“P2P Techniques for Web Information Retrieval and Filtering”

by Christos Tryfonopoulos

Much information of interest to humans is today available on the Web. People can easily gain access to information but at the same time, they have to cope with the problem of information overload. Consequently, they rely on specialised tools and systems designed for searching, querying and retrieving information from the Web. It is however extremely difficult to stay informed without sifting through enormous amounts of incoming information, and without utilising tools and techniques that would capture the dynamic nature of the Web.

We are interested in the problem of distributed resource sharing in wide-area networks such as the Internet and the Web. In the architecture that we envision, resources are annotated using attribute-value pairs, where value is of type text, and queried using constructs from Information Retrieval models. There are two kinds of basic functionality that we expect this architecture to offer: information retrieval (IR) and information filtering (IF) (also known as publish/subscribe or information dissemination). In an IR scenario a user poses a query (e.g., “I am interested in papers on bio-informatics”) and the system returns a list of pointers to matching resources. In an IF scenario, a user posts a subscription (or profile or continuous query) to the system to receive notifications whenever certain events of interest take place (e.g., when a paper on bio-informatics becomes available).

The Architecture

Selective dissemination of information to interested users is a problem that has received attention from various research communities including researchers from Information Retrieval, Databases, Distributed Computing, Digital Libraries, Agent Systems and others. In this article we

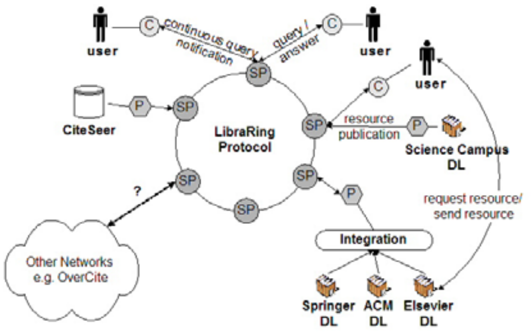

Figure 1. The architecture of LibraRing

will use the Digital Library (DL) paradigm to demonstrate our solutions for the Information Filtering case. We envision a distributed DL architecture (called LibraRing from the words *library* and *ring*) like the one shown in Figure 1.

In this architecture nodes can implement any of the following types of services: super-peer service, provider service and client service. Nodes implementing the super-peer service (super-peers) form the message routing layer of the network. Each super-peer is responsible for serving a fraction of the clients by storing continuous queries, matching them against incoming (published) documents and creating notifications. The super-peers run a Distributed Hash Table (DHT) protocol which is an extension of Chord. A node implementing the client service (client) connects to the network through a single super-peer node, which is its access point. Clients can connect, disconnect or even leave the system silently at any time. Clients are information consumers: they can subscribe to resource publications with continuous queries and receive notifications about published resources (e.g., documents) that match their interests. Finally, the provider service (provider) is implemented by information sources that want to expose their contents to the clients of the system. A node implementing this service connects to the network through a super-peer which is its access point. To be able to implement this service, an information source creates meta-data for the documents it stores using an appropriate data model and publishes it to the rest of the network using its access point.

The main focus of our work is on providing models and languages for expressing publications and subscriptions, protocols that regulate super-peer interactions and query indexing mechanisms that are utilized by each one of the super-peers.

Data model & query language

Publications and subscriptions in our architecture could be expressed using any appropriate language (e.g., XML and XPath). Whatever language is chosen will have a serious effect on the DHT protocols, as the DHT is the layer in which publications and subscriptions are indexed. We use a well-understood attribute-value model, called *AWPS*, that is based on named attributes with free text as value interpreted under the Boolean and VSM (or LSI) models. The query language of *AWPS* allows Boolean combinations of comparisons *A op v*, where *A* is an attribute, *v* is a text value and *op* is one of the operators “equals”, “contains” or “similar” (“equals” and “contains” are Boolean operators and “similar” is interpreted using the VSM or LSI model).

Indexing queries locally

In the architecture described above, clients subscribe to their access points with continuous queries that express their information needs, and providers expose their content using an appropriate meta-data model. Each super-peer is responsible for storing the queries, so that whenever a resource is published, the continuous queries satisfying it are found and notifications are sent to the appropriate clients. This work deals with the filtering problem that needs to be solved efficiently by each super-peer: Given a database of continuous queries *db* and a document *d*, find all queries *q* in *db* that match *d*. We have proposed data structures and indexing algorithms that enable us to solve the filtering problem efficiently for large databases of queries expressed in the model *AWP*, which is the Boolean subset of *AWPS*, and is based on named attributes with values of type text, and word proximity operators.

The main idea behind these algorithms is to store sets of words compactly by exploiting their common elements. In algorithm PrefixTrie a query is considered as a sequence of words sorted in lexicographic order, and a trie is used to store queries compactly by exploiting common prefixes. Algorithm BestFitTrie constitutes an improvement over PrefixTrie. BestFitTrie keeps the main idea behind PrefixTrie but (a) handles the words contained in a query as a set rather than as a sorted sequence and (b) searches exhaustively the forest of tries to discover the best place to store a new query. This allows BestFitTrie to achieve better clustering of the queries and thus smaller filtering times. Finally, algorithm ReTrie improves over BestFitTrie by considering the periodic re-organisation of the query database.

The DHTrie protocols

In our most recent work, we showed how to provide IR and IF functionality by using an extension of the Chord DHT. DHTs are the second generation structured P2P overlay networks devised as a remedy for the known limitations of earlier P2P networks such as Napster and Gnutella. They are the distributed version of the hash table data structure, and can locate a data item stored at any node in O(*log N)* messages, where N is the number of nodes.

In this context, we presented a set of protocols, collectively called DHTrie, that extend the Chord protocols with IR and IF functionality assuming that publications and subscriptions are expressed in the model *AWPS*. We also showed experimentally that local data structures and simple routing optimisations can make a big difference in a DHT environment. The experiments also showed that our protocols are scalable: the number of messages needed to publish a document and notify interested subscribers remains almost constant as the network grows. Moreover, the increase in message traffic shows little sensitivity to increase in document size. Since probability distributions associated with publication and query elements are expected to be skewed in such a scenario, achieving a balanced load among the nodes becomes an important problem. Thus, we studied an important case of load balancing for DHTrie and presented a new algorithm, based on the idea of load-shedding, which is also applicable to the standard DHT lookup problem.

Outlook

Distributed IR as studied here can benefit from techniques of traditional IR, distributed systems and networking (especially P2P networks), databases and distributed AI. With this perspective in mind our current work concentrates on two open problems: providing algorithms for maintaining word statistics in distributed environments and addressing the load balancing problem in the standard DHT lookup scenario.

Christos Tryfonopoulos is currently pursuing a PhD at the Technical University of Crete under the supervision of Associate Professor Manolis Koubarakis. He received his B.Sc. in Computer Science from University of Crete, and his M.Sc. in Computer Engineering from the Technical University of Crete. His research interests include IR and pub/sub over wide-area networks, P2P and Grid computing, and multi-agent systems. He can be contacted via: trifon@intelligence.tuc.gr