Querying Socio-spatial Networks on the World-Wide Web

Yerach Doytsher
Technion – Israel Institute of Technology
doytsher@technion.ac.il

Ben Galon*
Technion – Israel Institute of Technology
bgalon@technion.ac.il

Yaron Kanza*
Technion – Israel Institute of Technology
kanza@cs.technion.ac.il

ABSTRACT

Many devices, such as cellular smartphones and GPS-based navigation systems, allow users to record their location history. The location history data can be analyzed to generate *life patterns*—patterns that associate people to places they frequently visit. Accordingly, an SSN is a graph that consists of (1) a social network, (2) a spatial network, and (3) life patterns that connect the users of the social network to locations, *i.e.*, to geographical entities in the spatial network. In this paper we present a system that stores SNN in a graph-based database management system and provides a novel query language, namely SSNQL, for querying the integrated data. The system includes a Web-based graphical user interface that allows presenting the social network, presenting the spatial network and posing SSNQL queries over the integrated data. The user interface also depicts the structure of queries for the purpose of debugging and optimization. Our demonstration presents the management of the integrated data as an SSN and it illustrates the query evaluation process in SSNQL.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Web-based services; H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Design, Languages, Human Factors

Keywords

Socio-spatial networks, Social networks, Geographic information systems, GIS

1. INTRODUCTION

GPS-enabled devices that can record the geographic location of people have become prevalent. Such devices generate traces that provide the location history of individuals. The location history can be analyzed to produce *life patterns*, that is, to generate formulas that indicate frequently-visited locations and the associated *time patterns* of these visits. The life patterns of a person provide a probabilistic representation of the stay points and visited places of this person. For example, the time patterns of some person, Marge, may indicate that she stays at 742 Evergreen, Springfield almost every day from 2pm until 9am and that she visits the grocery store on Evergreen street almost every Friday between 10am and 11am. Life patterns represent such periodical, or nearly periodical, behaviors. A *confidence* value is assigned to life patterns to indicate the level of regularity of the pattern. We consider confidence values as probabilistic measures, *e.g.,* if a confidence of 0.8 is assigned to the pattern of Marge's visits at the grocery store, then we assume there is a probability of 0.8 that Marge visits the grocery store on any arbitrary Friday between 10am and 11am.

Extracting life patters of users from location loggings has been studied by Ye et al. [6]. Zheng et al. [7] showed how to mine a GPS log and find locations that are point-of-interest. They also studied the problem of finding similarity among users based on points where users stayed for a relatively long time, according to their GPS logs [5]. We assume that the techniques in these papers are being employed to generate life patterns.

Life patterns can be used to connect users of a *social network* to locations on a *spatial network*. In our model, a social network is a graph whose nodes represent users and the edges represent relationships between users. A spatial network is a graph whose nodes represent spatial entities and the edges mainly represent adjacency. A socio-spatial network (SSN) is essentially a graph that is constructed by combining a social network and a spatial network, using life-pattern edges. That is, we consider the life patterns as edges in which one node is a user and the other node is a geographical entity that is being visited periodically by the user. This provides a synergy between social services and GIS tools, and it can be used for analyzing location-based social events (see [4]).

One of the difficulties in querying an SSN is that queries should cope with the heterogeneity of the data. Queries should not return an uncontrolled mixtures of spatial entities and users, however, queries should be answered while taking into account both the spatial data and the social data jointly. Hence, we have developed SSNQL—a query language for socio-spatial networks. In SSNQL, users can pose queries that combine social information with spatial information. For example, in SSNQL one can formulate a query that finds friends of Marge who frequently buy at the same grocery store as she does.

*The work of these authors was partially supported by The Israeli Ministry of Science and Technology (Grant 3-6472) and by a grant from the Umbrella Cooperation (SUD2).

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

SSNQL has first been formally introduced in [1], and it was extended in [2] to handle travel routes, however, it was prior to the development and implementation of a complete data management system. In this paper we present a full-fledged system that manages socio-spatial networks and supports the evaluation of SSNQL queries over an SSN. The system stores the SSN in a graph-based database management system. It provides a Web-based user interface to facilitate the formulation of queries and it presents the results of queries in a way that makes it easy to browse and analyze these results.

2. SOCIO-SPATIAL NETWORK

A *socio-spatial network* (SSN) is essentially a graph, constructed by combining a social network and a spatial network, using life-pattern edges. In this paper we only provide a general description of the model.

Social Network. A social network is a graph in which the nodes represent users and the edges represent relations, of some type, among users, *e.g.,* friendship relations, kinship, business relations, academic relations, and so on. Users may have attributes such as name, hobbies, *etc.*

Spatial Network. A spatial network is a graph whose nodes represent spatial entities and the edges represent relationships between entities. The entities form a hierarchy where buildings are below neighborhoods and neighborhoods are below cities. Thus, there are two types of spatial relationships—adjacency and hierarchy. There is an *adjacency edge* between any two adjacent entities that are at the same level of the hierarchy. There is a hierarchical edge from an entity to an entity higher in the hierarchy when the first is contained in the second, *e.g.,* a house will have a hierarchy edge to the neighborhood that contains it.

Life Patterns. *Life patterns* represent regular visits of users in geographical locations. Thus, we use life patterns to associate between users and spatial entities. In general, a life pattern is a 4-tuple that comprises a user, a geographical entity, a *time pattern* and a *confidence value.* The life pattern specifies that the user visits the geographical entity at the times that are defined by the time pattern.

A time pattern can have one of the following two forms: (1) a daily pattern `(D; ;` h_1`;` h_2`)` to define events that occur every day between the hours h_1 and h_2; and (2) a weekly pattern `(W;` L `;` h_1`;` h_2`)`, where $L \subseteq \{1, 2, 3, 4, 5, 6, 7\}$ is a list of days during the week, to define events that occur on the days that appear in L, between the hours h_1 and h_2.

The confidence value of a life pattern determines the level of regularity of the pattern. When the confidence value is high the regularity is high, and when it is low, the regularity is low. Formally, a confidence value $0 < c \leq 1$ means that in $c \cdot 100$ percent of the times defined by the time pattern, the user stays at the geographical entity. For example, a life pattern $(u, l, [W; 2, 3, 4, 5, 6; 10, 12]; 1)$ specifies that the user u stays at location l in every workday (Monday to Friday) between the hours 10 and 12. The life pattern $(u, l, [W; 2, 3, 4, 5, 6; 10, 12]; 0.8)$ specifies that u stays at l only in 80 percent of the times defined by the time pattern.

We say that a life pattern P_1 *complies* with a pair of time pattern T_2 and confidence value c if the times defined by the time pattern T_2 contain those defined by the time pattern of P_1, and the confidence of P_1 is greater than or equal to c. In other words, T_2 and c are more general than the time pattern and the confidence of P_1.

Socio-spatial Network. A socio-spatial network (SSN) is a graph that comprises a social network, a spatial network and life-pattern edges that connect nodes of the social network to nodes of the spatial network. We refer to the social network and to the spatial network as the *subnetworks* of the SSN. We consider a set N of nodes of an SSN as *homogeneous* if all the nodes are from the same subnetwork, *i.e.,* either all the nodes of N are spatial entities or all the nodes are users.

3. SSNQL

SSNQL is a query language for socio-spatial networks. The language is an algebra that consists of seven operators, under set semantics. The language is formally presented in [1], and here we merely provide a brief overview of the operators. There are four unary operators. Each unary operator receives a homogeneous set of nodes and returns a homogeneous set of nodes.

- `select`(N, c) receives a set of nodes N and a condition c, and returns the nodes of N that satisfy c.
- `extend`(N, k) receives a set of nodes N and a number k and it returns all the nodes that are reachable from the nodes of N, in the subnetwork of N, by a path whose length is bounded by k.
- `bridge`(N, T, c) receives a set of nodes N of some subnetwork, a time pattern T and a confidence value c. It returns the nodes (of the other subnetwork) that are connected to the nodes of N by life pattern edges that comply with T and c.
- `mbridge`(N, T, c, p) returns the nodes (of the other subnetwork) that are connected to at least p percent of the nodes of N by life pattern edges that comply with T and c.

There are three binary operators. Each binary operator receives two homogeneous sets, from the same subnetwork, and it returns an homogeneous set of nodes of that network. The operators are `union`, `intersect` and `difference` and their semantics is as in set theory.

4. SYSTEM DESIGN

The system is implemented using a three-tier architecture comprising a client, a middleware server and a database. The middleware server serves as an SSN data-management tool. The architecture is depicted in Figure 1.

Client side. The system provides a Web-based graphical user interface for constructing and editing queries, viewing the structure of queries, executing queries and presenting the results. The client side is implemented using Javascript and HTML, to be run on Web browsers.

A general overview of the user interface is depicted in Figure 2. On the upper-left part of the GUI, a map of the queried area is presented using the Google Maps API. For queries whose results are geographical entities, the results are portrayed on this map. On the right side of the map, the system presents the results of query evaluation. The results are presented as a list—either as a list of spatial entities

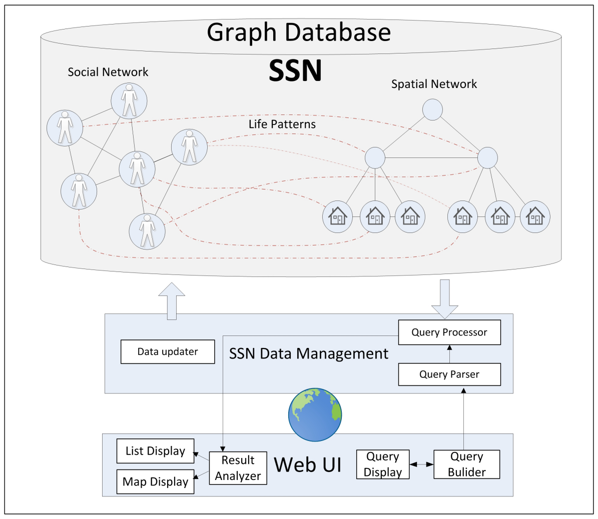


Figure 1: The architecture of the system.

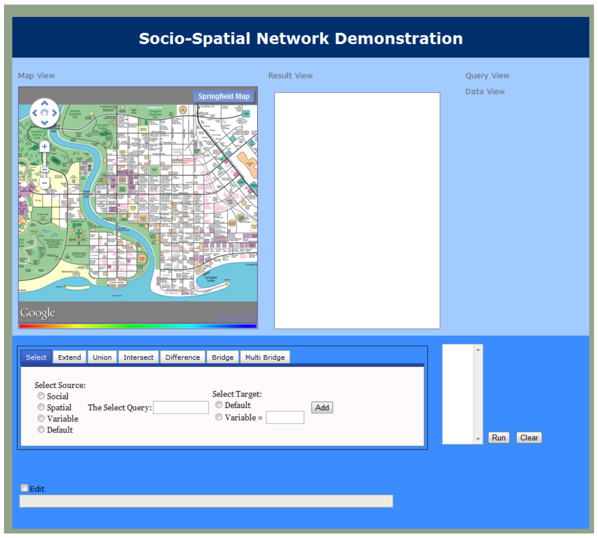


Figure 2: The user interface. The upper part contains a map of the region, and areas to present the query result and the query expression tree. The lower part is a graphical user interface for constructing, editing and running queries.

or as a list of users of the social network. The items of the list are sorted according to relevance, and the text size of the presented items is proportional to their relevancy. On the upper right part of the screen, the GUI depicts the structure of the query—the root is the entire query expression and the other nodes represent subexpressions. The bottom part of the screen contains a form-based GUI for formulating and editing queries. This GUI allows creating SSNQL expressions. To that end, it lets the user select the operators of the expression iteratively. In order to generate complex expressions, variables that represent subexpressions can be defined, and the operators of the language can be applied either directly—on the spatial network, on the social network or on subexpressions—or indirectly on variables that represent previously defined expressions.

Middleware Server. The middle tier is a server that manages requests from clients and interprets these requests to operations on the database. It has two main functionalities: it processes queries and it serves as a proxy for data management operations. This server is implemented in Java.

Query processing starts by parsing the query and generating an expression tree. The expression tree is depicted by the GUI (such tree is presented on the right side of Figure 4). The query evaluation is a parallel process where any two twigs that originate from the same node are being processes independently, in parallel. When evaluating a query, the system initiates a thread for each node of the tree (*i.e.*, for each subexpression). For leaves, the evaluation starts immediately. For an inner node, the thread is blocked until the threads of its children nodes generate data it can process. Some operations, such as `extend`, can start the evaluation as soon as they receive nodes. Other operations, such as `intersect`, need to wait until the processing of the children nodes is complete, before starting.

Database. We use Neo4j[1]—a graph-based database management system—to store the database. Neo4j is an open source transactional database server, implemented in Java.

The SSN is stored as a graph. One part of the graph is the spatial network and the other part of the graph is the social network. Life-pattern edges connect nodes of the social network to nodes of the spatial network. The labels on these edges are being used for representing the time patterns and the confidence values. We use indexes on nodes to efficiently retrieve nodes that satisfy given constraints, *e.g.*, when evaluating `select` operations.

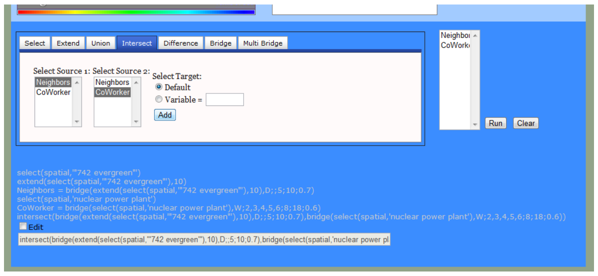
5. DEMONSTRATION

The demo presents the SSNQL query language and the system that was described in the previous section. In the demonstration, we illustrate the formulation of queries using the graphical user interface, the presentation of the query structure as an expression tree and the execution of queries. We show how query results can be viewed and analyzed using the system.

To illustrate the evaluation of queries, we generated data based on the characters of the television series "The Simpsons"[2]. We constructed a social network of approximately 100 people from the characters of this series. We also constructed a spatial network from the geographical description of the fictional city Springfield, where the series takes place. Life patterns were generated manually using knowledge about the characters. For example, appropriate patterns connect the members of the Simpson family (Homer, Marge, Bart, Lisa and Maggie) to the house of the family. There are patterns that connect people to their work place, *e.g.,* Homer Simpson, Carl Carlson, Lenny Leonard, Charles Montgomery Burns and Waylon Smithers are all connected to the power plant. There are additional patterns such as associating Homer Simpsons and Barney Gumble to Moe's Tavern, *etc.*

[1]`http://neo4j.org/`

[2]`http://www.thesimpsons.com/`



```
select(spatial,'"742 evergreen"')
extend(select(spatial,'"742 evergreen"'),10)
Neighbors = bridge(extend(select(spatial,'"742 evergreen"'),10),D;;5;10;0.7)
select(spatial,'nuclear power plant')
CoWorker = bridge(select(spatial,'nuclear power plant'),W;2,3,4,5,6;8;18;0.6)
intersect(bridge(extend(select(spatial,'"742 evergreen"'),10),D;;5;10;0.7),bridge(select(spatial,'nuclear power plant'),W;2,3,4,5,6;8;18;0.6))
intersect(bridge(extend(select(spatial,'"742 evergreen"'),10),D;;5;10;0.7),bridge(select(spatial,'nuclear power pl
```

Figure 3: The graphical user interface for constructing queries.

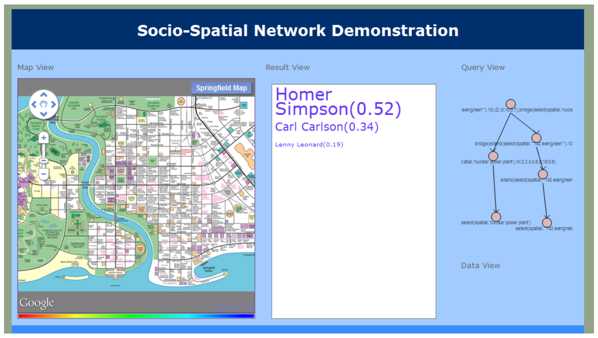


Figure 4: The presentation of the result for a query that returns users of the social network.

We show how to formulate queries using the system.[3] As an example, suppose that Homer Simpson wants to find coworkers who live near him to arrange a carpooling. He may formulate the following query (to simplify the explanation, we divided the query to subexpressions and assigned them to variables, yet the query can be written as a single expression by unfolding—replacing the variable names by the expressions they represent).

Suppose Homer Simpson lives in 742 Evergreen. The following query will find his home.
`Simpsons_Home = select(spatial, '742 Evergreen')`
Next, we find buildings that are at most 10 houses far from Homer's house.
`Neighborhood = extend(Simpsons_Home, 10)`
Using `bridge`, we find the residents of these houses. (People who stay in the house every day, from 5pm until 10am, with confidence of 0.7)
`Neighbors = bridge(Neighborhood, D; ; 17; 10; 0.7)`
In parallel, we find the Power Plant, where Homer works.
`PowerPlant = select(spatial, 'Nuclear Power Plant')`
Homer's coworkers are people who arrive at the power plant five days a week (Monday to Friday, represented as 2,3,4,5,6) and stay there from 8 until 18, with confidence 0.6.
`CoWorkers = bridge(PowerPlant,W;2,3,4,5,6;8;18;0.6)`
Finally, we intersect the neighbors and the coworkers to find potential people for the carpooling.
`Answer = intersect(Neighbors, CoWorkers)`
Figure 3 illustrates the formulation of this query in the system. Figure 4 illustrates the result when evaluating the query over the dataset of the demo.

The answer to a query can either be geographical entities or a list of people from the social network. Figure 5 illustrates an answer to a query that returns a list of people. The results can be examined by zooming in and zooming out, and colors are being used to indicate the relevancy level of entities—highly relevant entities are colored by red, and the color changes from red to blue as relevancy decreases (see the bar at the bottom of the map). By clicking the nodes of the expression tree of a query, the results of the subexpressions are presented. This is used for analyzing the results and it assists in query optimization and in debugging.

[3]A video that presents the system can be viewed online at `http://www.youtube.com/watch?v=WQFQ_ZFAZWs`.

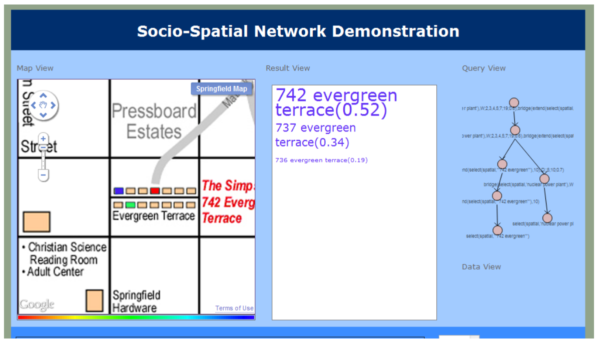


Figure 5: The presentation of the result of a query that returns geographical entities.

6. REFERENCES

[1] Y. Doytsher, B. Galon, and Y. Kanza. Querying geo-social data by bridging spatial networks and social networks. In *Proc. of the ACM SIGSPATIAL Int. Workshop on Location Based Social Networks*, pages 39–46, San Jose (CA, USA), 2010.

[2] Y. Doytsher, B. Galon, and Y. Kanza. Storing routes in socio-spatial networks and supporting social-based route recommendation. In *Proc. of the ACM SIGSPATIAL Int. Workshop on Location-Based Social Networks*, Chicago, (Il, USA), 2011.

[3] Q. Huang and Y. Liu. Analysis of a location-based social network. In *Proc. of the Int. Conf. on Computational Science and Engineering*, pages 263–270, Washington, DC, USA, 2009.

[4] Q. Huang and Y. Liu. On geo-social network services. In *Proc. of the 17th IEEE Int. Conf. on Geoinformatics*, Fairfax (VA, USA), 2009.

[5] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W. Ma. Mining user similarity based on location history. In *Proc. of the 16th ACM Int. Conf. on Advances in Geographic Information Systems*, 2008.

[6] Y. Ye, Y. Zheng, Y. Chen, J. Feng, and X. Xie. Mining individual life pattern based on location history. In *Proc. of the 10th IEEE Int. Conf. on Mobile Data Management*, pages 1–10, Taipei, Taiwan, 2009.

[7] Y. Zheng, L. Zhang, X. Xie, and W. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proc. of the 18th ACM Int. Conf. on World Wide Web*, pages 791–800, 2009.