Information Processing and Management 53 (2017) 388–404

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Intelligent algorithms for improving communication patterns in thematic P2P search

Ana Lucía Nicolini, Carlos Martín Lorenzetti*, Ana Gabriela Maguitman, Carlos Iván Chesñevar

Institute for Research in Computer Science and Engineering (ICIC), UNS - CONICET, San Andrés 800, Bahia Blanca, Argentina

ARTICLE INFO

Article history:
Received 2 July 2014
Revised 29 November 2016
Accepted 2 December 2016

Keywords:
P2P systems
Thematic search
Semantic communities

ABSTRACT

The Internet is a cooperative and decentralized network built out of millions of participants that store and share large amounts of information with other users. Peer-to-peer systems go hand-in-hand with this huge decentralized network, where each individual node can serve content as well as request it. In this scenario, the analysis, development and testing of distributed search algorithms is a key research avenue. In particular, thematic search algorithms should lead to and benefit from the emergence of semantic communities that are the result of the interaction among participants. As a result, intelligent algorithms for neighbor selection should give rise to a logical network topology reflecting efficient communication patterns. This paper presents a series of algorithms which are specifically aimed at reducing the propagation of queries in the network, by applying a novel approach for learning peers' interests. These algorithms were constructed in an incremental way so that each new algorithm presents some improvements over the previous ones. Several simulations were completed to analyze the connectivity and query propagation patterns of the emergent logical networks. The results indicate that the algorithms with better behavior are those that induce greater collaboration among peers.

1. Introduction

The current information age has facilitated the generation, publication and access to geographically spread resources and heterogeneous content. A peer-to-peer (P2P) system uses the computational power and the bandwidth of the participants instead of relying on a small number of servers (Balakrishnan, Kaashoek, Karger, Morris, & Stoica, 2003; Merabti, Liu, Yu, & Kundur, 2010). In recent years, many techniques for sharing and searching information on P2P networks have been proposed (Tigelaar, Hiemstra, & Trieschnigg, 2012), leading to new opportunities to preserve, foster and exploit the diversity of social communities on the Internet. In this scenario, we can identify several research challenges for developing mechanisms to manage and access distributed resources in a variety of formats. While research on P2P systems has facilitated the implementation of robust distributed architectures, there are still several limitations faced by current search mechanisms. In particular, these mechanisms are unable to reflect a thematic context in a search request and to effectively take advantage of the peers' interests to improve the network communication patterns.

* Corresponding author.
E-mail addresses: aln@cs.uns.edu.ar (A.L. Nicolini), cml@cs.uns.edu.ar (C.M. Lorenzetti), agm@cs.uns.edu.ar (A.G. Maguitman), cic@cs.uns.edu.ar (C.I. Chesñevar).

http://dx.doi.org/10.1016/j.ipm.2016.12.001
0306-4573/

P2P network topologies are typically classified as *structured* (Korzun & Gurtov, 2013) or *unstructured* (Jin & Chan, 2010), while some networks combine some properties of both (Rosenfeld, Goldman, Kaminka, & Kraus, 2009). A structured network is organized into a specific topology with a protocol aimed at ensuring a reasonable search performance. On the contrary, an unstructured network does not follow a specific pattern for the organization of its nodes and has a random or irregular topology. Because of this lack of structure, these networks have a relatively low search efficiency compared with the structured ones. As a consequence, the study of algorithms that improve search efficiency in unstructured P2P systems has been a relevant research area for some years (e.g. Tang, Xu, & Dwarkadas, 2003; Radicchi, Castellano, Cecconi, Loreto, & Parisi, 2004; Du, Wang, & Wu, 2008; Zhu, Wu, & Wang, 2009; Wang, 2011).

Current search services are rigid as they do not offer mechanisms to facilitate users access to information about possibly relevant topics with which they might not be familiar. Another limitation of the current search model is the lack of context sensitivity. Although some websites offer tools for personalized search, most of such tools lack of proper mechanisms to facilitate contextualization and collaboration. These factors are crucial in thematic and distributed search environments. In a distributed search model, participants collaborate by sharing the information stored on their computers. Differently from the client-server model, P2P systems have the capability of increasing their performance as the number of users increases. To take advantage of this potential it is necessary to develop adaptive and collaborative mechanisms to exploit the semantics of the users' communities, the resources that they store and their search behavior.

The main goal of this work is to provide unstructured P2P systems with mechanisms for context-based search and to propose algorithms that incrementally learn effective communication patterns in pure P2P networks.[1] In these networks, each participant operates in an autonomous manner, without relying on a specific server for solving communication and coordination tasks. In other words, peers can make decisions in an autonomous way in order to route queries to potentially relevant nodes based on the acquired knowledge about other peers' interests. An important aspect of our proposal is that there is no initial knowledge about where the information is stored. Our work focuses on studying how this knowledge can be acquired. In particular, eight algorithms are proposed that enable peers to learn how to route queries to relevant nodes. This collective learning process leads to the emergence of semantic communities, resulting in a logical network with improved communication patterns.

We compute the "clustering coefficient" of the emergent logical network to quantify the learning rate of the peers. Our simulations indicate that the most intelligent algorithms give rise to logical networks with higher clustering coefficients, clearly indicating the emergence of semantic communities. The existence of these communities reduces the scope of query propagation through the network, resulting in more effective communication patterns.

This article is organized as follows. Next section presents some background concepts used in the rest of the paper. Section 3 discusses the state of the art in the area, reviewing recent related work. The main contribution of this work is introduced in Section 4, where eight algorithms for thematic P2P search are described. The results of evaluating these algorithms through extensive simulations are analyzed in Section 5. Finally, Section 6 summarizes the conclusions and discusses some future research directions.

2. Background

A possible solution to improve communication overhead and scalability in large-scale unstructured P2P systems is to forward queries to a group of peers that are known to be potentially useful to answer the query. The selection of potentially useful peers is typically based on the peer's past activity or their semantic similarity to the original query (Barbosa, Costa, Almeida, & Almeida, 2004; Voulgaris, Kermarrec, Massouli, & van Oteen, 2004). A *semantic community*, in the context of a P2P network, is a group of nodes sharing common or related information. A concept associated with semantic community is *semantic locality* (Yan & Zhan, 2004). Traditionally, the notion of semantic locality has been used to refer to the ability to store information about peers offering semantically close services. Semantic locality has also been defined as "a logical semantic categorization of a group of peers sharing common data" (Sharan, 2006). The performance of routing algorithms in P2P networks can be improved by applying mechanisms that exploit the notion of semantic locality. These routing algorithms help peers to predict which other peers have knowledge useful to respond a query in a few steps and without overloading the system bandwidth (Tirado, Higuero, Isaila, Carretero, & Iamnitchi, 2010). This makes possible that a query be efficiently propagated in the network through relevant nodes, and suggests that collaborative and distributed search can benefit from the user context and the existence of semantic communities.

In order to evaluate the connectivity and communication patterns of the emergent logical networks we study several of their structural features, such as small-world topology (Steyvers & Tenenbaum, 2005; Watts & Strogatz, 1998), degree distribution, and k-core decomposition (Seidman, 1983). We also study the emergence of semantic communities by applying a methodology similar to the one used in Akavipat, Wu, Menczer, and Maguitman (2006).

[1] This article extends preliminary work on thematic search in a P2P context presented by the authors in a conference paper in 2013 (see Nicolini, Lorenzetti, Maguitman, & Chesñevar, 2013).

2.1. Clustering coefficient

The local clustering coefficient quantifies the nodes' tendency to cluster together in a single node's immediate network (i.e., the node and its neighbors) (Watts & Strogatz, 1998). In order to represent a network we will consider an undirected graph $G = (V, E)$, in which V is the set of nodes and E is the set of edges. We will write e_{ij} to denote an edge from node v_i to node v_j. For a node v_i, its neighborhood N_i is defined as the set of nodes v_j immediately connected to v_i, that is,

$$N_i =_{def} \{v_j \mid e_{ij} \in E \vee e_{ji} \in E\}.$$

The local clustering coefficient is based on the network's density or local density (Burt, 1992; Everett & Borgatti, 2005). For each node v_i, this is measured as the fraction of the number of edges connecting v_i's neighbors over the total number of possible edges between v_i's neighbors. Let k_i be the number of neighbors of a node v_i, that is, $|N_i|$. If a node v_i has k_i neighbors then there are at most $k_i(k_i - 1)/2$ edges among the nodes within the neighborhood (if the neighborhood is fully connected). Therefore, the local clustering coefficient for a node v_i in an undirected graph can be computed as follows:

$$C_i = \frac{2|e_{jk} \in E : v_j, v_k \in N_i|}{k_i(k_i - 1)}.$$

In order to calculate the clustering coefficient for the whole network, the individual local clustering coefficients are averaged across all nodes (Watts & Strogatz, 1998). Let n be the number of vertices in the network, that is $|E|$. The network's average clustering coefficient is computed as:

$$C_{average} = \frac{1}{n}\sum_{i=1}^{n} C_i.$$

2.2. Small-world topology

A graph or network is considered *small-world* if its links are globally sparse (the network is far from being fully connected), its $C_{average}$ is higher than the average clustering coefficient associated with a random graph and the length of the path connecting two nodes is orders of magnitude smaller than the network size (Watts & Strogatz, 1998).

This notion is typically representative of the global knowledge of the network and was selected in this work to compare the ability of the proposed algorithms to spread information associated with the nodes across the network. When the amount of information about the nodes in the network is insufficient, $C_{average}$ is small. However, as this information grows, the value of $C_{average}$ will grow as well. In addition, only a few hops are needed to send a message from the message's originator to its recipient, indicating that the length of the path connecting the originator and recipient nodes is small.

2.3. P2P Architectures

P2P systems are decentralized, large-scale computer networks, where peers operate as clients and servers at the same time. Peers can join and leave the system at any time. The power of P2P systems lies in their capability to provide services with practically unlimited scalability based on the principle of resource sharing (Tigelaar et al., 2012).

Search performance in a P2P system is highly dependent on the architecture of the underlying network, which can be characterized as *centralized, decentralized–structured*, or *decentralized–unstructured*:

Centralized: Napster (Carlsson & Gustavsson, 2001) and other similar systems have a directory hosted at central locations. Nodes in the P2P networks send their queries to the central directory server to find which other nodes hold the desired files. This approach does not scale well and is not reliable, as it has critical points of failure.

Decentralized – Structured: These systems have no central directory server but the connections between the nodes are controlled and the files are not placed at random but at specific locations that will make queries easy to satisfy. Freenet [2] is an example of such systems.

Decentralized – Unstructured: These are systems in which there is neither a centralized directory nor any control over the network topology or file placement. Gnutella (Ripeanu, 2001) is possibly the best-known example of this type of architecture. The network is constituted by nodes that join the network following some loose rules. The resultant topology has certain properties, but the placement of files on individual nodes is not based on any knowledge of the topology. To find a file a node queries its neighbors, typically using flooding as the communication method, where the query is propagated to all the neighbors within a radius. This kind of search does not scale well, generating large loads on the network participants and congestion on the whole network unless specialized mechanisms for query routing are implemented.

In this paper, we focus on decentralized unstructured P2P network architectures. Therefore, our main goal is to develop intelligent mechanisms for improving communication patterns.

[2] http://freenet.sourceforge.net.

3. Related work

Preliminary work on intelligent algorithms for P2P search (Nicolini et al., 2013) provided the basis for the work presented here. In contrast with this article, that preliminary work did not include all the algorithms presented here, nor an extensive empirical analysis and comparison of their associated results in the context of the state-of-the-art in P2P search.

Much research has focused on how to structure the network for routing queries in P2P systems. A proposed solution is to create a two-layer architecture: the upper one is the semantic layer that controls the super peers, whereas the lower layer is in charge of getting the relevant files (Eftychiou, Vrusias, & Antonopoulos, 2012). Other approaches that use super peers were proposed in Ismail, Quafafou, Nachouki, and Hajjar (2010), where decision trees are used to improve search performance for information retrieval. A novel approach of a hybrid forwarding framework is presented in Yu, Gerla, and Sanadidi (2015). The flexibility of this framework allows to perform proactive or reactive content discovery based on content characteristics. The framework utilizes the content identifiability and name prefixes to classify time-sensitive data with the purpose of applying the most suitable strategy to each category. For proactive content dissemination they propose a Hierarchical Bloom-Filter based Routing algorithm (Broder & Mitzenmacher, 2004).

There is an increasing interest in algorithms that dynamically modify the topology of the logical network, guided by mechanisms that allow participants to learn about the topics associated with the resources offered by other participants as well as their information needs (Akavipat et al., 2006; Du et al., 2008; Radicchi et al., 2004; Tang et al., 2003; Wang, 2011; Yeferny & Arour, 2010; Zhu et al., 2009). These systems offer a way to relax the restrictions imposed by a centralized, planned and sequential control, resulting in decentralized and concurrent systems with collective behaviors (Watts & Strogatz, 1998). In Meng, Ding, Peng, and Yue (2013) an algorithm is proposed that partitions the P2P network into different clusters based on their interests, giving rise to a hierarchical structure. A metric is defined to calculate the similarity between the node and cluster vectors to determine if a given node can join a cluster or not.

Algorithms that route queries based on peers' interests have attracted increasing research interest. For example, the algorithm presented in Sripanidkulchai, Maggs, and Zhang (2003) adds new connections to the Gnutella overlay network, linking nodes which share similar interests. Experimental results show that this algorithm can avoid unnecessary flooding, improving significantly the system performance. CAC-SPIRP (Guo, Jiang, Xiao, & Zhang, 2004) is a fast and low-cost P2P searching algorithm where peers holding large amounts of content self-identify and self-organize themselves into clusters; those queries most likely to be satisfied are the first to be routed. In this work, it is shown that by exploiting search interest locality it is possible to achieve significant performance improvements. A method for modeling users' interest in P2P document sharing systems based on *k*-medoids clustering is presented in Qin, Yang, and Liu (2014). In the proposed approach an overlay network is created based on the *k*-medoids clustering algorithm, which is combined with the users' historical queries to improve the initial user interest model.

Some algorithms apply local environment strategies for exploring the network and learning about other peers to improve search performance (Gómez Santillán, Cruz Reyes, Meza Conde, Schaeffer, & Castilla Valdez, 2010). These self-adaptive algorithms were specially developed to route text queries regardless of their topic. An algorithm based on flooding that forwards the query to the most effective nodes in an unstructured P2P network, without considering semantic communities, is presented in Sujeeth, Kumar Kond Reddy, and Kumar Reddy D. (2013). In Kalnis, Ng, Ooi, and Tan (2006) some queries are paused with the purpose of reducing flooding. In the meantime, a set of peers examine the messages that pass through them and take advantage of the results of similar queries originated in other parts of the network to provide a response to those messages.

There is a wide variety of search engines based on P2P technologies. For example, the model proposed by the YouSearch project (Bawa, Bayardo Jr., Rajagopalan, & Shekita, 2003) takes a centralized Napster-like design for query routing, where several participants can simultaneously find and index different portions of the web. Other systems such as NeuroGrid (Joseph, 2002) attempt to send queries to potentially useful nodes. Most of these systems use automatic learning techniques to adjust the metadata that describes the content of the nodes. Currently, there exist some software tools for decentralized search such as Faroo[3] and Yacy.[4]

An efficient algorithm named *state-based search* (SBS), which routes queries according to nodes' state information, is presented in Wu and Wu (2013). Experimental results show that the SBS algorithm is able to effectively improve performance by reducing search response time and achieving better load balance. Each SBS node maintains a state-based shortcut list with other nodes' state information. In SBS, efficient search can be achieved by discovering the nodes that share the desirable resources according to a local fuzzy logic-based routing algorithm.

Moderating query routing overhead through congestion control has also proven to be an alternative to improve query processing (Shen et al., 2016). This method attempts to moderate query routing overhead through congestion control. This method assures alternative routing paths to balance the query load among the peers under higher network churns. The method relies on the *Collaborative Q-Learning* algorithm, which learns network parameters such as processing capacity, number of connections, and number of resources in the peers, along with their congestion degree.

[3] http://www.faroo.com.
[4] http://www.yacy.net.

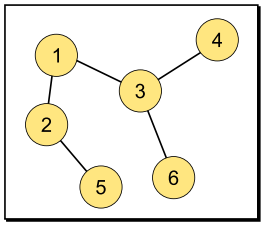

Fig. 1. Example network.

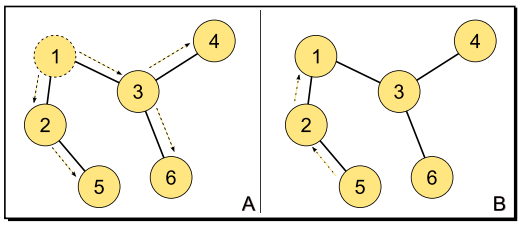

Fig. 2. Query and reply messages for Algorithm 1.0.

The analysis presented in Li et al. (2003) suggests that information retrieval in P2P networks is not feasible at the Web scale due to the overhead introduced in peers' communication. However, subsequent studies indicate that this overhead could be significantly reduced by developing systems that minimize the number of sent messages per query while keeping a high degree of precision and recall (Zeinalipour-Yazti, Kalogeraki, & Gunopulos, 2004). In these systems it is crucial to predict which nodes are appropriate to receive a query. Several research results have confirmed that grouping the participants by semantic similarity leads to an increment in the system efficiency (Akavipat et al., 2006; Barbosa et al., 2004; Crespo & Garcia-Molina, 2002; Doulamis, Karamolegkos, Doulamis, & Nikolakopoulos, 2009; Zeinalipour-Yazti, Kalogeraki, & Gunopulos, 2005). Following this strand of research, our work presents an objective study of the performance of different search strategies based on the concept of "semantic locality" in a distributed system.

4. Algorithms

This section describes a series of algorithms that offer a solution to the problem of searching in a decentralized, unstructured P2P system. Except for Algorithm 1.0, all these algorithms share the same basic structure: each node has an NT table (Nodes' Topic table) in which the learned knowledge is stored. An entry in this table consists of a topic and a set of nodes that are interested in this topic. A node is considered to be interested in a topic when it has a directory of articles related to this topic.

The main differences between the algorithms are the update policy adopted by each of them for updating the NT table and the mechanisms for selecting a neighbor for query propagation. We will use simplified hypothetical networks as the one presented in Fig. 1 to describe the behavior of the proposed algorithms.

In the rest of this article, dashed nodes will be the ones that generate a query, while dotted nodes will be the ones that introduce new knowledge. Dashed arrows will represent the path of a query, and dotted arrows will indicate the path followed by a response. In addition, dashed-dotted arrows will be used to represent the path of the update messages. We adopt a basic version numbering scheme to indicate that an algorithm incorporates improvements over a previous version. For instance, Algorithm 2.1 is an optimization of Algorithm 2.0. It should be noted that during the development of these algorithms we have assumed that a node always generates queries related to the topics in which it is interested.

4.1. Algorithm 1.0

This algorithm does not have any learning capability and therefore does not require the use of an NT table for each peer. The queries are routed in a brute-force search manner, as in Gnutella (Ripeanu, 2001). Each time a node generates a query it sends the message to every adjacent node. If a node that receives a query message can reply, it sends a reply message. Otherwise, it forwards the query to its adjacent nodes until exhausting the initially defined number of query hops. The maximum number of hops allowed is usually referred to as the *time to live* (TTL) parameter. Fig. 2 shows the behavior of this algorithm.

4.2. Algorithm 2.0

In this algorithm, when a node generates a query message, it first consults its NT table to find out which nodes are interested in that topic. Then one of these candidate nodes is selected in a random way and the query is sent to this node.

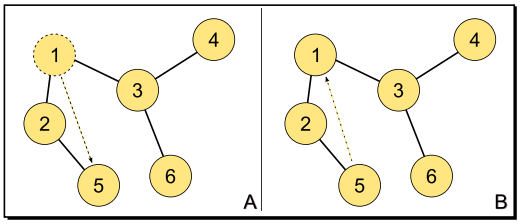

Fig. 3. Query and reply messages for Algorithm 2.0. Node 1 knows node 5.

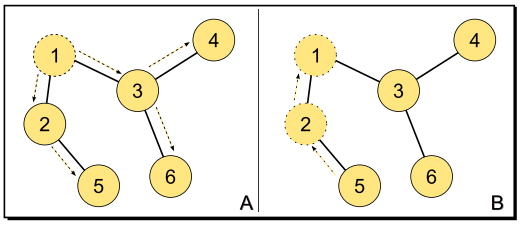

Fig. 4. Query and reply messages for Algorithm 2.0. Node 1 does not know node 5.

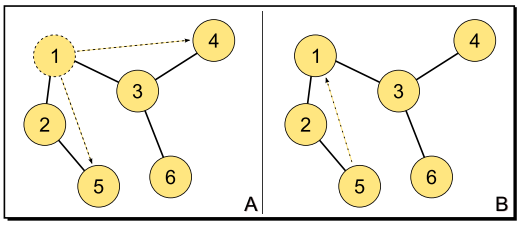

Fig. 5. Query and reply messages for Algorithm 2.1.

In the case that the query-issuing node does not have an entry for this topic in its NT table, it sends the query message to all of its adjacent nodes, in the same way as Algorithm 1.0. The learning phase occurs with the reply message. When a node can reply a query, it sends a reply message that follows the same path as the query. Each intermediate node in this path updates its NT table with the topic of the query that is being answered and the node that answered it.

For the sake of example, Fig. 3 shows the messages involved in a scenario where node 1 issues a query that matches a file stored on node 5, while node 1 knows node 5 through its NT table. On the other hand, Fig. 4 illustrates the situation in which node 1 does not know node 5.

4.3. Algorithm 2.1

The difference between this algorithm and Algorithm 2.0 is that a query is sent to *all* the candidate nodes that are interested in the query topic according to the knowledge stored in the NT table of the query-issuing node. Learning is performed in the same manner as in the previous algorithm. Fig. 5 illustrates a situation in which node 1 issues a query that can be replied by node 5. Node 1 knows through its NT table that nodes 4 and 5 are interested in the query topic, so instead of sending the query message to only one of these nodes, it sends the query to both nodes simultaneously.

4.4. Algorithm 3.0

The behavior of the query and reply messages is the same as in Algorithm 2.1. However, this algorithm incorporates the use of *update messages*. When a node learns new information it updates its NT table and then it sends an update message with the information learned—in the format (topic, node)—to all of its adjacent nodes. This new feature improves the learning rate of the network. In the previous algorithms, if a node is never found in the path of a reply message, it will not have the opportunity to learn new information. On the other hand, in this new version a node not only learns if it is in the path of a reply message but it also learns if it is adjacent to a node in this path.

Fig. 6 illustrates the case in which node 1 does not have node 5 in its NT table. Dotted nodes are those that learn which are the interests of node 5. We add a new edge to the hypothetical network to illustrate this situation. It should be noted that node 2 learns the interests of node 5 and therefore this knowledge is propagated with an update message only to the adjacent node 3. This is due to the fact that node 5 is the responding node and node 1 is in the response path.

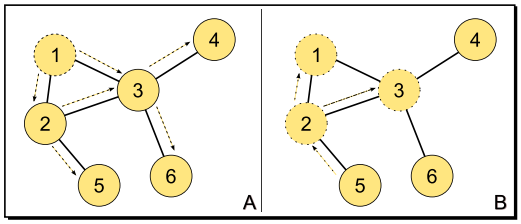

Fig. 6. Query, reply and update messages for Algorithm 3.0.

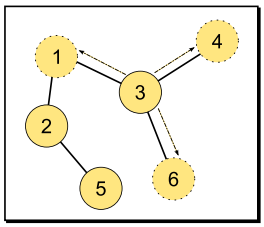

Fig. 7. Update messages for Algorithm 3.1.

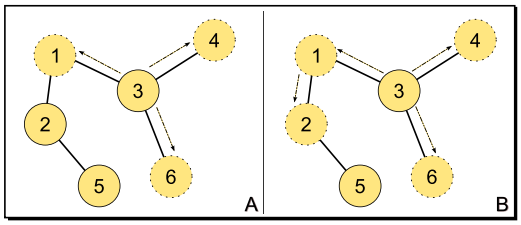

Fig. 8. Update messages for Algorithm 3.2.

4.5. Algorithm 3.1

This version differs from Algorithm 2.1 only in the initialization of a node that becomes part of the network. When a node joins the network, it sends an update message to all its adjacent nodes. So its immediate neighbors can know which is the new node, as well as its associated interests. Fig. 7 shows the behavior of a node (node 3 in this case) joining the network.

4.6. Algorithm 3.2

Algorithm 3.2 has a small difference with respect to Algorithm 3.0: when a node sends an update message to its neighbors, each of them forwards the original message to its own neighbors. In all the cases, it avoids forwarding the message to the node from which the message came. Fig. 8A illustrates the first hop: node 3 sends an update message to its neighbors. Fig. 8B shows the second hop: nodes 1, 4 and 6 forward the message to its own adjacent nodes. Dashed-dot-dot edges in Fig. 8B represent the first hop messages while the dashed-dot edge represents the second hop.

4.7. Algorithm 3.3

This algorithm incorporates the following changes with respect to version 3.0: when a node receives a reply message with a topic that is interesting for this node, it sends an update message to all of its adjacent nodes and to all the nodes which are “known” (through its NT table) to be interested in the topic of the reply message. This algorithm allows knowledge to propagate inside the community. Fig. 9 shows an example in which node 3 is interested in a particular topic and receives a reply message associated with this topic. Node 3 knows that nodes 2 and 5 are in its semantic community (interested in the same topic) and therefore when node 3 receives a reply message, it sends an update message to its neighbor nodes and to its semantic community (nodes 2 and 5).

4.8. Algorithm 4.3.3

This algorithm maintains the behavior of the previous version except that nodes send update messages in additional situations. If a query message arrives by broadcast and the node is interested in the topic of the query but cannot reply, it

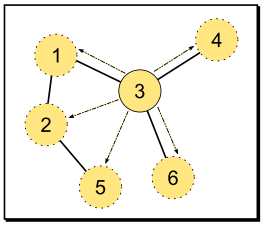

Fig. 9. Update messages for Algorithm 3.3.

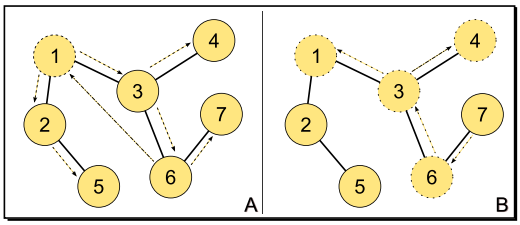

Fig. 10. Query, reply and update messages for Algorithm 4.3.3.

Table 1
A comparative summary of the algorithms' main features.

	Algorithm								
Features	1.0	2.0	2.1	3.0	3.1	3.2	3.3	4.3.3	5.0
Query to adjacent nodes (always)	✓	–	–	–	–	–	–	–	–
Query to adjacent nodes (only without knowledge)	–	✓	✓	✓	✓	✓	✓	✓	✓
Query to the semantic community	–	✓[a]	✓	✓	✓	✓	✓	✓	✓
Update to adjacent nodes	–	–	–	✓	–	✓[b]	✓	✓	–
Update to the semantic community	–	–	–	–	–	–	✓	✓	✓
Update when joining the network	–	–	–	–	✓	–	–	–	–
Update to query-issuing node	–	–	–	–	–	–	–	✓	✓

[a] Only to one randomly selected member.
[b] Two level propagation.

will send an update message to the node that originated the query. This behavior avoids excluding nodes that do not have many resources. Fig. 10 presents an example illustrating this situation. Suppose that node 1 generates a query that can be replied by node 7, but node 1 does not know about node 7. In Fig. 10A the query message arrives to node 6 (which has at least one interest in common with node 1), and therefore this node sends an update message to node 1. In Fig. 10B node 7 answers the query and the behavior is the same as in Algorithm 3.3.

4.9. Algorithm 5.0

The difference between this algorithm and Algorithm 4.3.3 is that this version skips update messages to adjacent nodes, so in this way update messages are sent only between nodes of the same semantic community. This algorithm keeps the learning rate high but decreases the global number of update messages, reducing network traffic.

4.10. Main features of the algorithms

Table 1 presents a comparative summary of the eight algorithms described in this section based on their main features.

The proposed algorithms implement the essential characteristics of a number of existing P2P systems, abstracting away minor details and parametrization issues and focusing mainly on their most salient features. Algorithm 1.0 implements the flooding algorithm also implemented by the Gnutella System (Ripeanu, 2001), where queries are forwarded in a breadth-first search until the TTL is reached. In the rest of the proposed algorithms, the use of routing tables and the exploitation of semantic locality are the most relevant features. This relates with existing search methods based on routing indexes (Crespo & Garcia-Molina, 2002), which use information about neighbors (stored in routing tables) to guide the search process. For instance, the Learning Peer Selection approach (Arour & Yeferny, 2014) implements a method with the ability to discover users' topical preferences by analyzing their download history. This gives rise to nodes profiles that are stored in tables. If the information stored in the tables is not sufficient to guide the search process, then the flooding approach is applied.

The concept of knowledge sharing is an emerging topic generally based on cooperation among peers. There is a fundamental trade-off between sharing knowledge to improve the network's global knowledge and the cost of sending the

Table 2
Number of articles associated with each topic.

Topic	Number of articles
T1: Agricultural Science	2403
T2: Biological Science	4250
T3: Health Science	23,754
T4: Earth Science	2775
T5: Geosciences	197
T6: Social Sciences	1705
T7: Applied Social Sciences	4421
T8: Humanities	4652
T9: Engineering	1289
T10: Linguistics, Literature and Arts	164
T11: Mathematics	9
T12: Chemistry	123
Total number of articles	45,742

messages with this information. Some of the algorithms proposed in this work (3.0–5.0) implement different variations of this concept through the use of update messages.

The establishment of connections in the logical network implemented by the proposed algorithms is similar to approaches that establish friendship relations, such as the one implemented in Self Learning Query Routing (Chen, Gong, & Huang, 2005). In this routing algorithm, the interests of the peers are learned based on their search result history, which allows to automatically construct friendship relations based on the similarity of users' interests.

5. Evaluation

5.1. Simulation setting

Several simulations were carried out using realistic data to compare the performance of the proposed algorithms. Evaluations were run with physical networks containing 1000 and 5000 nodes. These networks were generated using the Erdős–Rényi model (Erdős & Rényi, 1959). The parameter p in the model was set in such a way that each node had an average degree of approximately 10 (e.g., for $|N| = 5000$, $p = 0.002$). To ensure the correct operation of the algorithms we verified that the generated physical networks were connected. It is important to distinguish the notion of *physical network* from the one corresponding to *logical network*. Physical networks remain static through all the runs, whereas logical networks change on each run since they are the result of the evolution of the networks' global knowledge. There are no centralized directories or indexes. Each node in these networks is a content provider and an individual directory and is associated with one or more topics of interest.

The first challenge to address for carrying out a full comparative analysis was to select an appropriate simulation framework. After a careful evaluation of different alternatives, we opted for the OmNet++ framework (Pongor, 1993; Varga & Hornig, 2008) due to its flexibility and ease of use. The visualizations of the logical networks resulting from our simulations were created with JUNG (Java Universal Network/Graph Framework) [5] and the k-core decomposition of the logical networks and the corresponding visualizations were performed with the LaNet-vi tool [6] (Large Networks Visualization Tool). As input for our simulations, we used 45,742 scientific articles obtained from the Scientific Electronic Library Online SciELO. [7] The articles were previously classified into 12 different topics as shown in Table 2. These topics were distributed among nodes such that each node contains one or more related topics. Subsets of these articles were assigned randomly to different nodes, with the only constraint that each node should contain articles related to the node's interests.

With the purpose of comparing the proposed algorithms for query routing, each algorithm was initially executed ten times with a physical network of 1000 nodes. During each run, queries were issued by randomly selected nodes. The number of queries for a run was 150 and the maximum number of hops before a query was discarded was set to 50. Every new run of the same algorithm was able to take advantage of the knowledge learned during the previous runs. The comparison criteria used are the following:

- The average clustering coefficient of the logical network.
- The number of queries that have been satisfied.
- The number of messages sent by each node, taking into account update messages to analyze whether such messages were congesting the network.
- The number of hops needed to find an answer.

[5] http://jung.sourceforge.net.
[6] http://lanet-vi.fi.uba.ar/.
[7] http://www.scielo.org.

Table 3
A comparison of the number of answered queries ($|N| = 1000$).

Answered queries									
	Algorithm								
Run	1.0	2.0	2.1	3.0	3.1	3.2	3.3	4.3.3	5.0
1	92	142	125	115	100	106	110	135	128
3	80	81	98	99	94	98	97	97	94
5	63	87	92	90	93	92	93	94	89
7	120	84	97	88	102	100	86	83	95
10	75	80	94	92	104	97	92	91	98

Table 4
A comparison of the number of hops taken to find an answer ($|N| = 1000$).

Hops									
	Algorithm								
Run	1.0	2.0	2.1	3.0	3.1	3.2	3.3	4.3.3	5.0
1	30	37	49	48	46	46	45	25	29
3	29	48	39	27	6	32	8	2	2
5	29	46	20	12	5	8	26	2	2
7	28	49	18	7	5	17	3	2	2
10	30	50	21	7	5	10	9	2	2

Table 5
A comparison of the clustering coefficients ($|N| = 1000$).

Clustering Coefficient									
	Algorithm								
Run	1.0	2.0	2.1	3.0	3.1	3.2	3.3	4.3.3	5.0
1	0	0.0984	0.1016	0.1955	0.2713	0.2921	0.3638	0.696	0.6862
3	0	0.1510	0.1548	0.2450	0.2754	0.3286	0.4806	0.7062	0.7013
5	0	0.1787	0.1676	0.2443	0.2788	0.3356	0.5146	0.7088	0.7052
7	0	0.1914	0.1826	0.2551	0.2805	0.3380	0.5228	0.7098	0.7093
10	0	0.2130	0.1906	0.2636	0.2827	0.3434	0.5274	0.7135	0.7099

All the simulations were executed on a server with these characteristics:

- 32 processors (4 × 8 cores) Opteron.
- 128GB RAM
- Debian GNU/Linux 6.0 64 bits.
- kernel 3.8.3.
- Oracle JRE 1.7.0 21.

5.2. Communication patterns and small-word structure

Table 3 shows the performance of the evaluated algorithms based on the number of answered queries. The reported results correspond to the first, third, fifth, seventh and tenth runs. The resulting data indicates that the number of answered queries is higher in the first run of the algorithms, since in these cases the queries are blindly propagated through the whole network. As new runs are completed, the number of answered queries decreases.

Table 4 and Fig. 11 show the average number of hops (rounded up to the nearest integer) needed to find an answer for a query. In general, we can see that this number decreases as the overall knowledge of the network increases. Although we can notice that this behavior does not hold for all the analyzed algorithms, the most evolved algorithms (versions 4.3.3 and 5.0) exhibit a significant reduction in the number of hops from the first to the following runs.

The clustering coefficient was computed to analyze the evolution of the global knowledge of the network. A high clustering coefficient usually indicates the presence of semantic communities in the network. Table 5 and Fig. 12 show that the average clustering coefficient considerably increases throughout the runs. This is in agreement with the fact that during the latest runs the global knowledge of the network is higher and nodes can send queries directly to potentially useful nodes.

Concerning the number of messages sent, we can see that this number decreases as the number of runs increases. This is because the queries reach a useful node in fewer hops and as a consequence fewer messages are propagated. This result is reflected in Table 6 and Fig. 13. It should be clarified that update messages are considered part of the set of sent messages.

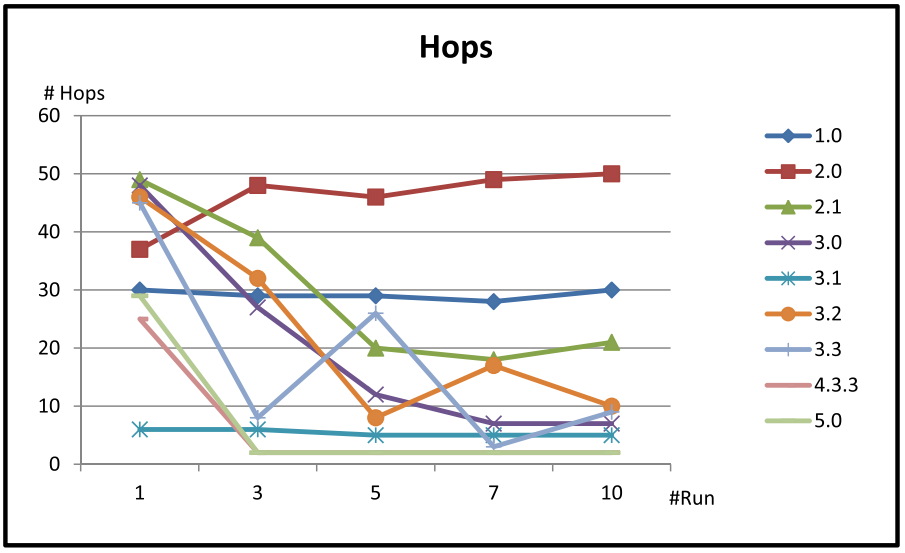

Fig. 11. A comparison of the number of hops taken to find an answer ($|N| = 1000$).

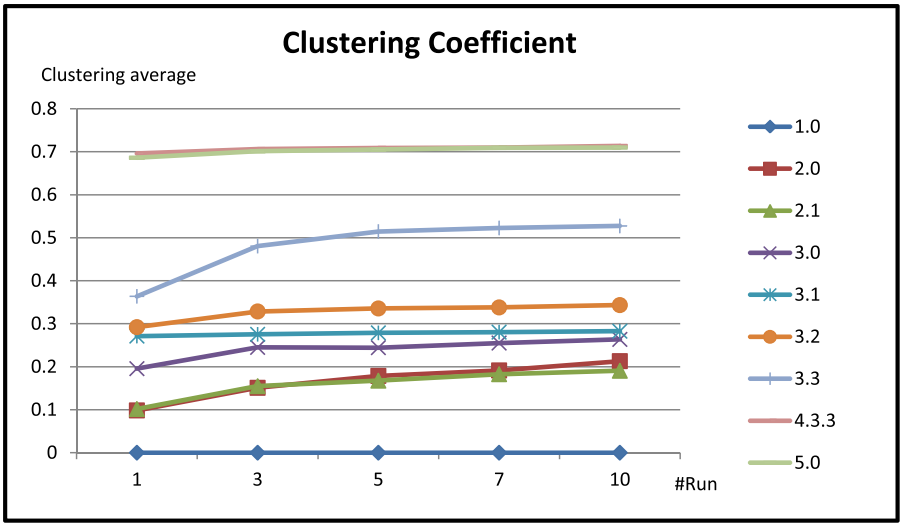

Fig. 12. A comparison of the clustering coefficients ($|N| = 1000$).

The results reported in Table 7 and Fig. 14 indicate that Algorithm 5.0 sends fewer update messages than Algorithm 4.3.3, while the associated global knowledge is not modified. Algorithms 2.0 and 2.1 do not implement update messages and the rest of the algorithms show a lower amount of update messages, as they do not share their knowledge with their communities.

5.3. Scalability evaluation

Scalability is an important issue in P2P networks. Well-designed P2P networks are typically scalable due to the fact that each peer potentially acts as a server. However, testing the scalability of a P2P routing algorithm using a centralized simulation framework is challenging due to hardware limitations resulting from the use of a single server. It is important to highlight that large scale simulations for P2P information retrieval systems are particularly memory intensive as they involve the replication of a great number of search engines (one for each node) using a single server. The internal structures created by the framework in order to maintain the network physical and logical connectivity, in addition to the nodes knowledge, were not designed to support networks of more than a few thousands of nodes. This makes it difficult to evaluate scalability in a realistic way. In order to partially assess scalability, we have focused on evaluating whether Algorithm 4.3.3 (the most promising algorithm based on the reported evaluations) maintains its performance trends (in terms of number of hops, clustering coefficient, number of sent messages, and number of update messages) when the network's size grows.

Table 6
A comparison of the number of sent messages ($|N| = 1000$).

	Sent messages								
	Algorithm								
Run	1.0	2.0	2.1	3.0	3.1	3.2	3.3	4.3.3	5.0
1	996,415	108,562	110,423	109,723	101,479	108,955	439,165	1,964,130	1,898,622
3	998,618	77,250	77,100	75,186	93,264	54,257	421,828	1,462,712	1,447,162
5	998,630	64,512	64,659	64,817	95,536	71,238	373,371	1,367,388	1,409,499
7	998,609	55,360	57,023	55,128	98,048	80,699	267,511	1,345,714	1,391,244
10	995,254	41,250	46,263	42,543	83,674	90,500	114,423	1,239,345	1,186,936

Fig. 13. A comparison of the number of sent messages ($|N| = 1000$).

Table 7
A comparison of the number of update messages ($|N| = 1000$).

	Update messages					
	Algorithm					
Run	3.0	3.1	3.2	3.3	4.3.3	5.0
1	10,755	98,300	21,881	62,152	214,890	201,085
3	2215	2340	7620	60,448	141,792	126,007
5	2172	2340	4991	61,062	129,781	144,702
7	2612	2340	4929	45,690	128,532	119,548
10	3789	2340	4046	18,345	127,907	114,369

Table 8
Performance analysis of Algorithm 4.3.3 ($N = 5000$ nodes).

Run	Number of Hops	Clustering Coefficient	Answered queries	Sent messages	Update messages
1	10	0.0913	53	9,806,320	1,052,825
3	2	0.2410	234	7,257,560	709,777
5	1	0.2664	401	6,925,420	649,921
7	1	0.2755	538	6,825,532	648,989
10	1	0.2810	567	6,196,961	635,841

The larger scale simulation was carried out using a physical network of 5000 nodes where the number of queries issued at each run was set to 750 and the maximum number of hops allowed for each query was set to 15. The results of these simulations are shown in Table 8. These results indicate that the connectivity and communication patterns observed for the analyzed metrics in a network of 1000 nodes are maintained when the number of nodes is increased to 5000.

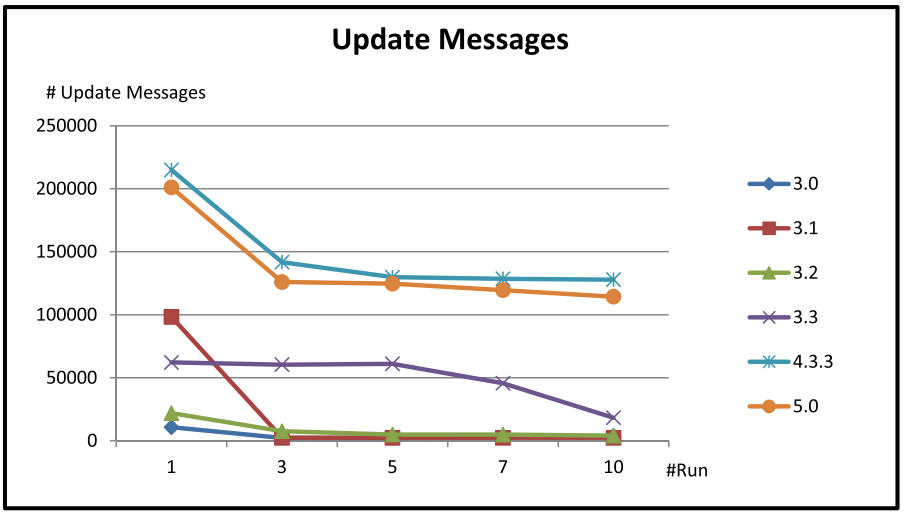

Fig. 14. A comparison of the number of update messages ($|N| = 1000$).

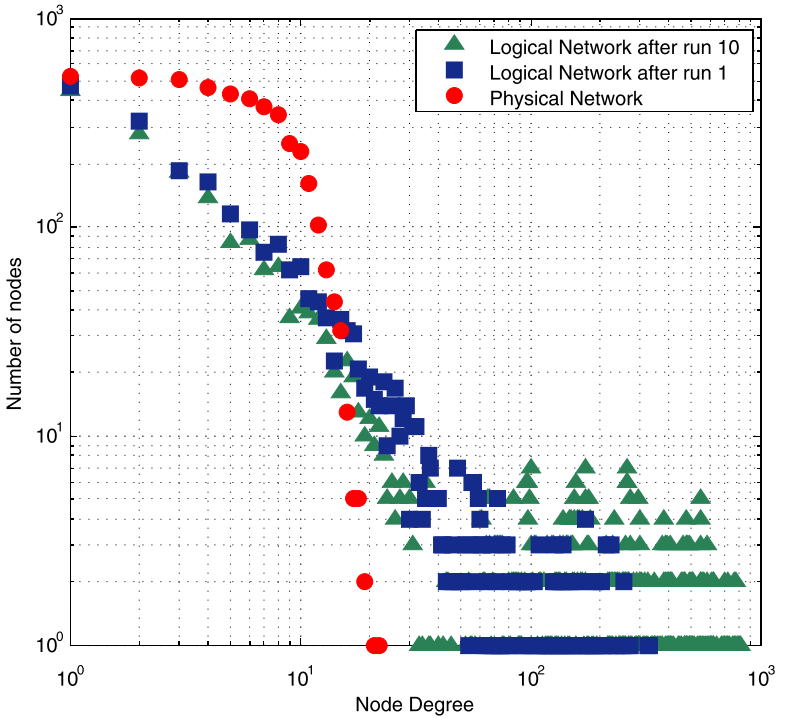

Fig. 15. Node degree distribution for the physical and the logical networks resulting from the first and tenth runs ($|N| = 5000$).

5.4. Degree distribution and K-core decomposition

The degree of a node is the number of nodes adjacent to it. The analysis of the degree distribution may reveal interesting features, such as the existence of hubs (i.e., nodes with high degree) or the scale-free nature of the network (i.e., networks that exhibit a power-law degree distribution.).

Fig. 15 shows the degree distributions (on a log-log scale) for the physical network and the logical networks resulting from the first and tenth run of Algorithm 4.3.3. The analysis of the degree distribution in the logical networks reveals the preferential attachment feature (Barabási & Albert, 1999), which means that nodes preferentially connect to the nodes with higher degrees. This gives rise to the formation of hubs. Although we can recognize a power-law distribution for the first run, the more evolved network tends to move away from this kind of power-law distribution as more and more nodes tend to reach high connectivity. This is most likely implied by the limited size of the network.

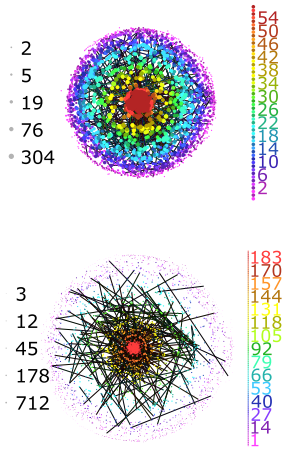

Fig. 16. Visualization of the k-core decomposition for the logical networks obtained in the first run (top) and tenth run (bottom) of Algorithm 4.3.3 (N = 5000).

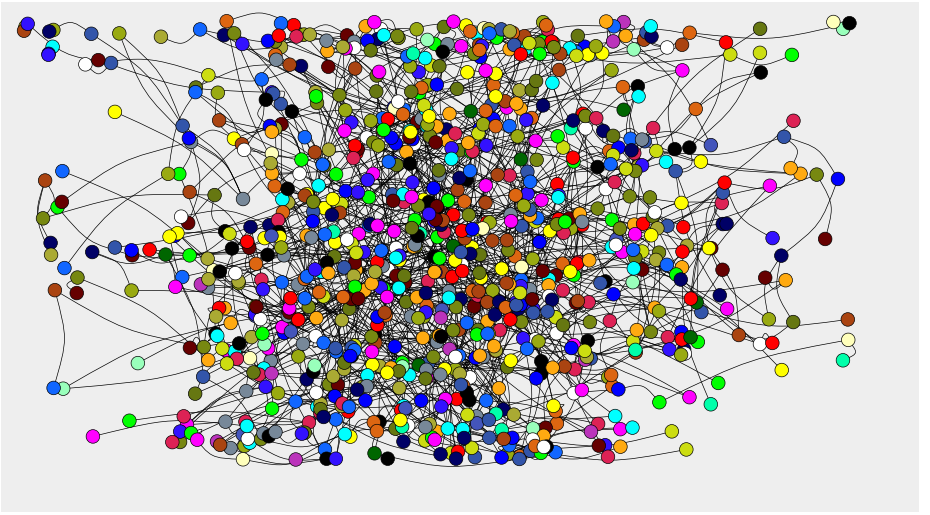

Fig. 17. Logical network obtained with Algorithm 1.0 (run = 10, $|N|$=1000).

To further investigate the connectivity patterns of the evolved logical networks, we analyzed their k-core decomposition (Seidman, 1983). H_k is a k-core of a graph G if the degree of all nodes in H_k is greater than or equal to k and H_k is the maximum subgraph of G with this property. The k-core decomposition of a network allows to measure the network cohesiveness. If the network is cohesive, it should be possible to remove some of its members without fragmenting the subset. This also implies that messages can reach their destinations by many alternate (and independent) paths. The LaNetvi tool was used to obtain and visualize the k-core decomposition of the logical network resulting from executing Algorithm 4.3.3 on a network of 5000 nodes. Fig. 16 shows the logical networks obtained in the first and tenth run. In these figures, the size of a node represents its degree (left legend) and colors represent cores (right legend). Higher-numbered cores are the more internal components of the networks. Both of these visualizations reveal highly hierarchical structures and high cohesiveness, which are key properties for allowing a fault-tolerant behavior. However, the cores obtained during the tenth run have the property of being more densely connected than those obtained during the first run, which implies that the logical network presents more robust routing capabilities as it evolves.

5.5. Semantic community structure of the evolved logical networks

It is useful to analyze the graphical representation of the logical network as it allows to determine whether semantic communities have emerged from the nodes' interactions. Fig. 17 shows a logical network obtained with a network of 1000 nodes after running Algorithm 1.0. In this case the nodes only know their physical neighbors, disregarding the topics of interest associated with the rest of the pairs. As a consequence, this network does not reflect the existence of semantic communities. On the contrary, the logical networks presented in Figs. 18 and 19, resulting from the third and seventh run

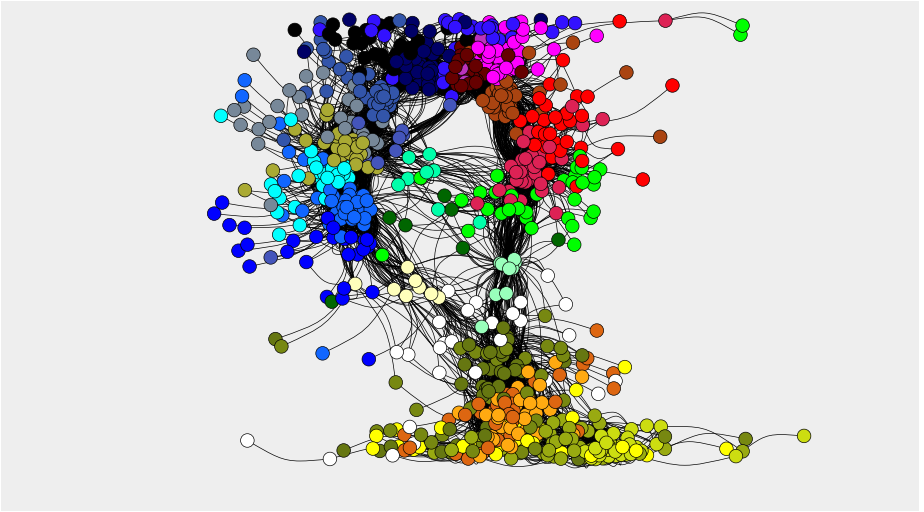

Fig. 18. Logical network obtained with Algorithm 4.3.3 (run = 3, $|N|$=1000).

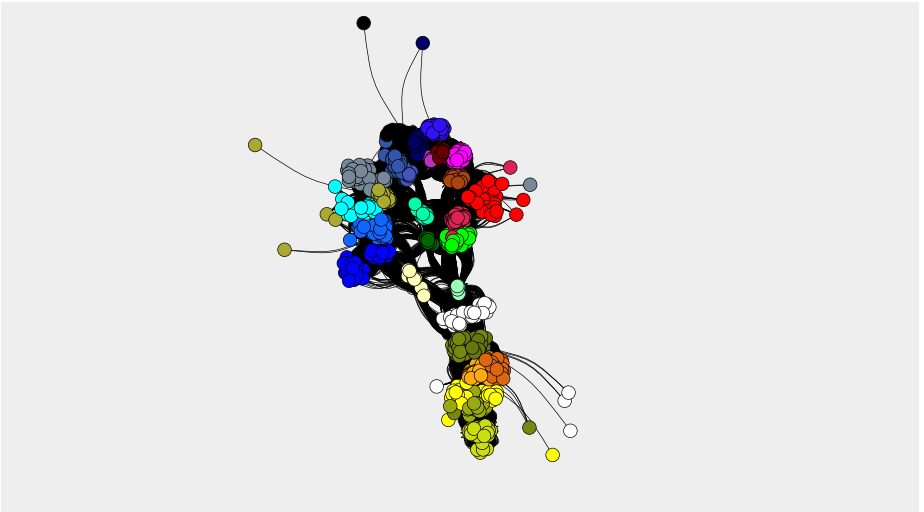

Fig. 19. Logical network obtained with Algorithm 4.3.3 (run = 7, $|N|$=1000).

of Algorithm 4.3.3, clearly indicate a natural division of the network into groups of related nodes. While a high clustering coefficient is an indication of high interconnectivity among the neighbors of nodes in a network, the visualizations presented here allow to recognize that these connections give rise to semantic communities. Colors are related to the topic in which each node is interested and allow to appreciate the grouping of nodes by similar topics. For instance, blue is associated with biological sciences while light blue is associated with health sciences, which is semantically similar to biological sciences. It may be the case that in the physical network a pair of nodes is far apart but the same pair is adjacent in the logical network. This is because the logical network reflects the semantic aspects of the nodes.

6. Conclusions and future work

The driving hypothesis for this work was that better communication patterns can emerge by learning about other peers' interests in a P2P network. Based on this assumption, we have proposed a family of algorithms that adopt different strategies for learning about other peers.

The results obtained by evaluating our algorithms through extensive simulations at different scales confirmed our hypothesis and indicate that the algorithms with better behavior are those that induce greater collaboration among peers (that is, when a node learns information about other peers, it should spread this knowledge across its community). Algorithms 4.3.3 and 5.0 make it possible for nodes to share knowledge with the community. For these algorithms we can see that although the number of initial messages sent is high, the number quickly decreases as the knowledge increases, requiring

fewer hops to reach an answer. Furthermore, these algorithms give rise to logical networks with semantic community structure (as indicated by the high clustering coefficient and networks' visualizations) and a small diameter (represented by the average hop count), revealing the formation of a logical network with a small-world topology.

Learning not only takes place when a node answers a query, but also when the node that generated the query is found to be semantically similar to the answering node. In this case, learning occurs independently of whether the node replies or does not reply the query. Another important result obtained from the simulations is that the proposed algorithms produce a similar impact on the logical network independently of whether the network is composed of 1000 or 5000 nodes, suggesting that the algorithms scale well to larger networks.

Part of our future work will focus on performing search based on semantic criteria, going beyond the currently implemented syntactic search mechanisms. For example, if a query contains the term "computer", then articles referring to PCs or to notebooks will be returned. Searching by semantic similarity occasionally results in a slight loss of precision but usually increases recall. Moreover, ambiguities could be reduced by filtering results based on the user's context. Further strategies aimed at improving performance will be implemented and tested. For instance, better precision and recall could be achieved by incrementally learning good topic descriptors and discriminators, in the way proposed in Lorenzetti and Maguitman (2009).

A limitation that we have observed is what we could describe as the "Closed Communities Problem". In this setting, one or more nodes can be separated from their community or can form another community with the same topic without knowing of each other. To solve this problem we plan to implement a curiosity mechanism that will prompt some participants to explore the network beyond their interests. Finally, we plan to run these algorithms in a more realistic distributed environment where the participants could occasionally change their interests and generate queries dynamically. Research in this direction is currently underway.

Acknowledgment

This work was supported by CONICET (PIP 11220120100487), MinCyT (PICT 2014-0624), and Universidad Nacional del Sur (PGI-UNS 24/N039).

References

Akavipat, R., Wu, L.-S., Menczer, F., & Maguitman, A. G. (2006). Emerging semantic communities in peer web search. In *Proceedings of the international workshop on information retrieval in peer-to-peer networks*. In *P2PIR '06* (pp. 1–8). New York, NY, USA: ACM. doi:10.1145/1183579.1183581.

Arour, K., & Yeferny, T. (2014). Learning model for efficient query routing in P2P information retrieval systems. *Peer-to-Peer Networking and Applications*, 1–17. doi:10.1007/s12083-014-0282-2.

Balakrishnan, H., Kaashoek, M. F., Karger, D., Morris, R., & Stoica, I. (2003). Looking up data in P2P systems. *Communications of the ACM, 46*, 43–48. doi:10.1145/606272.606299.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science, 286*(5439), 509–512. doi:10.1126/science.286.5439.509.

Barbosa, M. W., Costa, M. M., Almeida, J. M., & Almeida, V. A. F. (2004). Using locality of reference to improve performance of peer-to-peer applications. *SIGSOFT Software Engineering Notes, 29*(1), 216–227. doi:10.1145/974043.974079.

Bawa, M., Bayardo Jr. , R. J., Rajagopalan, S., & Shekita, E. J. (2003). Make it fresh, make it quick – searching a network of personal webservers. In *Proceedings of the 12th international world wide web conference*. In *WWW '03* (pp. 577–586). New York, NY, USA: ACM. doi:10.1145/775152.775234.

Broder, A., & Mitzenmacher, M. (2004). Network applications of bloom filters: A survey. *Internet mathematics, 1*(4), 485–509. doi:10.1080/15427951.2004.10129096.

Burt, R. S. (1992). *Structural holes: The social structure of competition*. Harvard University Press.

Carlsson, B., & Gustavsson, R. (2001). The rise and fall of napster - an evolutionary approach. In *Proceedings of the 6th international computer science conference on active media technology*. In *AMT '01* (pp. 347–354). London, UK, UK: Springer-Verlag. doi:10.1007/3-540-45336-9_40.

Chen, H., Gong, Z., & Huang, Z. (2005). Self-learning routing in unstructured p2p network. *International Journal of Information Technology, 11*(12), 59–67.

Crespo, A., & Garcia-Molina, H. (2002). Routing indices for peer-to-peer systems. In *Proceedings of the 22nd international conference on distributed computing systems (icdcs'02)* (pp. 23–32). IEEE Computer Society. doi:10.1109/ICDCS.2002.1022239.

Doulamis, N. D., Karamolegkos, P. N., Doulamis, A., & Nikolakopoulos, I. (2009). Exploiting semantic proximities for content search over p2p networks. *Computer Communications, 32*, 814–827. doi:10.1016/j.comcom.2008.12.005.

Du, N., Wang, B., & Wu, B. (2008). Community detection in complex networks. *Journal of Computer Science and Technology, 23*, 672–683. doi:10.1007/s11390-008-9163-6.

Eftychiou, A., Vrusias, B., & Antonopoulos, N. (2012). A dynamically semantic platform for efficient information retrieval in P2P networks. *International Journal of Grid and Utility Computing, 3*, 271–283. doi:10.1504/IJGUC.2012.051424.

Erdős, P., & Rényi, A. (1959). On random graphs. *Publicationes Mathematicae Debrecen, 6*, 290–297.

Everett, M., & Borgatti, S. P. (2005). Ego network betweenness. *Social Networks, 27*(1), 31–38. doi:10.1016/j.socnet.2004.11.007.

Gómez Santillán, C., Cruz Reyes, L., Meza Conde, E., Schaeffer, E., & Castilla Valdez, G. (2010). A self-adaptive ant colony system for semantic query routing problem in P2P networks. *Computación y Sistemas (CyS), 13*(4), 433–448.

Guo, L., Jiang, S., Xiao, L., & Zhang, X. (2004). Exploiting content localities for efficient search in P2P systems. In R. Guerraoui (Ed.), *Distributed computing*. In *Lecture Notes in Computer Science: 3274* (pp. 349–364). Springer Berlin Heidelberg. doi:10.1007/978-3-540-30186-8_25.

Ismail, A., Quafafou, M., Nachouki, G., & Hajjar, M. (2010). A global knowledge for information retrieval in P2P networks. In *Internet and web applications and services (iciw), 2010 fifth international conference on* (pp. 229–234). doi:10.1109/ICIW.2010.41.

Jin, X., & Chan, S.-H. G. (2010). Unstructured peer-to-peer network architectures. In X. Shen, H. Yu, J. Buford, & M. Akon (Eds.), *Handbook of Peer-to-Peer Networking* (pp. 117–142)). Springer. doi:10.1007/978-0-387-09751-0_5.

Joseph, S. (2002). Neurogrid: Semantically routing queries in peer-to-peer networks. In E. Gregori, L. Cherkasova, G. Cugola, F. Panzieri, & G. Picco (Eds.), *Web engineering and peer-to-peer computing*. In *Lecture Notes in Computer Science: 2376* (pp. 202–214). Springer Berlin Heidelberg. doi:10.1007/3-540-45745-3_18.

Kalnis, P., Ng, W. S., Ooi, B. C., & Tan, K.-L. (2006). Answering similarity queries in peer-to-peer networks. *Information Systems, 31*(1), 57–72. doi:10.1016/j.is.2004.09.003.

Korzun, D., & Gurtov, A. (2013). *Structured peer-to-Peer systems: Fundamentals of hierarchical organization, routing, scaling, and security*. Springer. doi:10.1007/978-1-4614-5483-0.

Li, J., Loo, B. T., Hellerstein, J. M., Kaashoek, M. F., Karger, D. R., & Morris, R. (2003). On the feasibility of peer-to-peer web indexing and search. In M. F. Kaashoek, & I. Stoica (Eds.), *Proceedings of the 2nd international workshop on peer-to-peer systems (iptps '03)*. In *LNCS: 2735* (pp. 207–215). Springer-Verlag. doi:10.1007/978-3-540-45172-3_19.

Lorenzetti, C. M., & Maguitman, A. G. (2009). A semi-supervised incremental algorithm to automatically formulate topical queries. *Information Sciences, 179*(12), 1881–1892. Including Special Issue on Web Search doi: 10.1016/j.ins.2009.01.029.

Meng, F., Ding, L., Peng, S., & Yue, G. (2013). A p2p network model based on hierarchical interest clustering algorithm. *Journal of Software, 8*(5), 1262–1267. doi:10.4304/jsw.8.5.1262-1267.

Merabti, M., Liu, Z., Yu, H., & Kundur, D. (2010). Advances in peer-to-peer content search. *Journal of Signal Processing Systems, 59*, 309–318. doi:10.1007/s11265-009-0343-6.

Nicolini, A. L., Lorenzetti, C. M., Maguitman, A. G., & Chesñevar, C. I. (2013). Intelligent Algorithms for Reducing Query Propagation in Thematic P2P Search. In *Anales del XIX Congreso Argentino de Ciencias de la Computación (CACIC)* (pp. 71–79). Mar del Plata, Buenos Aires, Argentina.

Pongor, G. (1993). Omnet: Objective modular network testbed. In *Proceedings of the international workshop on modeling, analysis, and simulation on computer and telecommunication systems*. In *MASCOTS '93* (pp. 323–326). San Diego, CA, USA: Society for Computer Simulation International. URL http://dl.acm.org/citation.cfm?id=646600.698549.

Qin, C., Yang, Z., & Liu, H. (2014). User interest modeling for p2p document sharing systems based on k-medoids clustering algorithm. In *Seventh international joint conference on computational sciences and optimization (CSO)* (pp. 576–578). IEEE. doi:10.1109/CSO.2014.113.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America, 101*(9), 2658–2663. doi:10.1073/pnas.0400054101.

Ripeanu, M. (2001). Peer-to-peer architecture case study: Gnutella network. In *Proceedings of the first international conference on peer-to-peer computing* (pp. 99–100). doi:10.1109/P2P.2001.990433.

Rosenfeld, A., Goldman, C. V., Kaminka, G. A., & Kraus, S. (2009). Phirst: A distributed architecture for p2p information retrieval. *Information Systems, 34*(2), 290–303. doi:10.1016/j.is.2008.08.002.

Seidman, S. B. (1983). Network structure and minimum degree. *Social Networks, 5*(3), 269–287. doi:10.1016/0378-8733(83)90028-X.

Sharan, A. (2006). *Exploiting semantic locality to improve peer-to-peer search mechanisms*. Ph.D. thesis. Rochester Institute of Technology.

Shen, X.-J., Chang, Q., Gou, J.-P., Mao, Q.-R., Zha, Z.-J., & Lu, K. (2016). Collaborative Q-Learning Based Routing Control in Unstructured P2P Networks. In *Multimedia modeling* (pp. 910–921). Springer. doi:10.1007/978-3-319-27671-7_76.

Sripanidkulchai, K., Maggs, B., & Zhang, H. (2003). Efficient content location using interest-based locality in peer-to-peer systems. In *Proceedings of the twenty-second annual joint conference of the ieee computer and communications: 3* (pp. 2166–2176). IEEE. doi:10.1109/INFCOM.2003.1209237.

Steyvers, M., & Tenenbaum, J. B. (2005). The large-scale structure of semantic networks: statistical analyses and a model of semantic growth. *Cognitive science, 29*(1), 41–78.

Sujeeth, T., Kumar Kond Reddy, M., & Kumar Reddy D. , K. (2013). A selective neighbor search algorithm in unstructured peer-to-peer networks. *International Journal of Latest Trends in Engineering and Technology, 2*(4), 408–411. URL http://ijltet.org/wp-content/uploads/2013/07/60.pdf.

Tang, C., Xu, Z., & Dwarkadas, S. (2003). Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proceedings of the 2003 conference on applications, technologies, architectures, and protocols for computer communications*. In *SIGCOMM '03* (pp. 175–186). New York, NY, USA: ACM. doi:10.1145/863955.863976.

Tigelaar, A. S., Hiemstra, D., & Trieschnigg, D. (2012). Peer-to-peer information retrieval: an overview. *ACM Transactions on Information Systems (TOIS), 30*(2), 9:1–9:34. doi:10.1145/2180868.2180871.

Tirado, J. M., Higuero, D., Isaila, F., Carretero, J., & Iamnitchi, A. (2010). Affinity P2P: A self-organizing content-based locality-aware collaborative peer-to-peer network. *Computer Networks, 54*(12), 2056–2070. doi:10.1016/j.comnet.2010.04.016.

Varga, A., & Hornig, R. (2008). An overview of the omnet++ simulation environment. In *1st international conference on simulation tools and techniques for communications, networks and systems & workshops*. In *Simutools '08* (pp. 60:1–60:10). Brussels, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). URL http://dl.acm.org/citation.cfm?id=1416222.1416290.

Voulgaris, S., Kermarrec, A., Massouli, L., & van Oteen, M. (2004). Exploiting semantic proximity in peer-to-peer content searching. In *Proceedings of the 10th IEEE international workshop on future trends of distributed computing systems* (pp. 238–243). Washington, DC, USA: IEEE Computer Society. doi:10.1109/FTDCS.2004.1316622.

Wang, L. (2011). Sofa: An expert-driven, self-organization peer-to-peer semantic communities for network resource management. *Expert Systems with Applications, 38*, 94–105. doi:10.1016/j.eswa.2010.06.020.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature, 393*(6684), 440–442. doi:10.1038/30918.

Wu, K., & Wu, C. (2013). State-based search strategy in unstructured p2p. *Future Generation Computer Systems, 29*(1), 381–386. doi:10.1016/j.future.2011.08.002.

Yan, F., & Zhan, S. (2004). A peer-to-peer approach with semantic locality to service discovery. In H. Jin, Y. Pan, N. Xiao, & J. Sun (Eds.), *Grid and cooperative computing - gcc 2004*. In *Lecture Notes in Computer Science: 3251* (pp. 831–834). Springer Berlin Heidelberg. doi:10.1007/978-3-540-30208-7_116.

Yeferny, T., & Arour, K. (2010). LearningPeerSelection: A query routing approach for information retrieval in P2P systems. In *Internet and web applications and services (ICIW), 2010 fifth international conference on* (pp. 235–241). doi:10.1109/ICIW.2010.42.

Yu, Y.-T., Gerla, M., & Sanadidi, M. (2015). Scalable vanet content routing using hierarchical bloom filters. *Wireless Communications and Mobile Computing, 15*(6), 1001–1014. doi:10.1002/wcm.2495.

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2004). Information retrieval techniques for peer-to-peer networks. *Computing in Science Engineering, 6*(4), 20–26. doi:10.1109/MCSE.2004.12.

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2005). Exploiting locality for scalable information retrieval in peer-to-peer networks. *Information Systems, 30*(4), 277–298. doi:10.1016/j.is.2004.03.001.

Zhu, T., Wu, B., & Wang, B. (2009). Social influence and role analysis based on community structure in social network. In *Proceedings of the 5th international conference on advanced data mining and applications*. In *ADMA '09* (pp. 788–795). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-642-03348-3_84.