$\mathcal{DS}^4$: A Distributed Social and Semantic Search System*

Dionisis Kontominas[1], Paraskevi Raftopoulou[1],
Christos Tryfonopoulos[1], and Euripides G.M. Petrakis[2]

[1] University of Peloponnese, Greece {cst06086,praftop,trifon}@uop.gr
[2] Technical University of Crete, Greece petrakis@intelligence.tuc.gr

Abstract. We present $\mathcal{DS}^4$, a Distributed Social and Semantic Search System that allows users to share and search for content among friends and clusters of users that specialise on the query topic. In $\mathcal{DS}^4$, nodes that are semantically, thematically, or socially similar are automatically discovered and logically organised into groups. Content retrieval is then performed by routing the user query towards social friends and clusters of nodes that are likely to answer the query. In this way, search receives two facets: the social facet, addressing friends, and the semantic facet, addressing nodes that are semantically close to the query. $\mathcal{DS}^4$ is scalable (requires no centralised component), privacy-aware (users maintain ownership and control over their content), automatic (requires no intervention by the user), general (works for any type of content), and adaptive (adjusts to changes of user content or interests). In this work, we report on our effort to design the next generation of social networks that will offer open and adaptive design, and privacy-aware content management.

1 Introduction

In recent years a number of social networking services have been developed to offer users a new way of sharing, searching, and commenting on user-generated content. Following the development of such services, people have shown great interest in participating in 'social' activities by generating and sharing vast amounts of content, ranging from personal vacation photos, to blog posts or comments, and like/agree/disagree tags. All these social networking services are typically provided by a centralised site where users need to upload their content, thus giving away access control and ownership rights to make it available to others. This centralised administrative authority may utilise the content in any profitable way, from selling contact details to marketing firms, to mining of user information for advertising purposes. Furthermore, the rate of growth of both content and user participation in such services raises significant concerns on the scalability of the centralised architectures used as they are called to serve millions of users and gigabytes of content every day.

In this work, we present a distributed content sharing architecture that allows users to share and search for content in a fully decentralised way, while at the same time maintaining access control and ownership of their content. Our work builds upon research results from the P2P paradigm, such as those utilising unstructured, small-world, and semantic overlay networks (SONs) [1, 6,

* This research leading to these results has received funding from the European Communitys Seventh Framework Program (FP7/2007-2013) under grant agreement No 296170 (Project PortDial).

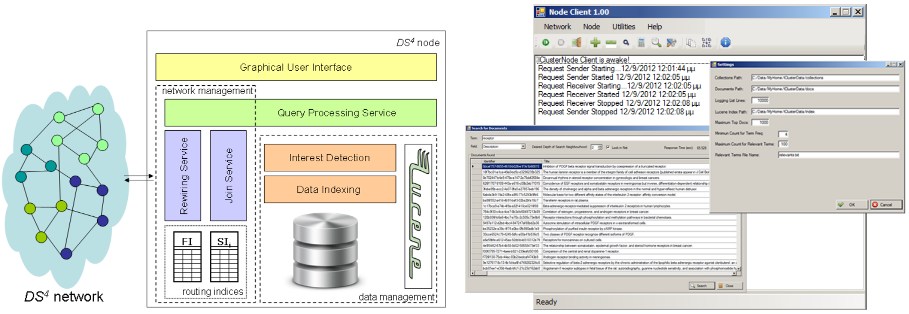
8, 2, 5]. Replacing the centralised authority with a distributed self-manageable community of nodes, removes access control and ownership issues while at the same time ensures high-scalability and low maintenance costs. For this reason recent efforts in the industry and the literature have also resorted to the P2P paradigm to build decentralised online social networks/platforms (like Diaspora, KrawlerX, OpenSocial and [3, 4]), mainly by relying on distributed hash tables (DHTs) to provide architectures [3] or prototype systems [4]. Contrary to DHTs that focus on providing accurate location mechanisms, $\mathcal{DS}^4$ emphasises on node autonomy, content-based clustering of nodes, and loose component architecture by applying the SON paradigm. In $\mathcal{DS}^4$, node organisation is achieved through a *rewiring protocol* that is (periodically) executed by each node. This protocol operates by establishing connections among *semantically similar nodes* (in addition to the social connections) and by discarding connections that are outdated or pointing to dissimilar nodes. The goal of the rewiring protocol is to create *clusters of nodes* with similar interests. User queries can then be resolved by routing the query towards friends and nodes specialising to the query topic. In this way, content search is leveraged to another type of friendship often ignored in social networks: the *semantic friendship* emerging from common user interests.

2 System Overview

Architecture. We consider a distributed social network, where each user, characterised by its interests, is connected to friends and other network nodes sharing similar interests. The interests of a user are identified automatically, i.e., by applying clustering on its local content repository. The network nodes use a rewiring service and form clusters based on their likelihood to have similar interests. Each user maintains two *routing indices* holding information for *friend* and *short/long-range* links to other network nodes. Friend links correspond to the social relationship aspect of the network, short-range links correspond to *intra-cluster* information (i.e., links to nodes with similar interests), while long-range links correspond to *inter-cluster* information (i.e., links to nodes having different interests). The latter are used to maintain connectivity of remote clusters in the system. The reorganisation (or rewiring) procedure is executed locally by each node and aims at clustering nodes with similar content, so as to allow forwarding queries to friends and node clusters that are similar to the issued query.

The main idea behind $\mathcal{DS}^4$ is to let nodes that are semantically, thematically, and socially close self-organise, to facilitate the content search mechanism. The services regulating node join, generation of semantic node clusters, and query processing in $\mathcal{DS}^4$ are discussed in the following sections. Figure 1(a) shows a high-level view of a $\mathcal{DS}^4$ node and the different types of services implemented.

Join Service. When a user node connects to the $\mathcal{DS}^4$ network, its interests are automatically derived by its local content. For each interest, the node maintains a semantic index (*SI*) containing the contact details and interest descriptions of nodes sharing similar interests. These links form the semantic neighborhood of the node; the links contained in *SI* are refined accordingly by using the rewiring service described below. Furthermore, each node maintains a friend index (*FI*) containing the contact details and interest descriptions of the social neighborhood of the node, comprised of explicitly declared friends in the $\mathcal{DS}^4$ network.



```
IClusterNode Client is awake!
Request Sender Starting...12/9/2012 12:01:44 μμ
Request Sender Started 12/9/2012 12:02:05 μμ
Request Receiver Starting...12/9/2012 12:02:05 μμ
Request Receiver Started 12/9/2012 12:02:05 μμ
Request Receiver Stopped 12/9/2012 12:02:08 μμ
Request Sender Stopped 12/9/2012 12:02:08 μμ
```

Collections Path: C:/Data/MyHome/IClusterData/collections
Documents Path: C:/Data/MyHome/IClusterData/docs
Logging List Lines: 10000
Lucene Index Path: C:/Data/MyHome/IClusterData/index
Maximum Top Docs: 1000
Minimum Count for Term Freq: 4
Maximum Count for Relevant Terms: 100
Relevant Terms File Name: relevants.txt

Desired Depth of Search Neighbourhood: 2 ☑ Look in Net Response Time (sec): 65.528

(a) (b)

Fig. 1. (a) $\mathcal{DS}^4$ node architecture and (b) GUI with results and settings screen.

Rewiring Service. The rewiring service is applied to re-organise the semantic neighborhoods of the nodes by establishing new connections and discarding old ones, forming groups of nodes with similar interests. Each node may initiate a rewiring procedure. The node computes its average similarity to its short-range links contained in *SI* as a measure of cluster cohesion. If the similarity computed is greater than a threshold then the node does not need to take any further action, since it is surrounded by nodes with similar interests. Otherwise, the node initiates a cluster refinement process by forwarding a message in the network, with a time-to-live (TTL), using the semantic and social connections and collecting the interests of other nodes.

The issued message is forwarded with equal probability to (i) a number of randomly chosen nodes contained in a node's *SI*, (ii) a number of randomly chosen nodes contained in a node's *FI*, or (iii) the most similar nodes to the message initiator, found in either the *SI* or the *FI*. The rationale of applying either of the forwarding strategies is that the message initiator should be able to reach similar nodes both directly (through other similar nodes), but also indirectly (through propagation of the rewiring message through non-similar nodes). Each node that receives the rewiring message adds its interest in the message, reduces TTL by one, and forwards it in the same manner. When the TTL of the message reaches zero, the message containing the contact info and interests of all nodes that received the message is sent back to its initiator. To speed up the rewiring process, every intermediate node receiving the rewiring message may utilise the message information to refine its semantic connections.

Query Processing Service. Queries are issued as free text or keywords and are formulated as term vectors. The node issuing the query forwards a message in the network with a TTL using both its social and semantic connections. The issued message is forwarded both to (i) nodes that have interests similar to the query and are contained in the *FI* of the query initiator (social search) and (ii) a small number of nodes contained in the *SI* of the query initiator, chosen as described below (semantic search). Initially, the message initiator compares the query against its interests and, if similar, the query is forwarded to all of its short-range links, i.e., the message is *broadcasted* to the node's neighborhood (*query explosion*). Otherwise, the query is forwarded to a small fixed number of nodes that have the highest similarity to the query (*fixed forwarding*). The combination of the two routing strategies is referred to in the literature as the *fireworks* technique [5, 7]. All the nodes receiving the query message reduce TTL by one and apply the same forwarding technique; the query message is not forwarded further in the network when TTL reaches zero. Additionally to query forwarding, every node receiving a query message compares it against the identified interests and, if similar, matches it against the locally stored content. Subsequently, pointers to the matching content are sent to the query initiator, who orders candidate answers by similarity to the issued query and presents the list to the user.

3 Demonstration Summary

In our demonstration, we will present the $\mathcal{DS}^4$ prototype system build upon Microsoft .NET Framework v4.0, using C#, and the Lucene v2.9.1.2 library. A user may utilise the node join service to connect to the social network and invoke interest creation to automatically cluster the content to be shared and identify user interests. Additionally, the user may also manage his/her own document collection in the local index store, and add, remove, or modify the content or the metadata (e.g., tags). Interest creation may be invoked by the user when a significant amount of content in its local store has changed, or when the user wants to add/remove an interest. Apart from sharing the content with the rest of the community, the user may use the query processing service to issue multi-keyword queries on the (meta-)data and discover new content. The content discovery process is automatic and returns (i) relevant results from the users' local store and (ii) content created by friends (social search) or nodes specialising on the query topic (semantic search). Finally, a user may refine/refresh its connections manually by invoking the rewiring procedure at any time. All actions are facilitated through a graphical user interface (Figure 1(b)).

The demonstrator will present a use case of the implemented prototype that will include the following steps: node initialisation and interest creation, addition/deletion/modification of node content, query execution and discovery of content from friend and non-friend nodes, and rewiring of node connections. The users of the demonstrator will test the system on a network of nodes that will share textual content. For more details on architectural, efficiency, and effectiveness issues the interested reader is referred to [6, 7] and the $\mathcal{DS}^4$ project website: `http://www.uop.gr/~praftop/ds4/`.

References

1. K. Hui, J. Lui, and D. Yau. Small-world Overlay P2P Networks: Construction, Management and Handling of Dynamic Flash Crowds. *Computer Networks*, 2006.
2. A. Loser, M. Wolpers, W. Siberski, and W. Nejdl. Semantic Overlay Clusters within Super-Peer Networks. In *DBISP2P*, 2003.
3. A. Loupasakis, N. Ntarmos, and P. Triantafillou. eXO: Decentralized Autonomous Scalable Social Networking. In *CIDR*, 2011.
4. R. Narendula, T. Papaioannou, and K. Aberer. My3: A highly-available P2P-based online social network. In *P2P*, 2011.
5. C. H. Ng, K. C. Sia, and C. H. Chang. Advanced Peer Clustering and Firework Query Model in the Peer-to-Peer Network. In *WWW*, 2002.
6. P. Raftopoulou and E. Petrakis. iCluster: a Self-Organising Overlay Network for P2P Information Retrieval. In *ECIR*, 2008.
7. P. Raftopoulou, E. Petrakis, and C. Tryfonopoulos. Rewiring strategies for semantic overlay networks. In *DPD*, 2009.
8. S. Voulgaris, M. van Steen, and K. Iwanicki. Proactive Gossip-based Management of Semantic Overlay Networks. *CCPE*, 19(17), 2007.