Towards Content-Based Publish/Subscribe for Distributed Social Networks

Christos Tryfonopoulos[1] Paraskevi Raftopoulou[1] Vinay Setty[2] Argiris Xiros[1]
[1]University of the Peloponnese, Greece
[2]Max-Planck Institute for Informatics, Germany
{trifon, praftop, axiros}@uop.gr vsetty@mpi-inf.mpg.de

ABSTRACT

Over the last few years a number of distributed social networks with data management capabilities have been introduced both by academia and industry. However, none of these efforts has so far focused on supporting content-based publish/subscribe functionality in a distributed social networking environment. In this work we present a social networking architecture in the literature that offers content-based pub/sub functionality -in addition to the usual social interaction and data management tasks- in distributed social networks, outline the associated distributed protocols, and identify interesting pub/sub problems (e.g., diversity, novelty, rate, relevance) in such scenarios. To the best of our knowledge, our system is the first of its kind to offer such an unique combination of features in a decentralised setting. Finally, we highlight the feasibility of the proposal by means of experimentation with real social networking data and the implementation of a prototype system.

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 poses a significant cognitive overload to users that have to cope with a never-ending stream of social updates that amounts to hundreds or even thousand notifications per day.

Idea, challenges, and approach. In this work, we present a *distributed social networking architecture* that allows users to *share, search for*, and *subscribe to* content in a fully decentralised way, while at the same time maintaining access control and ownership of their content. The content here can be textual such as status updates, blog posts, tweets, or tagged multimedia content such as photos and videos. Such a design is ideal for implementing scientific or enterprise social networks, where people are particularly reluctant to upload their data to a third party. Our work builds upon research results from the peer-to-peer paradigm, such as those utilizing unstructured, small-world, and semantic overlay networks [9, 10, 16]. Such decentralised approaches to social networking have also been adopted both by academia and industry by building a number of distributed social platforms [14, 3, 3, 15] and distributed social data management systems [6, 5, 4, 7, 11, 8]. Although all these approaches offer different types of distributed social networks that allow users to create communities, share content, and send messages, none of them focus on supporting *content-based publish/subscribe* (pub/sub) in the context of distributed social networks.

Our proposed solution offers fundamental social interactions, while emphasizing on *content-based search and pub/sub* functionality, *user autonomy*, and *data ownership*. In addition, we propose a distributed social networking system that supports *content-based pub/sub* allowing users to effectively cope with the effort and cognitive overload produced by social updates and enabling them to receive notifications on subscribed content from semantically similar sources of information. To do so we leverage on the concepts of *semantic friendship* [11] between users and *approximate pub/sub* techniques [18]. In our design, semantic friendship emerges from common user interests/profiles 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 this protocol is to create *groups/clusters of nodes* with similar interests. User queries and subscriptions can then be resolved by routing them towards friends and nodes specializing to the query topic. Moreover, in approximate pub/sub, publications are processed locally and nodes query or subscribe to only a few selected content sources (i.e., social or semantic friends) that are most likely to satisfy the user's information demand; in this way, we trade recall for message traffic and scalability.

Contributions. To this end, the contributions of this work

are threefold:

- We propose a social networking architecture that offers content-based pub/sub functionality, in addition to the usual social interaction and data management tasks typically supported in such scenarios. In this context, we also identify interesting extensions (e.g., rate, relevance, diversity, novelty) to the general problem of pub/sub in distributed social networks and outline possible solutions.
- We present the *protocols* and *services* that regulate node interactions, provide details on the distributed social/semantic pub/sub and search, and outline the notion of semantic friendship that plays a central role in our design.
- We show the effectiveness of the proposed pub/sub protocols by means of preliminary experimentation with real social networking data. We also demonstrate the applicability of the proposed architecture in a real-life setting through the deployment of a prototype system.

Application scenario. As an example of an application scenario let us consider Anna, a doctor whose main field of expertise is pathology. Mary is interested in connecting with fellow doctors in her hospital and following the work of prominent professionals in the area and is willing to share her own work and ideas with other doctors; this interaction would help enrich her circle of professional contacts and would provide a leverage for professional improvement. Currently, she would have to (i) use one of the centralised social networks, like LinkedIn or ResearchGate (aimed for professional or research use), to follow the work of other researchers, (ii) periodically resort to a number of different digital libraries like PubMed to locate interesting bibliography, and (iii) use a file hosting service, like Dropbox or Fileserve, to exchange patient datasets with her colleagues.

All these actions performed on different centralised systems would force Anna to create and manage a set of (non-interoperative) user profiles for each service, give away data ownership rights for sensitive clinical patient data just to be able to share them with peers, and would impose an extra burden on her by having to repeatedly search for interesting content related to her job. Clearly, Mary would benefit from accessing a Web 2.0-inspired solution that is able to provide social interactions with uni/multicast data dissemination and socially-/semantically-aware content based search and pub/sub functionality, while allowing her to maintain data ownership and access control in an *integrated service*. Such a system would be a valuable tool, beyond anything supported in current centralised social networks, that would allow Mary to save both time and effort while interacting with colleagues and peers.

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 *social friends* and *semantic friends*. Social friend links correspond to the social relationship aspect of the network, while semantic friend links are of two types: short-range links (i.e., links to nodes with similar interests) and long-range links (i.e., links to nodes with dissimilar interests to maintain connectivity of remote clusters). The reorganization (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.

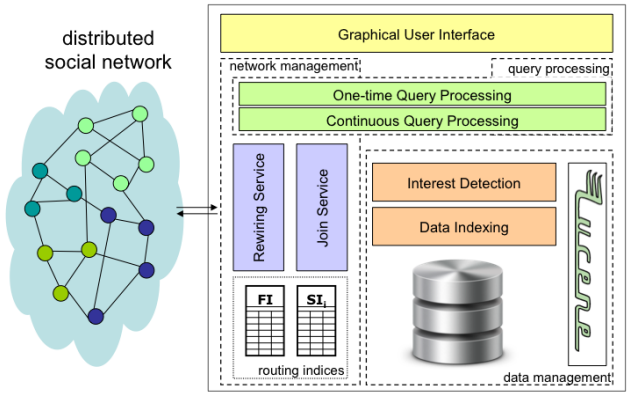

Figure 1: High-level view of node architecture.

The main idea behind our proposal is to let nodes that are semantically, thematically, and socially close self-organise, to facilitate the content search mechanism. Figure 1 shows a high-level view of a node and the different types of services implemented, which are described in detail below.

Join Service. When a user node connects to the 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 network.

Rewiring Service. The rewiring service is applied to reorganise 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.

Continuous Query Processing Service. Continuous queries are issued as free text or keywords under the Vector Space Model and are formulated as term vectors. The user node subscribing with a continuous 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 pub/sub) and (ii) a small number of nodes contained in the *SI* of the query initiator (semantic pub/sub) chosen as described below. Initially, the message initiator compares the continuous query against its interests and, if similar, the message is forwarded to all of its short-range links, i.e., the message is *broadcasted* to the node's neighborhood (*query explosion*). Otherwise, the message is forwarded to a small fixed number of nodes that have the highest similarity to the continuous query (*fixed forwarding*). The combination of the two routing strategies is referred to in the literature as the *fireworks* technique [9]. All the nodes receiving the message reduce TTL by one and apply the same forwarding technique; the message is not forwarded further in the network when TTL reaches zero. Additionally to forwarding, every node receiving a message compares the continuous query against the identified interests and, if similar, stores it in its local continuous query data structures to match it against future publications. Nodes will utilise these data structures at publication time to find quickly all continuous queries that match a publication. This can be done using an appropriate local filtering algorithm such as SQI [17].

Publications are kept locally at each content provider in the spirit of [18]. This lack of publication dissemination mechanism is a design decision that avoids document-granularity dissemination (e.g., as in [18]) and offers increased scalability by trading recall. Thus, only the nodes indexing a continuous query can notify the interested user, although other provider nodes may also publish relevant documents. When a node wants to publish a new document to the network, it matches it against its local continuous query database to decide which continuous queries match the document and thus, which user node should be notified. Then, the provider node delivers a notification for each continuous query by sending to the query initiator a pointer to the matching content; if the user is not online, then the responsible node stores the message and delivers it to the user upon reconnection.

One-Time Query Processing Service. A node issuing the one-time query forwards a message in the network following the mechanism described in the previous section. 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.

Implementation. The prototype system (based on $\mathcal{DS}^4$ [11]) was build upon Microsoft .NET Framework v4.5, using C#, the WPF and WCF libraries for the graphical user interface and the communication protocols respectively, the Lucene v3.0.3 library for the indexing and handling of all content management tasks, and the log4net library for logging and reporting. The full set of join/leave, rewiring, and one-time query processing protocols are implemented, while the implementation of the continuous query processing protocols is currently under way.

3. DIVERSIFIED CONTENT RETRIEVAL IN PUB/SUB

In previous works, the pub/sub paradigm has been used for delivering social notifications at a large scale [13]. In our pub/sub setting, humans are the main recipients of information and social notifications are known to result in cognitive overload for social networking users. For this reason, in Section 2, we have employed approximate pub/sub techniques that store the continuous query only in a few carefully selected sources, avoiding to deliver all matching social notifications to the subscribed users. However, in a social setup, reducing notification load is not enough since there is also the need to filter information so as to ensure that (i) the notifications are highly relevant to users' information need, (ii) notification delivery rate is within an acceptable threshold, and (iii) notifications are diverse and novel in content.

While there are efforts to address $top{-}k$ filtering in pub/sub with diversity constraints [2], they are not designed for distributed environments and relevance is typically based on pre-assigned preferences to queries. Moreover, in [12] cost-effective techniques to select a subset of notifications so as to restrict the notification delivery rate to the users are proposed, but they ignore diversification. In this work, we highlight the challenges and propose early solutions in designing a distributed architecture to filter events that are highly relevant to the users, diverse in content and reduce the cognitive overload by restricting the notification delivery rate.

Inspired by [1], we introduce the concept of *diversity by proportionality* for continuous queries that are processed incrementally in real-time in a distributed setup. In diversity by proportionality, we are given query q with a ranked list of documents $R = \{d_1, d_2, ..., d_m\}$, that are relevant for one or more aspects $A = \{a_1, a_2, ..., a_n\}$ with popularity $P = \{p(a_1), p(a_2), ..p(a_n)\}$. The goal is to select S, a subset of R, such that different aspects are represented proportionally to their popularity and $|S| = k$. The aspects in A are derived from the social interests of the users such as their friends list, the groups joined, or geographic locations interested in. For example, assume that Anna is interested only in top-10 notifications ($|S| = k = 10$) for the continuous query "Nobel Prize for Medicine" and has colleagues in Greece, Norway, and Germany (three geographical aspects) who have published relevant documents. According to diversity by proportionality, if 60% of the published documents are from Greece, 30% from Germany, and 10% from Norway, then the result set S should contain the six most relevant documents from Greece, three from Germany and one from Norway. Maintaining the proportionality while ensuring high relevance of the documents is a challenge; for example, it may be possible that all 30% of documents from Germany are significantly lower in relevance compared to documents from Greece and Norway. In order to ensure a balance between proportionality and overall quality

of the results, an adaptation of Sainte-Laguë method, typically used for allocating seats in the parliament could be utilised to address this challenge [1].

Implementing diversity by proportionality for social networks in a decentralised setup has several challenges:

- First and foremost, we need to identify different aspects for the continuous queries issued by the users. We propose to derive the aspects based on the social interests of users and the tags and categories from existing relevant documents for the continuous queries of users.
- Second, we need to maintain global statistics for different aspects. Since we are interested in continuous queries which are known beforehand, we can exploit the similarity in continuous queries and user interests for maintaining the popularity and relevant documents of different aspects. The maintenance of aspects can be distributed across participating nodes by electing a rendezvous node for each aspect or through the gossiping protocol adopted for rewiring.
- Third, when a new document is published we need to identify the relevant aspects for the document and update the popularity and relevant document lists for them. This can be performed locally at the publishing node by analyzing the locally stored continuous queries. Then an update message needs to be issued to all the rendezvous nodes of the relevant aspects.
- Finally, if the popularity of an aspect changes, then we need to ensure that changes to the result set S of all the affected continuous queries are forwarded to the users by re-applying Sainte-Laguë method.

4. EXPERIMENTAL EVALUATION

Our dataset comprised of about 700K users, 10M tags, and 21M bookmarks belonging to 170K categories obtained from a home crawl of the Delicious social bookmarking site. In our setup, bookmarks were resources and the corresponding tags were the resource descriptions; the set of resource descriptions in a node was used to derive node's interests. The continuous queries employed in the evaluation were strong representatives of bookmark categories. We submitted 10 different continuous queries per node, and averaged the result for 10 different network topologies, while resource publication order was randomized on a per node basis.

Figure 2 illustrates the retrieval effectiveness of the network as a function of time when continuous queries are stored in different types of friends. At time zero, the values for recall are low for all different strategies as no semantic grouping is yet initiated, while in the following timepoints nodes begin to organize into semantic groups resulting in different recall effect depending on the strategy used. Storing the continuous queries only to random or similar social friends does not improve recall since the indexing is static and does not follow the node interests (although the importance of semantic information is demonstrated if similar, instead of random, social friends are selected). Contrary, when the query initiator periodically (i.e., when reorganizing its semantic friends) uses semantic information to forward the continuous query to more relevant nodes recall improves. We are currently exploring different parameters for deciding when to move a continuous query including database selection and publication prediction techniques [18].

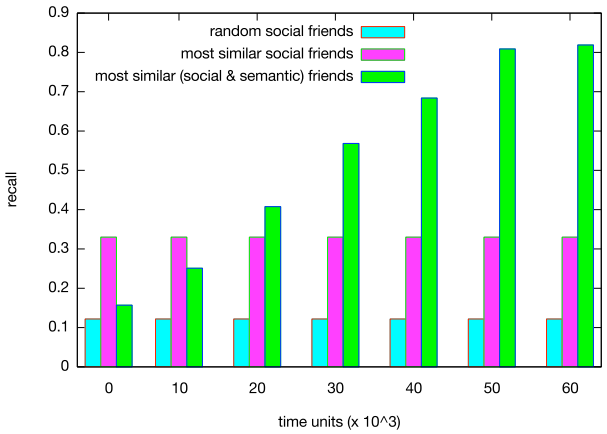

Figure 2: Recall when continuous queries are stored in different types of friends.

5. REFERENCES

[1] V. Dang and W. B. Croft. Diversity by proportionality: an election-based approach to search result diversification. In *SIGIR*, 2012.

[2] M. Drosou, K. Stefanidis, and E. Pitoura. Preference-aware publish/subscribe delivery with diversity. In *DEBS*, 2009.

[3] K. Graffi, C. Gross, P. Mukherjee, A. Kovacevic, and R. Steinmetz. LifeSocial.KOM: A P2P-Based Platform for Secure Online Social Networks. In *P2P*, 2010.

[4] L. Han, M. Punceva, B. Nath, S. Muthukrishnan, and L. Iftode. SocialCDN: Caching techniques for distributed social networks. In *P2P*, 2012.

[5] G. Liu, H. Shen, and L. Ward. An efficient and trustworthy P2P and social network integrated file sharing system. In *P2P*, 2012.

[6] A. Loupasakis, N. Ntarmos, and P. Triantafillou. eXO: Decentralized Autonomous Scalable Social Networking. In *CIDR*, 2011.

[7] G. Mega, A. Montresor, and G. Picco. Efficient dissemination in decentralized social networks. In *P2P*, 2011.

[8] R. Narendula, T. Papaioannou, and K. Aberer. My3: A highly-available P2P-based online social network. In *P2P*, 2011.

[9] 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.

[10] P. Raftopoulou and E. Petrakis. iCluster: a Self-Organising Overlay Network for P2P Information Retrieval. In *ECIR*, 2008.

[11] P. Raftopoulou, C. Tryfonopoulos, E. Petrakis, and N. Zevlis. DS4: Introducing Semantic Friendship in Distributed Social Networks. In *CoopIS*, 2013.

[12] V. Setty, G. Kreitz, G. Urdaneta, R. Vitenberg, and M. van Steen. Maximizing the number of satisfied subscribers in pub/sub systems under capacity constraints. In *INFOCOM*, 2014.

[13] V. Setty, G. Kreitz, R. Vitenberg, M. van Steen, G. Urdaneta, and S. Gimåker. The Hidden Pub/Sub of Spotify: (industry article). In *DEBS*, 2013.

[14] R. Sharma and A. Datta. SuperNova: Super-peers based architecture for decentralized online social networks. In *COMSNETS*, 2012.

[15] P. Stuedi, I. Mohomed, M. Balakrishnan, Z. Mao, V. Ramasubramanian, D. Terry, and T. Wobber. Contrail: Enabling Decentralized Social Networks on Smartphones. In *Middleware*, 2011.

[16] S. Voulgaris, M. van Steen, and K. Iwanicki. Proactive Gossip-based Management of Semantic Overlay Networks. *CCPE*, 19(17), 2007.

[17] T. Yan and H. Garcia-Molina. The SIFT Information Dissemination System. In *TODS*, 1999.

[18] C. Zimmer, C. Tryfonopoulos, K. Berberich, M. Koubarakis, and G. Weikum. Approximate Information Filtering in Peer-to-Peer Networks. In *WISE*, 2008.