Scalable Self-Organizing Structured P2P Information Retrieval Model Based on Equivalence Classes

Yaser A. Al-Lahham and Mohammad Hassan
Faculty of Science and Information Technology, Zarqa University, Jordan

Abstract: *This paper proposes a new autonomous self-organizing content-based node clustering peer to peer information retrieval (P2PIR) model. This model uses incremental transitive document-to-document similarity technique to build Local Equivalence Classes (LECes) of documents on a source node. Locality Sensitive Hashing scheme is applied to map a representative of each LEC into a set of keys which will be published to hosting node(s). Similar LECes on different nodes form Universal Equivalence Classes (UECes), which indicate the connectivity between these nodes. The same LSH scheme is used to submit queries to subset of nodes that most likely have relevant information. The proposed model has been implemented. The obtained results indicate efficiency in building connectivity between similar nodes, and correctly allocate and retrieve relevant answers to high percentage of queries. The system was tested for different network sizes and proved to be scalable as efficiency downgraded gracefully as the network size grows exponentially.*

Keywords*: Peer-to-Peer systems, Information retrieval, Node clustering, Equivalence class, Mapping, Incremental transitivity.*

Received February 5, 2012; accepted May 22, 2012

1. Introduction

Peer to Peer (P2P) systems were proposed as distributed file sharing systems of large set of collaborating nodes. Each node plays a role as information provider (server) or consumer (client) at the same time [19].

P2P models are generally designed as a layer on-top-of the Internet. A P2P system provides different services such as resource sharing, robust routing and redundant storage. It was considered as an appropriate alternative to the client-server architecture for some application domains [1].

In the meantime, scalability is a necessary requirement of P2P systems. Earlier P2P systems attempted to achieve scalability by flooding queries randomly to neighbours, which swamped the network with unanswered queries. Time to Live Limit (TTL) was used to forward queries that reduced the chance to find relevant answer(s). Distributed Hash Tables (DHTs) were proposed as deterministic models of file allocation and lookup. DHTs assign each keyword a numeric key; a node that has closer identifier to a key allocates a pointer to the file that contains the keyword. Therefore, each file could be pointed by many nodes. DHTs efficiently allocating a single keyword but it encounters larger communication cost when attempting to find answers to queries of several keywords [24]. Peer-to-peer information retrieval (P2PIR) systems proposed to incorporate full text retrieval to widen the keyword allocation, such that document (or topic) allocation is used instead of a single keyword. P2PIR systems are implemented by adding a new layer of information retrieval on top of P2P architecture [2].

As large numbers of nodes joining the P2PIR systems, information allocation and lookup is enhanced by creating direct links between similar nodes in a P2PIR system, because a request could be directed to a subset of nodes that most likely having relevant information [3, 4, 5, 7, 10].

Many models of P2PIR have been proposed, such models could be categorized into: extensions to simple keyword search, self-organizing P2PIR systems, and content-driven node clustering approaches.

Extensions to keyword search: for example, Schmidt and Parashar [20], and Loguinor [18] defined keywords as coordinates of a multi-dimensional space, and re-indexed the documents as points according to the new defined space. While self-Organizing P2PIR approaches incorporate a predefined semantic structure, each node is responsible to prepare its local indices, grouping them into semantic groups and register them in order to connect to the suitable subset of nodes in the predefined semantic structure. For example, Crespo and Garcia-Molina in [8] proposed SONs (Semantic Overlay Networks). Shen in [21] built a P2PIR overlay depending on the hierarchical summary indexing. Stoica in [22] Proposed the pLSI algorithm to build the (pSearch) search engine.

Node clustering approaches aggregating nodes according to their semantic attributes. The network construction involves three major tasks: extracting the semantic properties of a node to determine the proper location in the semantic space, grouping peers that have similar semantics into node-clusters, and finally interconnect clusters to form the overlay. Li in [17] presented an overlay network, namely semantic small world **(**SSW**)** that facilitates semantic based search in P2P systems. Zhua in [26] proposed an architecture of IR on semi-collaborated P2P networks.

Bawa in [3] proposed SETS: Search Enhanced by Topic Segmentation, in which the overlay is partitioned into topic segments, such that nodes having similar documents are most likely belong to the same segment. Crespo and Garcia-Molina in [8] proposed a distributed connectivity-based decentralized node clustering (CDC) model, where nodes are supervised by originator nodes. Bhattacharya in [4] focused on supporting similarity queries for text information retrieval in DHT based overlay networks. Jin in [16] proposed "Query routing for semantic overlays", Support Vector Machine is used to establish the semantic overlay.

Each node in a P2PIR is given an identifier that is generated using the logical relationship (not the geographical) between nodes. Some P2P systems identify peers as points in a predefined geometrical space (called the identifier space), in such systems data keys are deterministically selected from the same identifier space, these systems are called *structured* P2PIR systems. On the other hand *unstructured* P2PIR systems randomly select both node identifiers and data keys.

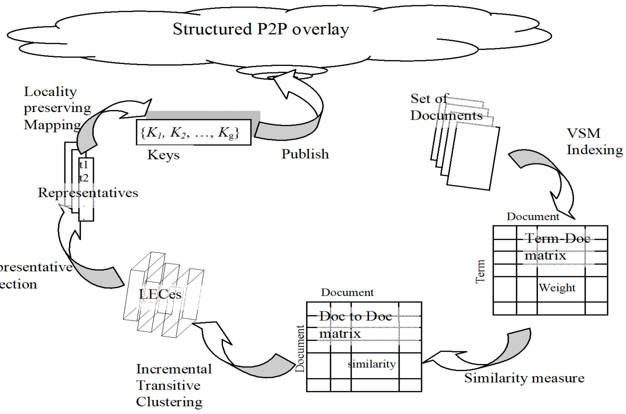


Figure 1. A general view of content indexing and publication used in this researc.

This paper presents a scalable self-organizing model such that node-to-node connectivity is locally established using node content. Node aggregation is not being explicitly monitored by nodes of special roles. Figure 1 presents a general view of content indexing and publication. The proposed model preserves P2PIR scalability throughout aggregating nodes that have similar content into a same connectivity group; then directing (mapping) queries to the groups of nodes that is most likely having relevant information. This objective could be achieved by a process of four steps: (1) Partition the local content of each node into subsets of documents (Local Equivalence Classes LECes), using incremental transitive document clustering [12]. (2) Representing each LEC by a vector of features (terms). (3) Map a LEC representative into a key, using locality preserving mapping function; i.e. similar LECes have closer keys. And (4) publish (send a copy of) the LEC representative to a P2P node that owns its key. According to the similarity between published LECes, nodes can autonomously establish connections to other similar nodes (each set of similar LECes on connected nodes forms a Universal Equivalence Class UEC). Comparison with other approaches will be presented in the evaluation sections.

The rest of paper is organized as follows: section 2 presents the proposed process of LEC formation, representation, and mapping. Section 3 explains the scenario of building UECes. Section 4 demonstrates the lookup over the proposed model. Section 5 explains the replication process and system maintenance. The implementation and evaluation of the proposed system are discussed in section 6. Finally section 7 presented the conclusion.

2. Local Equivalence Classes (LECes)

A source node (data provider) partitions its local content into LECes of documents, and selects a representative for each (LEC) that is formed using incremental transitive document-to-document similarity as in [12].

2.1. LEC Representative

LEC representative is formed out of terms that selected from documents' vectors of the incremental-transitive cluster hierarchy. We will adapt the representative-selection method that was introduced in [13], such that the terms of a representative vector are descending sorted according to weight, and the representative vector is reduced to a fraction of its original size (top-n terms are selected). This selection method ensures that the average similarity between documents' vectors in a cluster and its reduced representative is slightly downgraded even when the size is reduced to one third of its original size [13].

2.2. Mapping LEC Representative to Key Space

Similar LECes generated by different source nodes should be mapped to host node(s) that own their keys. To achieve this goal, variant of Locality Sensitive Hash (LSH) [9] is used.

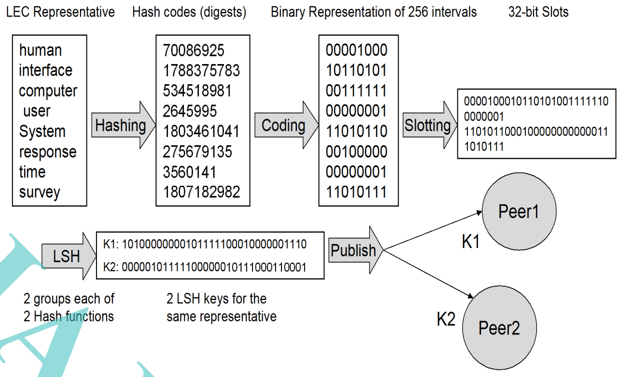


Figure 2. LSH Locality preserving mapping scheme.

LSH maps nearest points in a space to closer images in the target space with high probability [4, 11]. We will adapt a LSH scheme that introduced in [14]. This scheme groups LSH out of a family of approximate min-wise independent permutations [5]. The LSH Key generation process accepts a list of terms as LEC representative, encoding each term by a universal hash function, then converting a sequence of encoded terms into a stream of bits, generating a subsequence of bits (each subsequent has a size equals to the size of the target key), and apply the LSH scheme to get a number (g) of keys for each LEC. An example showing the steps of the mapping process is illustrated in Figure 2.

3. Universal Equivalence Classes (UECes)

Nodes that have similar LECes are connected forming disjoint subsets of similar LECes that are called universal equivalence classes (UECes). The term “disjoint” means that a set of nodes that are spanned by a UEC have more connectivity than to other nodes of other sets. Two requirements are needed to build the UECes:

(1) a set of LECes on all nodes:

$$EQ = \{Nj\_Li: N_j \in \aleph, \text{ and } Li \in N_j\} \qquad (1)$$

Li is a LEC owned by the node N_j, and $\aleph$ is the set of all nodes in the system.

(2) “Connectivity” ($\wp$) equivalence relation that used to form the universal equivalence classes.

Definition **1**- "Connectivity relation ($\wp$)": two nodes N_1 and N_2 are connected if they have at least two similar LECes, or they are hosting two similar LECes:

$N_1 \wp N_2 = \{(N1\_Li, N2\_Lj): sim(Li, Lj) \geq s_r$, or:

$$N2\_Lj \in ST_{N1}, \text{ and } N1\_Li \in ST_{N2}\} \qquad (2)$$

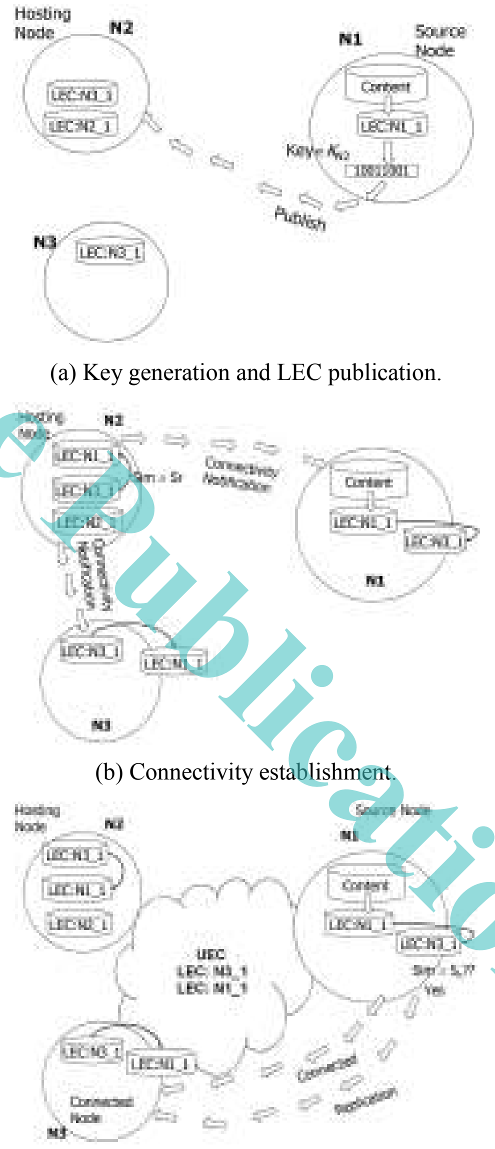
ST_{N1}, ST_{N2} are the semantic tables of nodes N_1, N_2 respectively, and *sim* is the Jaccard similarity. *N1_Li*, and *N2_Lj* are two LECes on nodes N1, N2 respectively.

$\wp$ could be proved to be an equivalence relation as follows:

$\wp$ is reflective, since each node has all of its LECes connected to themselves. $\wp$ is symmetric, because if $N1 \wp N2$ (*N1* connected to *N2*) then $\exists$ *N2_Lj* and *N1_Li*, such that $N2\_Lj \in ST_{N1}$, consequently, according to definition 1, $N1\_Li \in ST_{N2}$, which determines $N2 \wp N1$ (*N2* is connected to *N1*). $\wp$ is Transitive since source nodes publish each of its LECes to hosting nodes, so, it is directly connected to a set N_c of hosting nodes.

$$N_c = \{N_{h1}, N_{h2}, \ldots, N_{hg}\}, \text{ and } \forall N_{hi} \in N_c, \quad \exists\ N\_Li \in ST_{Nhi}, \ i=1,.., g \qquad (3)$$

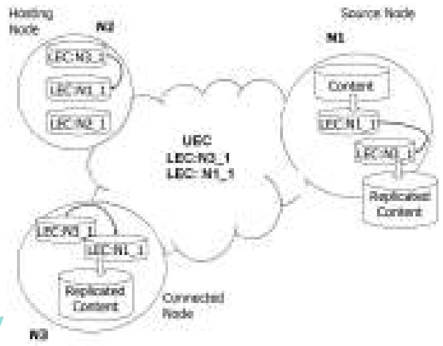
Each hosting node $N_{hi} \in N_c$ transitively connects the source node *N* to a set of other source nodes that have similar LECes to the LEC owned by *N*. A connectivity set of a UEC, denoted as Ω(UEC), is the subset of nodes that have at least one similar LEC to a LEC in Ω(UEC), a node could belong to more than one connectivity set. Detailed UEC formation process is illustrated in Figure 3.



(a) Key generation and LEC publication.

(b) Connectivity establishment.

(c) Similar LEC found, and content replicated.



(d) A UEC connects nodes.

Sim: similarity, Sr: similarity threshold, *Sh*: replication threshold, *KN2*: Keys managed by node: *N2*

Figure 3. UEC formation process.

4. Lookup for a User Request

Queries are not submitted to/by special nodes, so any node could issue and manipulate it. A query is treated as a new document on the issuing node. The issuing node searches locally for a relevant LEC, if it finds any, it expands query by adding terms of the relevant LEC to it, then it forwards query to all of the nodes that are connected through that LEC.

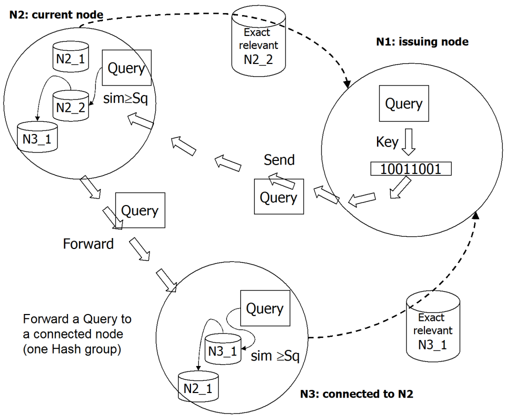
In case the issuing node has no relevant LECes, it maps the query vector into a (Qkey) using the LSH scheme, and publishes it to a set of hosting nodes.

A node that receives a query is called a current node, so we have a set of current nodes:

$$Cur=\{N_i: Qkey \in K_{Ni}\},\ i=1, .., g \qquad (4)$$

g: number of hash functions' groups.

Since LSH scheme maps similar keys to nearby nodes with high probability, there is a chance to find relevant LECes hosted by some members of *Cur*. If the relevant LEC is *owned* by the current node, then it returns it to the issuing node, and forwards the query to all of its connected nodes. If the relevant LEC is *hosted* by the current node, then it forwards the query to the owner source node to manipulate the query.



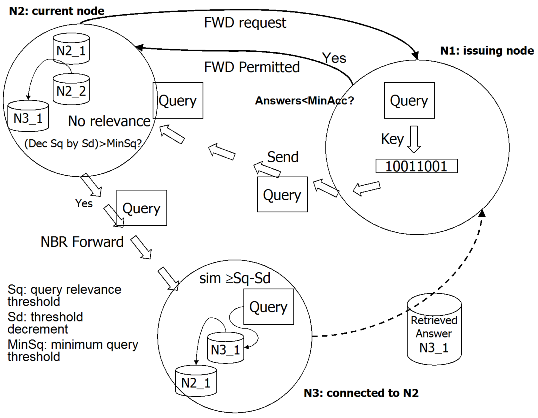


Figure 4. Query submission and forward procedure.

```
Algorithm:  manipulateQuery(Query Q)
get all relevantLECes to Q from the local semantic table
If there is no relevant LECes then
      Send a FORWARD_REQUEST to the issuing node
       If FORWARD_PERMITTED then
          Forward Query To P2P Neighbours
       Else
          Discard Q
      End If
Else
      for each LEC in relevantLECes do
          if the LEC owned by the current node then
             Send a QUERY_ANSWER to issuing node
             record LEC on the query answer list
        Else        // forward query to connected nodes
             if the owner Node is not visited before  then
                Send QUERY_FORWARD to the owner node
          end if
      end if
   next LEC
  end if
end.
```

In case no LECes are found relevant to a query, the current node forwards it to its neighbouring nodes in the P2P overlay (after permitted by the issuing node). A search controller must be set, such that each current node saves a *query-node list* (*NodeIds* of all pre-visited nodes) to prevent entering into a loop. Figure 4 illustrates an example of query submission, and query manipulation procedure is presented in the algorithm: *manipulateQuery*.

5. Replication and UEC Maintenance

Controlled replicas of content on selected nodes improve the efficiency and robustness of the P2PIR system [4, 6]. In our model, each node of a connectivity set of a UEC could cache the content of LECes having similarity ≥ replication threshold *(Sh)* from connected nodes. Semantic table (global index) –also- could be replicated on neighbour nodes in a P2P overlay.

Content replication: LEC content is replicated on a node if it has a LEC of high similarity to the replicated one. Consequently, nodes that have high similar LECes are most likely having same interests that forms an interest group, which is considered a good tool for building efficient P2PIR systems [6].

Semantic table replication: Nodes in structured P2P overlays keep routing information about number of nodes in its neighbourhood. In our model, each node keeps a copy of semantic table(s) of its neighbour hosting node(s). Semantic table replication preserves the availability of the system. It could also be helpful during lookup; since a query could be sent to nodes in a same neighbourhood of nodes that host similar LECes.

5.1. UEC Maintenance

This section addresses the issues of departing nodes; either normally or by fault. P2PIR model requires to validate the node-connectivity status whenever a considerable change occurred on the UECes, the term considerable could be: (1) a constant period of time elapsed since the last validation, (2) a predefined number of departs or joins, (3) newly added LECes having *sim* ≥ threshold, or (4) newly replicated LECes.

5.1.1. New Additions

New nodes: A new node obtains its semantic table while connectivity establishment. The neighbouring nodes should copy, to the new detected node, the entries of its semantic tables that store LECes having keys closer to the identifier of the new node.

LEC Change (new documents added): new documents must be indexed and attached to a LEC they have higher similarity to. This causes new terms to be added to the LEC representative. If a major change occurred to the LEC representative, then it becomes necessary to re-calculate the LEC key, and the connectivity to other LECes on the hosting nodes. The term (major change) means that the ratio of newly added terms exceeds a predefined limit.

New LECes added (Node re-clustering): Newly added documents could form new LECes that establish new connections to other nodes. The new connectivity status could re-organize the UECes, such that a subset of connected nodes is involved in the re-arrangement procedure. This procedure makes it possible to maintain clusters without needing to re-organize the overall system, since each node can locally maintain its connections without contacting any central node.

5.1.2. Nodes' Withdrawals

Nodes withdrawal imposes other procedures to preserve the connectivity within a connectivity set, since neighbouring nodes within the underlying P2P network are not necessarily belong to same connectivity set.

Normal withdrawals: leaving node must alert other nodes in the connectivity set, such that entries in its semantic table are retained by its direct neighbour in the P2P overlay, where the term (direct) is determined according to the underlying P2P topology. A leaving node alerts other node that has replicated the content of its LECes, which broadcasts acknowledgment to every other node in the connectivity set. For example let N_1, N_2, $N_3 \in \Omega(U_1)$ are three connected nodes, N_1 leaves the system, and N_2 is selected by N_1 as the most connected node, suppose N_1 has a LEC: $N_1\_L_x$, *so* N_2 becomes the new owner of $N_1\_L_x$, N_2 replicates the content of $N_1\_L_x$. N_2 sends an acknowledgement to N_3, in order to change the *NodeId* of all entries related to $N_1\_L_x$ to be: $N_2\_L_x$. If no replications of a LEC found, then all of its entries in the semantic tables of all nodes are cancelled.

Faulty node withdrawal: When a faulty node is detected, all connected nodes to that node must cancel its connections in order to preserve the consistency of node-to-node connectivity. Faulty connection could be resolved locally on each member of $\Omega(UEC)$, since the connectivity information is recorded as entries in the semantic table of each node in the connectivity set.

6. Implementation and Evaluation

A Java application called (SemanticNode) has been developed. This application defines the operations of the proposed P2PIR system. The SemanticNode is registered as an application on the *PlanetSim* simulator [10]. A testing data set of 10000 documents is randomly selected from the Reuters21578 collection. The selected documents are randomly distributed into 100 groups, where a group represents the content of one source node. The Chord [22] overlay (implemented by *PlanetSim*) is selected as the underlying P2P overlay. Basically, a network of 1000 nodes is constructed, where each node is assigned an identifier of 64 bits. A number of nodes (mostly 50) were selected randomly among the network to initialize their semantic tables[1]. During the initialization phase, each SemanticNode application publishes its LECes to hosting nodes, then building their connectivity sets.

6.1 Connectivity Evaluation

Connectivity evaluation aims to examine the "aggregation of connected nodes into connectivity sets of UECes".

Evaluation procedure: Prior to the simulation, the actual node-to-node connectivity is computed, and used as controlling criterion. While the LEC-to-LEC Jaccard similarity is computed in a similar way as they are in a single centralized node. The evaluation procedure includes: First, for each source node,

[1] It is not necessary for the content of all nodes to be available at the bootstrapping. This is only an example state.

determine the set of actually-connected nodes: R_{Ni}. Then, run the SemanticNode application on the selected set of source nodes. Step that follows is determining the directly-connected nodes C_{Ni} (discovered by the SemanticNode), and the *transitive connected* nodes.

In order to determine the set of actually-connected nodes, $|R_{Ni} \cap C_{Ni}|$ should be determined for each node, which represents the number of actually-connected nodes that are discovered by the SemanticNode.

Two factors are used to determine the effectiveness of the connectivity relation ($\wp$):

- The first is the ratio of the discovered directly-connected nodes to the actually-Connected nodes ($|R_{Ni} \cap C_{Ni}|/|R_{Ni}|$) that examines the symmetry of ($\wp$).
- The second is the ratio of nodes that have transitive connectivity that examines the transitivity of ($\wp$).

Symmetric connectivity evaluation: The connectivity ratio $|R_{Ni} \cap C_{Ni}|/|R_{Ni}|$, was recorded for 50 source nodes out of a network of 1000 nodes for (low, medium, and high) degrees of connectivity. The distribution of numbers of source nodes over 10 intervals is plotted in Figure 5, it shows that more connected nodes are being discovered as the degree of connectivity increased. The average connectivity ratios are: 35%, 43%, 54% for low, medium and high connectivity degrees respectively. This result gives more evidence that the first factor is accomplished, even when smaller number of hash functions' groups is used. The symmetric property of the direct connectivity among nodes was also tested by examining the number of symmetric similar LECes. An experiment of 50 source nodes shows that about 60% of connected nodes have symmetric connectivity with more than 90% of their similar LECes, see Figure 6.

Transitive connectivity evaluation: two experiments were performed using the same system parameters. The set of transitively-connected nodes is determined for each source node, and the distribution is plotted in Figure 7(b). The cardinality of the direct-connectivity set distribution is plotted in Figure 7(a). The connectivity of most of the source nodes is enhanced, since connected (and hosting) nodes became reachable by at most two hops. It is clear that transitivity makes the curve of the direct connectivity Figure 7(a) shifted toward larger cardinality intervals. The results above give more evidence that the proposed model (even when using a lower degree of connectivity) meets the transitivity condition. So the connectivity relation $\wp$ successfully forms the universal equivalence classes that span similar source (and hosting) nodes.

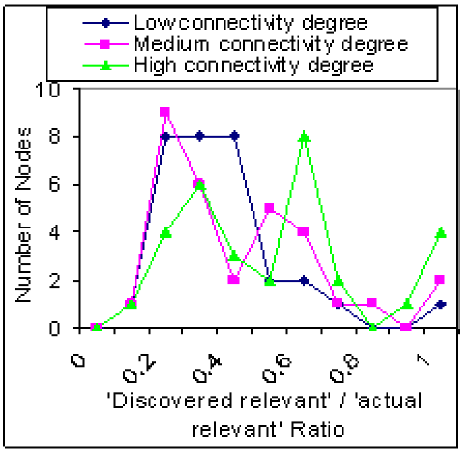


Figure 5. Distribution of connected nodes.

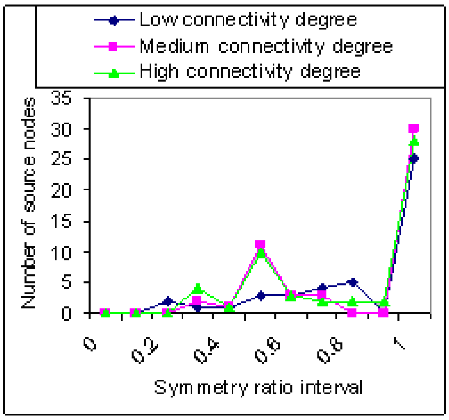
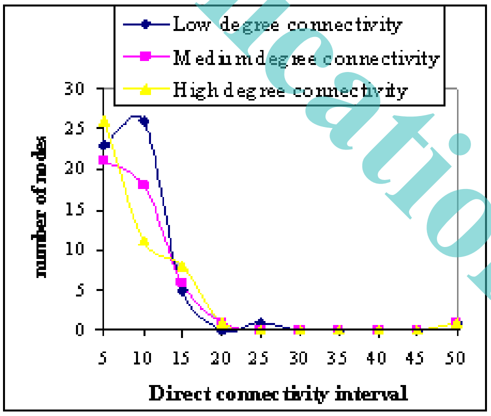


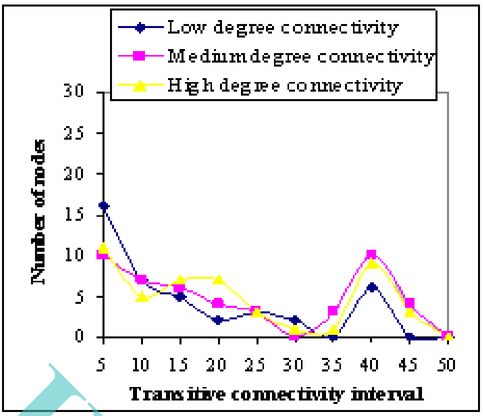
Figure 6. Symmetric vs. Non-Symmetric LECes.

6.2. Lockup Evaluation

There are three types of query answers: (1) No answer or missed query: no relevant answer within the maximum allowed number of hops is found. (2) Exact relevant answer: a query got a relevant answer directly from one hop. (3) Retrieved query answers: a query got answers after being forwarded through more than two successive hops.



(a) Direct connectivity to other nodes.



(b) Transitive connectivity to other nodes.

Figure 7. Transitivity test of ($\wp$)

The search process was applied on 431 random queries, and evaluated according to the following parameters:

- Mapping: number of hash functions' groups used to map both LECes and queries.
- Replication: The presence or the absence of content replication.
- Connectivity degree: (low, medium, or high).

The first experiment uses (1, 2, and 4) hash functions' groups to map random queries. This experiment shows that mapping has a significant effect on query answering; where the number of responses (both exact relevant and retrieved answers) is enhanced when more hash groups are used; see Figure 8(a).

The second experiment examines the effect of content replication and the degree of connectivity on the retrieval efficiency. This experiment indicates that higher degree of connectivity and content replication enhances the retrieval efficiency. Queries that found at least one relevant LEC are plotted versus the number of hash functions' groups in Figure 8(b). Ratio of queries that got answers were: 74%, 68%, and 46% using 4, 2, and 1 hash group respectively.

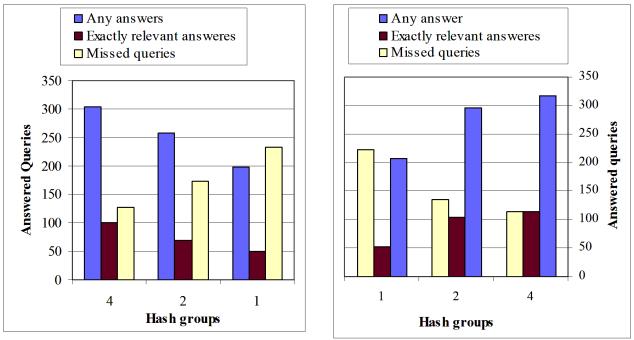
These results are acceptable according to other models. For example, Gupta model [11] used random queries, and mapping scheme of 5 hash groups each of 20 functions, 38% of queries had a match, while the average similarity between queries and partitions was 0.9 to 1.0.

In our model, we got 23% of queries having an exact relevant answer, and 74% of queries have a retrieved answer, using a mapping scheme of 4 hash groups and an average similarity of 0.5. Yee and Frieder [23] returned answers of (1060 to 1213) queries out of 10000; i.e. 11% to 12% of queries have a match. Cohen in [7], got about 22% of queries having relevant answers in case of 1% of nodes having data, Reuters21578 have been used for testing. Zhou et al. [25], got 55% average recall ratio by using an average of 8 hops, our model returns 42% of the exact relevant answers for an average number of hops=2.75.

Table 1 presents the number of relevant LECes of 431 queries against the number of hops for two and four hash functions' groups. It is clear that the ratios of the exact relevant answers are better for 2 or less hops.

6.3. System Scalability Evaluation

The objective is testing the relation between network size and system response. Network size used is growing exponentially: 1000, 2000, 4000, 8000, and 16000 nodes.



(a)*Without content replication*

(b)*With content replication*

Figure 8. Answered queries vs. hash groups.

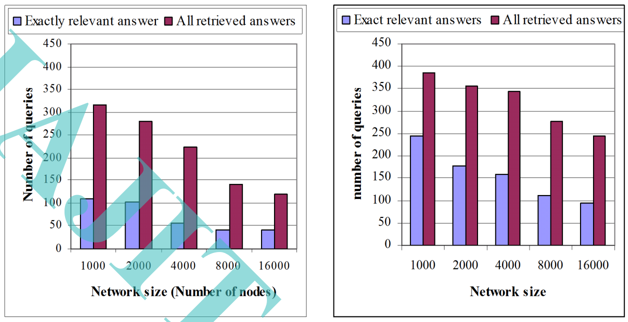
Table 1. Retrieved LECes vs. number of hops.

	Two Hash Groups			Four Hash Groups		
Hops	**Ret**	**Rel**	**Ratio**	**Ret**	**Rel**	**Ratio**
0	44	9	0.21	24	15	0.63
1	208	81	0.39	172	92	0.53
2	228	22	0.10	148	34	0.23
3 to 5	535	22	0.04	525	41	0.08
6 to 10	905	17	0.02	937	22	0.02
>= 11	1437	23	0.02	2435	41	0.02

All experiments were performed using two groups of hash functions to map both LECes and query submission, without content replication. The number of queries that got at least one answer is recorded for the three answer types, and plotted as in Figure 9.

Figure 9(a) shows that number of queries that have exact relevant answers is not affected when upgrading from 1000 to 2000 nodes. Whereas it decreases rapidly while upgrading from 2000 and 4000 nodes before it returns back to graceful decreasing for the rest of the network sizes.

This result could be justified as follows: as the network grows; the subset of keys that are owned by a node becomes smaller. So, the chance for two similar LECes (or a query) to be mapped to the same node becomes smaller. Consequently, relevant LECes to a query could be mapped to two nodes in a neighbourhood, making the chance to forward a query to neighbouring (not connected) node more probable. Therefore a query will find a retrieved answer instead of exact relevant answer. In our model, the average downgrading is about 19% as the network size grows exponentially, while other researchers, for example Chen in [15], got an average downgrading ratio of 37% for a linearly growing network. Experiments show also that the system scalability is better preserved using more hash groups; as could be observed in Figure 9(b).



(a) *Queries mapped by 2 hash groups* (b) *Queries mapped by 8 hash groups*

Figure 9. Answered Queries vs. network size.

7. Conclusions

The proposed model of P2PIR system has efficiently built a content-based connectivity between nodes. The model is better preserving the system scalability, where the number of queries that got an exact relevant answer downgrades, in average, around 19% when the size of the network is doubled. The proposed model directs queries to small subsets of connected nodes that are most likely having relevant answers; i.e. 74% of queries got at least one answer, even when the implemented system used only 12% of the minimum number of required hash groups (according to the LSH scheme).

The efficiency of the proposed model could be enhanced by increasing the degree of connectivity. Connectivity increases the ratio of queries that could have exact relevant answers using at most two hops. Efficiency could also be enhanced using a higher number of groups for the LSH scheme, and content replication.

These results give evidence to accept the assumption of connectivity as pseudo equivalence relation, and consequently accept the assumption of sets of connected nodes as universal equivalence classes.

References

[1] Aberer K., Hauswirth M., and Schmidt R., “Improving Data Access in P2P Systems,” *IEEE Internet Computing,* vol. 6, no. 1, pp. 58 - 67, January 2002.

[2] Aberer K., Klemm F., Rajman M., and Wu J.: “An Architecture for Peer-to-Peer Information Retrieval,” in the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval Toronto, Canada, 2003, pp. 17-24.

[3] Bawa M., Manku G. S., and Raghavan P.: “Sets: Search enhanced by topic segmentation,” in the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’03), Toronto, Canada, 2003, pp. 306-313.

[4] Bhattacharya I., Kashyap S. R., and Parthasarathy S.: “Similarity Searching in Peer-to-Peer Databases,” in The 25th IEEE International Conference on Distributed Computing Systems (ICDCS), Columbus, OH, 2005, pp. 329-338.

[5] Broder A. Z., Charikar M., Frieze A. M., and Mitzenmacher M.: “Min-Wise Independent Permutations,” Journal of Computer and System Sciences, vol. 60, no. NA, pp. 630-699, 2000.

[6] Chirita W. N. P. A., Scurtu O.: “Knowing where to Search: Personalized Search Strategies for Peers in P2P Networks,” in SIGIR Workshop on Peer-to-Peer Information Retrieval, Sheffield, 2004.

[7] Cohen E., Fiat A., and Kaplan H.: “Associative Search in Peer to Peer Networks: Harnessing Latent Semantics.,” The International Journal of Computer and Telecommunications Networking vol. 51,no. 8 pp. 1861-1881, 2007.

[8] Crespo A. and Garcia-Molina H.: “Semantic Overlay Networks for P2P systems”, vol. 3601/2005 Lecture Notes in Computer Science - Agents and Peer-to-Peer Computing. Berlin / Heidelberg: Springer, 2005.

[9] Datar M., Immorlica N., Indyk P., and Mirrokni V. S.: “Locality-Sensitive Hashing Scheme Based on p-Stable Distributions,” in the twentieth annual symposium on Computational geometry (SCG’04), Brooklyn, New York, USA 2004, pp. 253-262.

[10] Garcia P., Pairot C., Mondejar R., Pujol J., Tejedor H., and Rallo R.: “PlanetSim: A New Overlay Network Simulation Framework,” in the 19th IEEE International Conference on Automated Software Engineering (ASE 2004), Workshop on Software Engineering and Middleware (SEM 2004), Linz, Austria, 2004, pp. 123-136.

[11] Gupta A., Agrawal D., and Abbadi A. E.: “Approximate Range Selection Queries in Peer-to-Peer Systems,” in the CIDR Conference, 2003, pp. 254-273.

[12] Hasan Y. A. M., Hassan M., and Ridley M. J.: “Incremental Transitivity Applied to Cluster Retrieval,” The International Arab Journal of Information Technology, vol. 5, no. 3, July 2008, pp. 311-319.

[13] Hasan Y. A. M., Hassan M., and Ridley M. J.: “Efficient Approach for Building Hierarchical Cluster Representative,” IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.1, January 2011, pp. 178.

[14] Hassan M. and Hasan Y.: "Locality Preserving Scheme of Text Databases Representative in Distributed Information Retrieval Systems," in the second International Conference of Networked Digital Technologies NDT, Prague, Czech republic, July 2010, pp. 162.

[15] Jin H. and Chen H.: "SemreX: Efficient search in a semantic overlay for literature retrieval," Future Generation Computer Systems, vol. 24 no. 6 pp. 475-488, 2008.

[16] Jin H., Ning X., Chen H., and Yin Z.: "Efficient Query Routing for Information Retrieval in Semantic Overlays," in the 2006 ACM symposium on Applied computing SAC'06, Dijon, France, 2006, pp. 1669-1673.

[17] Li M., Lee W.C., and Sivasubramaniam A.: "Semantic Small World: An Overlay Network for Peer-to-Peer Search," in the 12th IEEE International Conference on Network Protocols (ICNP'04), 2004, pp. 228-238.

[18] Loguinor D., Kumar A., Rai V., and Ganesh S.: "Graph-Theoretic Analysis of Structured P2P System: Routing Distances and fault resilience," IEEE/ACM Transactions on Networking (TON), vol. 13,no. 5 pp. 1107-1120, 2005.

[19] Lua E. K., Crowcroft J., Pias M., Sharma R., and Lim S., "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes," *IEEE Communications Survey and Tutorial,* vol. 7, no. 2, pp. 72 - 93, March 2004.

[20] Schmidt C. and Parashar M.: "Enabling Flexible Queries with Guarantees in P2P Systems," IEEE Internet Computing, vol. 8, no. 3 pp. 19-26, May/June 2004.

[21] Shen H. T., Shu Y., and Yu B.: "Efficient Semantic-Based Content Search in P2P Networks," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 7 pp. 813-826, July 2004.

[22] Stoica I., Morris R., Karger D., Kaashoek M. F., and Balakrishnan H.: "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in ACM SIGCOMM'01, San Diego, California, USA, 2001, pp. 17-32.

[23] Yee W. G. and Frieder O.: "On Search in PeertoPeer File Sharing Systems," in SAC'05, Santa Fe, New Mexico, USA, 2005.

[24] Yu-En L., Steven H., and Pietro L.: "Keyword Searching in Hypercubic Manifolds," in the Fifth IEEE International Conference on Peer-to-Peer Computing (P2P'05), 2005, pp. 150-151.

[25] Zhou A., R. Zhang, W. Qian, Q. H. Vu, and T. Hu, "Adaptive Indexing for Content-Based Search in P2P Systems," *Data and Knowledge Engineering,* vol. 67, no. 3, December 2008, pp. 381-398.

[26] Zhua Y. and Hub Y.: "Efficient Semantic Search on DHT Overlays," Parallel Distributed Computing vol. 67, no. 5 pp. 604-616, May 2007.

Yaser A. Al-Lahham received the BS degree from University of Jordan in 1985, the MS degree from Arab Academy (Jordan) in 2004, and the PhD in Computer science from Bradford University (UK) in 2009. He is working as an assistant professor in the Department of Computer Science at Zarqa University in Jordan. His research interest includes P2P information retrieval systems, text clustering, and data mining.

Mohammad Hassan received the BS from Yarmouk University in Jordan in 1987, the MS degree from Univ. Of Jordan in 1996, and the PhD in Computer Information Systems from Bradford University (UK) in 2003. He is working as an assistant professor in the department of computer science at Zarqa University in Jordan. His research interest includes information retrieval systems and database systems.