Peer-to-Peer Semantic Search Engine

Tomáš Havryluk, Ivan Jelínek

Abstract: *Peer-to-Peer is a relatively old but still an evolving branch of web technologies. This article gives a survey of Peer-to-Peer characteristics and their relationship to requirements usually set upon Peer-to-Peer systems. From this analysis flows the idea of building a three-layer Peer-to-Peer semantic search engine.*

Key words: *Peer-to-Peer, P2P, requirements, semantics, multi-ring*

INTRODUCTION

The growing amount of information on the Internet brings many problems for the central oriented systems searching the web. Because most of these problems come out from the basic fact that the whole system consists of one central element, there is a tendency to develop systems based on the peer-to-peer architecture [2].

FROM GOOGLE TO PEER-TO-PEER

Central search engines like Google are based on a central index (a database) that contains data gathered by a program – called a robot or a spider - specially designed for their collecting from the Internet. There are many problems resulting from this way of searching. The volume of the searched data is very large and still growing, therefore searching based on central indexes isn't flexible enough – index update lasts very long. The data is also encapsulated in various formats in which the spiders are not able to search. There is a very weak support of communities containing interrelated data that hangs together with the problem of getting relevant data during the search. The whole search system depends on the performance of one active central server, so in case of a failure of this central entity, the whole network is functionless [2].

These problems can be solved using the Peer-to-Peer search engine model. In this model the indexes are local and always up-to-date (less data to manage), the queries are processed individually by each node (various formats support), nodes can implement more time-consuming algorithms, it is easy to support communities and failure of a single node does not affect the whole system. But the use of distributed search networks also brings many disadvantages and problems that have to be solved. One of them is for example the latency of the network that goes hand to hand with the problem of flooding the network with lots of data. Both of these depend on the topology of the network and used routing algorithms when distributing the queries among particular nodes.

EXAMPLES OF REALIZED SYSTEMS

Napster

Distributed search engine substituting the Google is not the only one way of use of the Peer-to-Peer architecture. Probably the mostly known system that put the term of Peer-to-Peer to mind of general public was Napster. Its main purpose was to allow the users to share and search audio files (mp3s) via Internet. However Napster is not a true Peer-to-Peer system. Changing the data (files) is done through the direct connection of interacting nodes but the search process is realized by central servers. The using of the system consists of two steps. First, the peer connects to the server and uploads information about the shared content. The way the information is stored on the server is similar to indexing with the difference that the server gets the information of data occurrence automatically and does not have to search the node himself. After this the connected node can send queries for specific data and gets the response in the form of

addresses of nodes actually sharing the requested files [2].

Gnutella

Typical representative of fully decentralized systems for sharing and searching data is Gnutella. Like other systems it builds the Peer-to-Peer architecture at the application level. This means creating logical network over the physical one, where even not directly connected nodes can act as neighbours. The topology of the network can be described as a random diagram where each node stores list of links to his neighbours. These are used for passing the queries for data to other nodes.

The method used for searching the network is called flooding. A node that looks up for some data specifies its request to a message called "Query" and sends it to all of his neighbours. Each of them sends the message to their neighbours etc. After receiving the query nodes reply with "QueryHit" message if they have the requested data or do not react at all. Because the nodes are connected randomly there can be circles in the network topology. Using flooding means that messages could be passed in those circles forever. Therefore each query is assigned unique ID and each node writes down all the IDs that passed by in recent time and ignores all incoming messages with the same ID [8].

FreeNet

In both of the already mentioned networks - Gnutella and Napster, the data exchange is realized through direct connection of two nodes, the data source is always known to the asking node. That means no anonymity - which is one of the characteristics of the FreeNet network. In order to use sources provided by FreeNet peers, it is necessary to create request for data that is passed to other nodes. The message travels through the network till the maximum number of visited nodes expires or there is a node that replies. Like in the Gnutella system, the reply is send back to the requesting peer over the same path it arrived. The difference is that with the reply goes back the data too. It has the advantage that all forwarding nodes can store the forwarded data for later use in reply instead of forwarding the same query next time it arrives. It causes that frequently asked data are distributed over many nodes which decreases the time for their locating.

The fact that each node knows only his immediate nodes leads to the situation that both the consumers and the producers of the data are very hard to identify, which means high level of anonymity in the network [2].

Each of the already mentioned systems has its advantages and disadvantages. Napster is fast when looking up for data because the file lists are uploaded on central servers. That means a big disadvantage too, because in case of shutting down these central entities the whole service becomes unavailable. The next fact is that Napster can not safeguard anonymity of communicating nodes. The biggest disadvantages of the Gnutella network become from its way of connecting the peers and searching the network. Random connection of peers and usage of flooding algorithm to search the network results in unequal distribution of data flow among the peers and often in flooding the network[8]. Finally the FreeNet does not always guarantee that a file is found, even if the file is in the network and has a potentially long search path in a large network. Finally none of the mentioned systems use semantics to search for the data [2]. Of course that the named networks do not represent the whole world of Peer-to-Peer systems, they were used only to demonstrate some of the possible directions of Peer-to-Peer design and problems associated with this task. More detail description of requirements set on Peer-to-Peer networks is included in the next paragraph.

THE ANALYSIS OF NETWORK PERFORMANCE

It is obvious that the goal is to design and describe system that provides the best

performance towards deposited requirements. What are these requirements and what is their meaning? On which design steps depends the performance?

Latency, Communication load, Equality, Flexibility

Latency means the time needed to search the whole network (if possible) for specified data. It depends on the number of nodes in the network, the network topology and on the search and routing algorithms too [2]. With the algorithms and topology is also connected the question of communication load in the network and its (at least approximate) distribution among all nodes – their equality [5]. The usage of proper ways to distribute the queries on known topology can provide low latency and low and equal communication load of the nodes. Naturally, it all depends on the number of nodes in the network and the amount of produced queries too. Finally it is necessary to keep in mind the flexibility of the network, which means the possibility of connecting and disconnecting nodes and overhead of this operations as well as the ability of reconfiguration after node failures. Described dependencies are displayed on Figure 1.

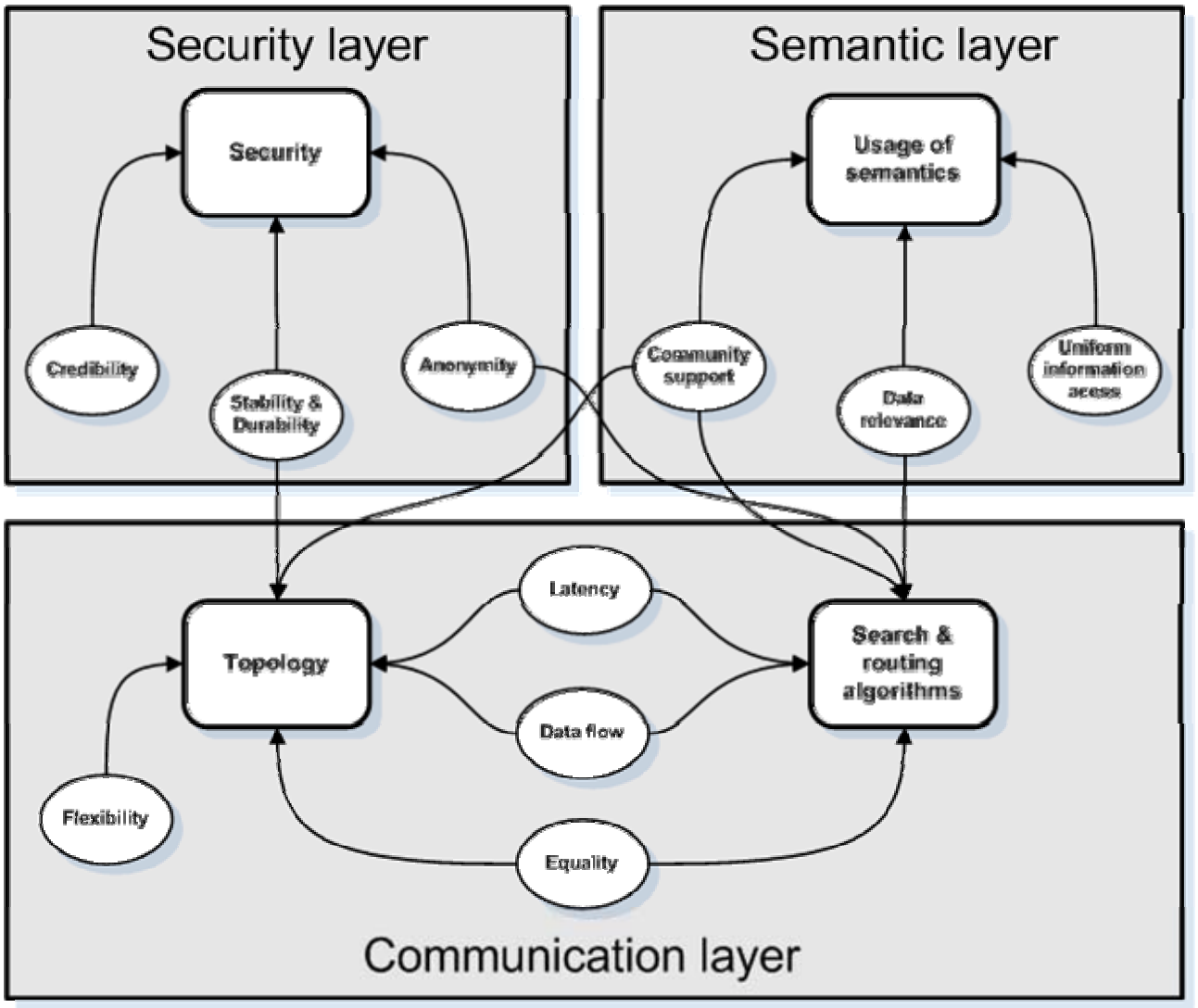

Figure 1: Peer-to-Peer requirements dependency on design steps

Data relevance, Community support, Uniform information access

With uncontrollable growth of information on the Internet it is still harder for the user to search for the requested information. In this situation it is upon each node to evaluate the received query using appropriate semantics and offer a maximum relevant answer. At the same time the community support offers opportunity for distributing queries among

topic-related nodes because there is no need to send messages to nodes that probably do not have the searched data and it is a good chance that the data found within the community will be more relevant [6]. The community support is quite connected to the used searching algorithms and network topology, while the principle of the uniform information access [6] allows each node to provide its data stored in various formats, including databases, so it could be searched by nodes that do not even know anything about the inner data interpretation. The uniform information access allows each node to use proper ontology too.

Stability & Durability, Anonymity, Credibility

One of the advantages of distributed systems is the absence of central elements whose failure leads to malfunction of the whole system. We can talk about stability of the network or durability of the stored data like the ability to handle node failures as well as attempts to shut down the network. This fact is closely connected to the question of the level of security of the network and partially depends on the network topology too. The anonymity takes its place when it is not desirable to identify the source nodes of data or the data requests. As a result from the description of the FreeNet, the anonymity goes hand to hand with the way of routing the messages and can be taken as a part of the network security too. The next important requirement is the credibility of the network which means to safeguard immunity towards any attempts to compromise the network producing incorrect information or fake files. [3]

All the mentioned requirements and their dependencies on network properties are displayed on Figure 1. The fact that a requirement (shown as an oval) depends in such a way on a network property (a rounded rectangle) is symbolized with an arrow leading from the requirement to the property it depends on. It is a question if all of the requirements could be even satisfied together [2]. The properties and requirements have been grouped into three layers as seen on the picture which means the first step in designing a new Peer-to-Peer system.

DESIGNING A PEER-TO-PEER NETWORK

What does it include to create a correctly working Peer-to-Peer system that can satisfy the set requirements?

1) Communication layer

The communication layer includes the network topology as well as the message logistics. There are many possible ways of connecting the nodes. The most simple of them is to connect the nodes randomly like in the Gnutella network. This topology is easy to manage (connect and disconnect nodes) but does not guarantee stability or a minimal data bandwidth between nodes situated in far-off parts of the network [8]. More effective is to force the nodes to form a defined structured or at least to form defined substructures that will be connected randomly. In this case comes to account the multi-ring topology which is flexible enough and brings advantages like easy community management (each ring means a community) or selective query propagation (around a ring, into selected rings) [6].

The next part of the communication layer is to choose an optimal algorithm that will provide maximum speed (minimal latency) as well as maximum searched space while processing each query and without flooding the network with large amount of data. It is obvious that the algorithm should be chosen with the reference to the chosen topology also.

2) Security layer

Purpose of this layer is to safeguard stability and durability as well as to make the

network resistant to Denial-of-service (DOS) attacks and attempts to forged stored data [3]. For this purpose are used technologies and techniques like cryptography, Byzantine agreement or correlated failure analysis. Further information of securing a Peer-to-Peer network can be found in [1], [3].

3) Semantic layer

The first purpose of this layer is to ensure maximum relevance of the data provided to the user. It means to process the queries with reference to their semantics and to route them through the network with reference to the topic-related communities, where the probability of getting more relevant data is higher. The routing has close binding to the network topology and directly affects the search algorithms used in the communication level. It is possible that designing this layer can cause many changes in the design of the communication layer [7].

The second purpose of this layer is to provide uniform information access. As mentioned above it is upon each node to use proper ontology for representing the possessed data and to provide it in general known formats [6].

CONCLUSION AND FUTURE WORK

Peer-to-Peer is still an evolving branch of computer science. Although there are many Peer-to-Peer systems, most of them provide sufficient performance only towards a subset of demanded requirements. It is a question if all of the requirements could be ever satisfied together, but it is necessary to design and examine better Peer-to-Peer systems that will provide maximum performance towards the set requirements.

The analysis of network performance shows the existence of dependencies between the set requirements and the network characteristics. That leads to the idea of forming a three-layer semantic search engine. First it is necessary to choose topology and the search algorithms to create the communication layer of the system. The objective of the second layer is to safeguard the security of the whole network which means making the network immune towards attempts of discrediting or even disabling the network. Finally the role of the third-semantic layer is to ensure maximal data relevancy applying the results from the area of semantic web which also means solving the question of the uniform information access. After designing the system it has to be implemented to verify its rightness in experiments, so it could be used in practice.

ACKNOWLEDGEMENT

This research has been supported by MSMT under research program No. 6840770014.

This research has been supported by the grant of the Czech Grant Agency No. 201/06/0648.

REFERENCES

[1] Balakrishnan, H., Kaashoek, M., Karger, D., Morris, R., Stoica, I. Looking Up Data in P2P Systems. Communications of the ACM Magazine, vol. 46, no. 2, 43-48, 2003

[2] Bures, M., A. Moravek, I. Jelinek. New generation of web technologies. Prague: VOX a.s. - Publishing, 2005, in Czech

[3] Kubiatowicz, J. Extracting Guarantees from Chaos. Communications of the ACM Magazine, vol. 46, no. 2, 33-38, 2003

[4] Lee, J. An End-User Perspective on File-Sharing Systems. Communications of the ACM Magazine, vol. 46, no. 2, 49-53, 2003

[5] Lethin, R. Technical and Social Components of Peer-to-Peer Computing. Communications of the ACM Magazine, vol. 46, no. 2, 30-32, 2003

[6] Moravek, A. Peer to Peer Search Engine, Postgraduate Study Report DC-PSR-2004-13, CTU Prague, 2004

[7] Petrie, Ch. Peer to Peer Pragmatic Semantic Unification, IEEE Internet Computing Journal, 1089-7801/05, 96-97, 2005

[8] Ripeanu, M., I. Foster, A. Iamnitchi. Peer-to-Peer Networking: Mapping the Gnutella Network, IEEE Internet Computing Journal, 1089-7801/02, 50-57, 2002

ABOUT THE AUTHORS

Tomas Havryluk, Ivan Jelinek, Department of Computer Science and Engineering, Czech Technical University, Phone: +420 777 726 470, E-mail: havryt1@fel.cvut.cz