SQTime: Time-enhanced Social Search Querying

Panagiotis Lionakis[1], Kostas Stefanidis[2], and Georgia Koloniari[3]

[1] Department of Computer Science, University of Crete, Heraklion, Greece
lionakis@csd.uoc.gr
[2] Institute of Computer Science, FORTH, Heraklion, Greece
kstef@ics.forth.gr
[3] Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece
gkoloniari@uom.edu.gr

Abstract. In this paper, we present SQTime, a system for social search queries that exploit temporal information available in social networks. Specifically, SQTime introduces different types of queries aiming at satisfying information needs from different perspectives. SQTime is built upon a social graph and query model both augmented with time, and develops methods for query processing and time-dependent ranking.

1 Introduction

Due to the increasing popularity of social networks and the vast amount of information they contain, recently, there have been many efforts in enhancing Web search based on social data. An important dimension of social networks is their dynamic nature. New information is added through user activities and updates, representing changes in their interests. To deal with this temporal aspect, we use an annotated graph model that incorporates time, similarly to [2], for representing the social network, and propose a new time-enhanced query model.

Unlike previous approaches that focus on graph evolution [4, 5], our query model deals with social search and introduces two query types: *user-centric* and *system-centric* queries. User-centric queries offer a personalized search feature by exploiting a user's social relationships. System-centric queries provide a global search feature with applications in online-shopping and target-advertising, so as to select the best target group for a new product or the best products to promote to a given user. We allow both the explicit and implicit use of time [3]. To allow queries to express time explicitly, we extend user- and system-centric queries to time-dependent queries that include temporal hard constraints. Furthermore, we enhance query results with an implicit use of time by providing a time-dependent ranking, so that more recent results are returned first.

SQTime incorporates a framework for processing queries based on the time-enhanced query model, we first introduced in [6], and offers the user the opportunity to explore both the explicit and implicit use of temporal information and its impact when querying the social graph.

2 The SQTime Framework

We model a social network as an undirected graph, $G = (V, E)$. Nodes in V correspond to the entities that belong to the social network, while edges in E

capture the relationships between the entities in V. We discern between two types of entities: (i) users U, that consist of the social network participants, and (ii) objects O, that include all other entities in the social network, e.g., applications, events and photos. Edges between users capture the friendships between them, while edges between users and objects declare that users consume the objects. We consider extending the typical graph model with temporal information towards making social search time-dependent. This way, an element, node or edge, of G is *valid* for the time period for which the corresponding element of the social network it represents is also valid. To incorporate times into the graph, each element is annotated with a label $l = (t_{start}, t_{end})$ that determines the time interval for which the element is valid.

Our goal is to support queries for the graph structure that also exploit the time dimension of the graph elements. In user-centric queries, a user is interested in retrieving information about other users or objects that satisfy specific predicates and are connected to the user directly or through their friends. We consider two general query categories. The first category gives priority to the friends of the user, while the second one to the objects to be consumed. This model is enhanced with time by including separate constraints for the validity in specific time intervals, for the elements that are included in a query. For example, "*get my friends, valid in 2013, that have been attended sports events*" retrieves my friends, valid in 2013, that have some connection to events that satisfy the given predicate, even if the events themselves are valid at another time period.

In general, given the graph $G = (V, E)$, for a user-centric query for friends $Q(u_i, P)$, where P is a set of query predicates, processing proceeds as follows:

Step 1: Retrieve the user nodes $u_j \in U$, say U', st. $\exists(u_i, u_j) \in E$.
Step 2: Retrieve the object nodes $o_l \in O$, say O', st. o_l satisfies all predicates in P.
Step 3: Remove all nodes u_j from U', st., for at least one node $o_l \in O'$, $\nexists(u_j, o_l) \in E$.
Step 4: The remaining nodes form the result $res(Q)$.

For the time-dependent query (Q, T), where $T = [s, d)$ is a time constraint, T concerns the nodes in Step 2, and is treated as another predicate in P, or the result nodes in Step 4, and therefore, introduces the following filtering steps:

Step 5: From the nodes in $res(Q)$ remove all nodes v_j, st. $l(v_j).t_{start} < s$ or $l(v_j).t_{end} \geq d$.
Step 6: The remaining nodes form the result $res(Q, T)$.

Similarly, for system-centric queries, the system requires locating: (i) users connected to particular objects, or (ii) objects connected with particular users, and augment them with time, to retrieve only valid information (details in [6]).

SQTime ranks results according to the freshness of the connections between users and objects. Freshness is recorded in our graph model on the edges' labels. In general, our motivation is based on the fact that recently added edges better reflect the current trends and thus, they could contribute in ranking the results. For example, assume a user-centric query for friends $Q(u_i, P)$. $ranked\_res(Q)$ of Q is a ranked list of the users in $res(Q)$; ranking is achieved with respect to the

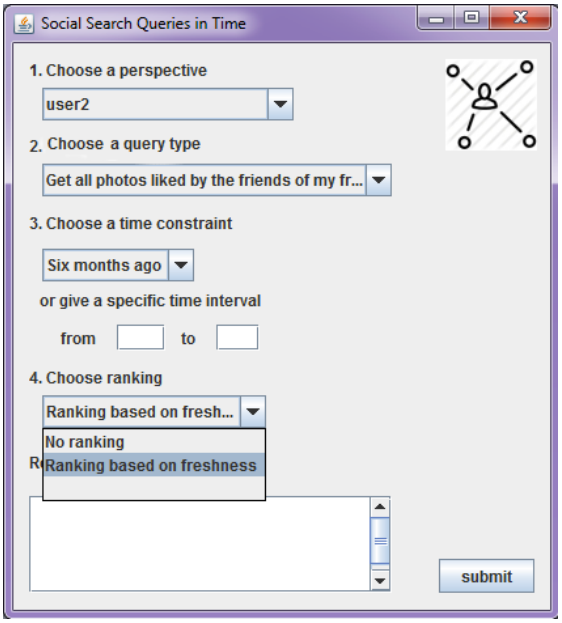

Fig. 1. SQTime querying interface.

labels of the edges that connect the users in $res(Q)$ with the objects satisfying P and, in particular, with respect to the time the connections were established. Abstractly, for ranking, we first assign to each result element a score equal to the maximum t_{start} value among the values of the labels of its connections to the nodes that satisfy the predicates of the query, and then sort the resulting elements according to that score.

3 Demonstration

This demo of SQTime aims at allowing users to explore the different types of social search queries in our model, and the implications of enhancing queries with time both implicitly and explicitly. SQTime is built on top of a social graph we constructed with a data set from [1], that includes anonymized information about the evolution of the Flickr social network and in particular, user-to-user links, photos and favorite markings of photos by users. The photos shown in the demo are randomly assigned to given photo ids to preserve the relationships among them and the users. The demo offers a graphical interface for running time-dependent queries and visualizing the returned results sets.

Querying Interface. The SQTime querying interface, illustrated in Fig. 1, provides lists of available options for each of the building blocks of a query, which users can combine in four easy steps to compose their own queries. All options are expressed in natural language not requiring from the user to have an understanding of the underlying social graph structure. Firstly, the user selects a

query perspective. For user-centric queries, the demo offers a selection of different users from our dataset with varied characteristics. For instance, it includes users with a large number of friends in the social graph, as well as users with fewer friends. When the system perspective is selected, as the only available attribute for the users is their anonymized id, the user is called to specify a selection on the ids to determine a group of users involved in the query. The second part of the query determines whether the query concerns users or objects, i.e., Flickr photos. Options such as "*get all photos liked by my friends*", for user-centric queries, and "*get all photos liked by the group of users specified in step 1*", for system-centric queries, are available. Beyond queries in the direct neighborhood of a user, SQTime includes queries at distance two. Thus, one may also select to "*get all photos the friends of my friends liked*".

The first two steps guide the user to forming her basic social query. The next two steps guides the user to augment her query with time. Step 3 supports the explicit use of time allowing users to add a time constraint in their queries. One can either select a predefined constraint that concern the most recent past, as such queries are more popular, or specify her own time period for the constraint. For example, if we select the time constraint "*six months ago*", "*get all my friends*" is transformed into "*get all my friends that were valid six months ago*". The last parameter SQTime enables us to configure, is ranking. Two options are provided, "*no ranking*", where all results are returned in random order, and "*ranking based on freshness*". This option enables the user to see the implicit use of time, as the results are returned ranked based on their freshness.

Results Presentation. The result presentation interface visualizes query results in a way that is intuitive and clearly illustrates their temporal relationships. For a query for objects, the qualifying photos are displayed to the user and listed in a results' text box. For clarity, the demo limits the returned results to 10. If no ranking is used 10 random photos from the result set are returned. When ranking based on freshness is selected, the 10 fresher results are displayed. As shown in Fig. 2 (left), SQTime displays results as photos of different size, where photos of bigger size represent the higher ranked objects, i.e., the fresher ones. For queries for users, a graph is used to display the qualifying users and their relationships. For instance, Fig. 2 (right) illustrates the results for a user-centric query for friends. The user in the center of the displayed graph is the user whose perspective we have selected, while the user ids around him represented the returned friends. When ranking based on freshness is used, shades of green are used to represented the ids of each user, with bolder colored ids reflecting the fresher results. Note that in this case all qualifying users are returned.

References

1. M. Cha, A. Mislove, and K. P. Gummadi. A Measurement-driven Analysis of Information Propagation in the Flickr Social Network. In *WWW*, 2009.
2. C. Gutierrez, C. A. Hurtado, and A. A. Vaisman. Introducing time into RDF. *IEEE Trans. Knowl. Data Eng.*, pages 207–218, 2007.
3. H. Joho, A. Jatowt, and R. Blanco. A survey of temporal web search experience. In *WWW*, 2013.

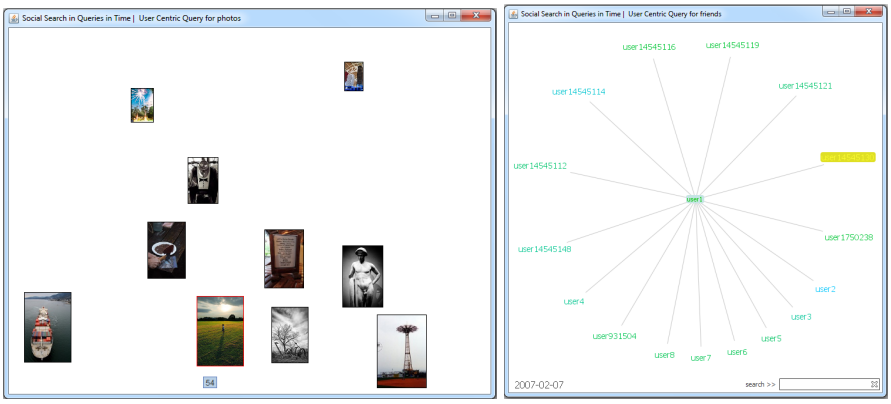

Fig. 2. SQTime results presentation.

4. U. Khurana and A. Deshpande. Efficient snapshot retrieval over historical graph data. In *ICDE*, 2013.
5. G. Koloniari, D. Souravlias, and E. Pitoura. On graph deltas for historical queries. In *WOSS*, 2012.
6. G. Koloniari and K. Stefanidis. Social search queries in time. In *PersDB*, 2013.