[1\ldots k]\\ i<j}} |G_i||G_j|$ and x is a node in cluster G_i and y is a node in cluster G_j. We define diameter of cluster G_i as $D(G_i) = \max\{JSD(x,y)\}$ (Eq. 6) *where* $x, y \in G_i$. We also define the average distance to the reference cluster as follows:

$$avg\_dist(G_i) = \frac{1}{\#of\ nodes} \sum_{x\in G_i} JSD(x,\overline{G_i}) \qquad \text{(Eq. 7)}$$

where $\overline{G_i}$ is defined in (Eq. 3).

We have used 356 nodes in PlanetLab environment to construct the P2P system based on our hierarchical clustering algorithm explained in this paper.

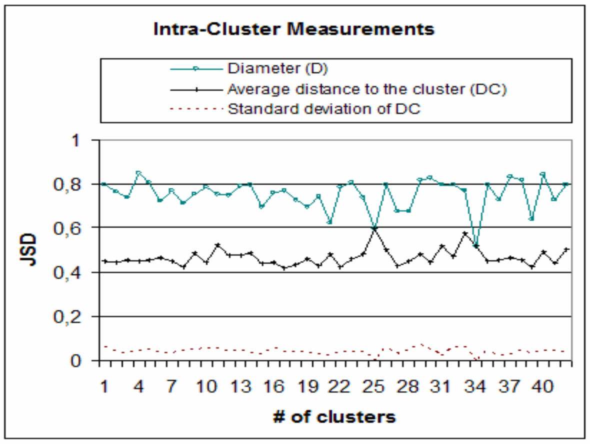

Figure 2. Intra-cluster measurement.

In Figure 2, we have shown the intra-cluster measurements. We have measured the diameter of clusters, average distances of nodes to their reference clusters and the standard deviation of the average distances. We have observed that the average of the diameters is 0,75 and average of the average distances to the clusters is 0,47, which is below the JSD threshold value. We expect the diameter be more than the threshold, because it represents the maximum distance between two nodes in the cluster.

We measured the accuracy of the P2P system we have constructed within PlanetLab environment, using 356 nodes as 86.3%. We have used (Eq. 5) to compute the accuracy. We have also measured the correctness of node placement according to (Eq. 4). Figure 3 shows the value of γ for each node in the system. Among 356 nodes, only 37 of the nodes can find a closer cluster than their reference cluster. That is, the percentage of correct placement of the system we have tested according to the criteria $\gamma \le 1.0$ is 89.6%.

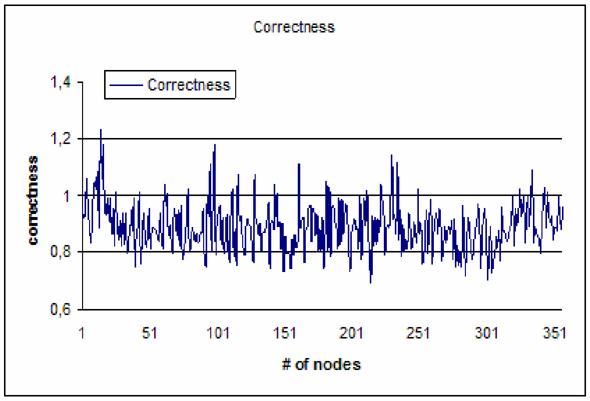

Figure 3. Correctnes of node placement

B. *Search Performance*

We have distributed video files across P2P system. Each video has a related keyword set (interest set) and an ID associated with it. Videos are distributed to peers according to their interest profiles; peers' interest profiles also reflect the interest set of the videos they maintain. Each peer may have different number of videos.

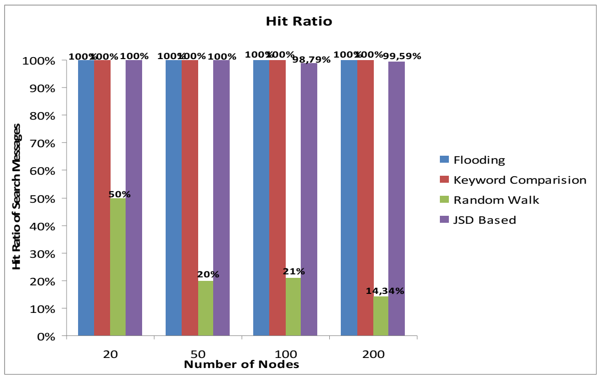

Figure 4. Hit ratio

In search performance evaluation, we have compared JSD-based search with keyword-based, flooding and random walk search. When search is at level 0, we have searched for video ID. However, in level 1 and higher levels, search is performed according to the interest set of the video file. In keyword-based search, we forward the query to a cluster as long as the video keyword set matches with the accumulated video keyword set of the cluster. In flooding-based search, the query is flooded to the entire network. In keyword-based and flooding-based search, we expect hit ratio be 100%. For JSD-based search, we have measured the JSD distance between the requested video file's keyword set within the cluster's accumulated video keyword set. If the JSD distance is less than the threshold, we forward the search to that cluster. We have constructed P2P systems with 50 nodes, 100 nodes and 200 nodes.

The hit ratios of keyword-based, flooding and random walk search and JSD-based search are shown in Figure 4. Hit ratio for JSD-based search remains closer to flooding and keyword-based search. Random walk performs a very poor hit ratio.

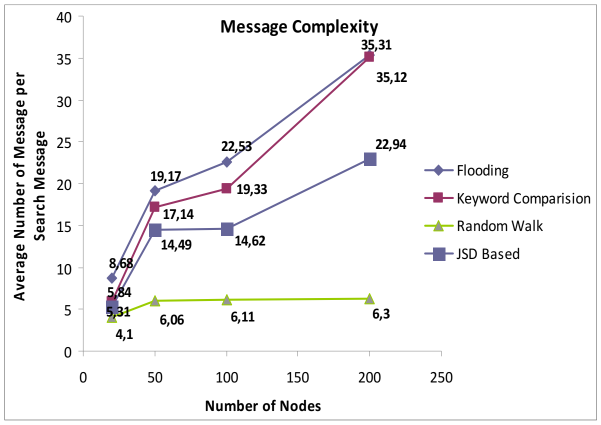

Figure 5. Average number of messages per search

The average number of messages distributed across P2P network per query is depicted in Figure 5. Random walk search has the least number of messages. Flooding and keyword-based search generates the highest number of messages per query. For 200 nodes case, JSD-based search is almost generates 37% less messages than flooding and keyword based search while maintaining over 99% hit ratio.

Comparing search times, JSD-based search performs better than flooding and keyword-based search (Figure 6).

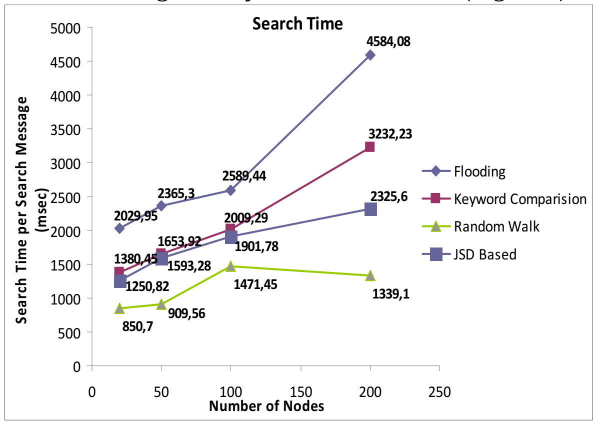

Figure 6. Average search time

As for the average number of hubs (number of nodes a query passes through), JSD-based search passes through less number of nodes (Figure 7). The difference between the average number of messages and the average number of hubs per search is that for the average number of hubs, we only count the number of nodes a search message passes through and for the average number of messages we include the messages used to notify the parent for search result in addition to the search messages.

Another feature to compare the search methodologies mentioned is the average number of false-positives; a query is forwarded to level 0 cluster with the expectation of finding the requested video within the cluster, however the requested video is not found in that cluster. The false-positive rates are shown in Figure 8; JSD-based search gives lower false-positive rate than flooding and keyword-based search.

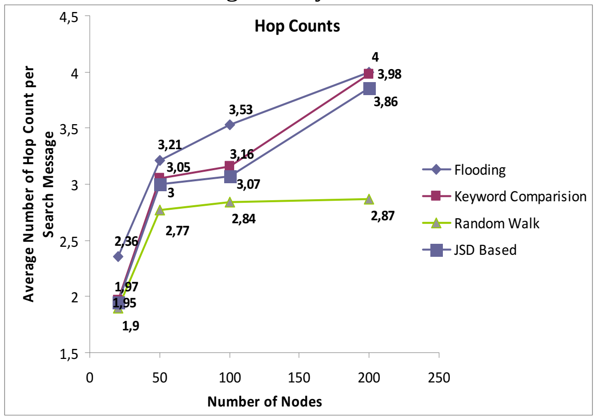

Figure 7. Average hop count per search

Random walk search gives the best result in message complexity, search time, hop count and false-positive rate. However, it produces the lowest hit ratio (14,34% for 200 nodes) while other methods gives more than 99% hit ratio.

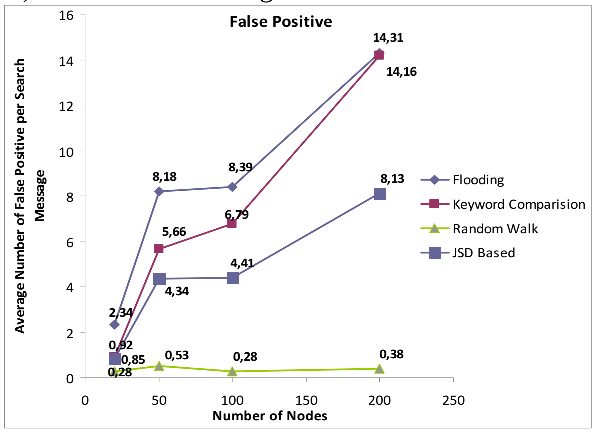

Figure 8. Average number of False-Positives per search

V. Conclusion

We proposed a hierarchical clustering mechanism for constructing an overlay network that takes account of interest similarity among peers. We have implemented a novel JSD-based search mechanism for limited keyword collections within the hierarchical system. We have provided clustering performance results that show how well the clustering mechanism works, with accuracy of more than 86% and with a correctness of more than 89% for the node settings we have used within PlanetLab environment. The overall performance of the search technique we proposed in this paper is better than random walk, flooding and keyword-based search.

VI. Acknowledgement

This work is supported by the Scientific and Technical Research Council of Turkey (TUBITAK), Project Number: 108M411.

References

[1] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, “Search and Replication in Unstructured Peer-to-Peer Networks”, in Proc. of the 16th Int’l Conference on Supercomputing, pp. 84-95, New York, USA , 2002.

[2] Gnutella 0.6, http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html (accessed August 2010).

[3] Freenet, http://freenetproject.org/ (accessed August 2010).

[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A Scalable Peer-to-Peer Lookup Service For Internet Applications”, in Proc. ACM SIGCOMM, pp. 149-160, San Diego, CA, USA, 2001.

[5] A. Rowstron and P. Druschel, “Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems,” in IFIP/ACM Int’l Conference on Distributed Systems Platforms (Middleware) , pp.329–350, Heidelberg, Germany, 2001.

[6] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An Infrastructure for Fault-Resilient [Tolerant] Wide-Area Location and Routing," University of California, Berkeley, EECS Department, Tech. Rep. UCB/CSD-01-1141, April 2001, http://www.eecs.berkeley.edu/Pubs/TechRpts/2001/CSD-01-1141.pdf (accessed August 2010).

[7] A.P. Majtey, P.W. Lamberti, and D.P. Prato, “Jensen-Shannon Divergence as a Measure of Distinguishability between Mixed Quantum States”, arXiv:quant-ph/0508138, 2005, http://arxiv.org/abs/quant-ph/0508138 (accessed August 2010).

[8] PlanetLab, https://www.planet-lab.org/ (accessed August 2010).

[9] X. Zhang, Q. Zhang, Z. Zhang, G. Song, and W. Zhu, “A Construction of Locality-Aware Overlay Network: mOverlay and its performance”, IEEE J. Select. Areas Commun., Special Issue on Recent Advances in Service Overlay Networks, vol. 22, pp. 18–28, 2004.

[10] A. Crespo and H. Molina, “Semantic Overlay Networks for P2P Systems”, Technical Report, 2002, Available at: http://www-db.stanford.edu/~crespo/publications/op2p.ps (accessed August 2010).

[11] Y. Wang, W. Wang, K. Sakurai, and Y. Hori, “On Studying P2P Topology Construction Based on Virtual Regions and Its Effect on Search Performance”, in Proc. of The 3rd Int’l Conference on Ubiquitous Intelligence and Computing, pp. 1008-1018, Wuhan, China, 2006.

[12] X. Bai, S. Liu, P. Zhang, and R. Kantola, ”ICN: Interest Based Clustering Network”, in Proc. of the 4th Int’l Conference on P2P Computing, pp. 219-226, Zurich, Switzerland, 2004.

[13] K. Sripanidkulchai, B. Maggs, and H. Zhang, “Efficient Content Location Using Interest-based Locality” in P2P Systems. In Proc. of IEEE INFOCOM, pp. 2166-2176, San Fransisco, CA, USA, 2003.

[14] A. Löser, F. Naumann, W. Siberski, W. Nejdl and U. Thaden, “Semantic Overlay Clusters within Super-Peer Networks”, in Proc. of the Int’l Workshop on Databases, Information Systems and P2P Computing, pp. 33-47, Berlin, Germany, 2003

[15] K. Watanabe, N. Hayashibara, and M. Takizawa, “A Superpeer-based Two-layer P2P Overlay Network with the CBF Strategy”, in Proc. of the 1st Int’l Conference on Complex, Intelligent and Software Intensive Systems, pp. 111-118, Washington, DC, USA, 2007

[16] S. Haiying, “PAIS: A Proximity-Aware Interest-Clustered P2P File Sharing System”, in Proc. of the 2009 9th IEEE/ACM Int’l Symposium on Cluster Computing and the Grid, pp. 164-171, Shanghai, China, 2009

[17] S. K. A. Khan and L. N. Tokarchuk, “Interest-based Self Organization in Group-Structured P2P Networks”, in Proc. of the 6th IEEE Conference on Consumer Communications and Networking Conference, p. 1237-1241, Las Vegas, NV, USA, 2009

[18] Napster, www.napster.com (accessed August 2010).

[19] B. Yang, and H. Garcia-Molina, “Improving search in peer-to-peer networks”, in Proc. of the 22nd IEEE Int’l Conference on Distributed Computing Systems, pp. 5-14, Vienna, Austria, 2002.

[20] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, “A Local Search Mechanism For Peer-To-Peer Networks”, in Proc. of the 11th ACM Conf. on Information and Knowledge Management, pp. 300-307, McLean, VA, 2002.

[21] C. Tang, Z. Xu, and M. Mahalingam, “pSearch: Information Retrieval in Structured Overlays”, ACM SIGCOMM Computer Communication Review, vol.33, no.1, pp. 89-94, 2003.

[22] X. Zhang, J. Liu, Q. Zhang, and W. Zhu, "gMeasure: A Group-Based Network Performance Measurement Service For Peer-To-Peer Applications", IEEE Global Telecommunications Conference, vol.3, no.17-21, pp. 2528- 2532, Taipei, Taiwan, R.O.C., 2002.