Explaining Social Relationships

Tiago C. Marques[1], Vasco Furtado[1], Deborah L. McGuinness[2]

[1]Universidade de Fortaleza
Washington Soares
Fortaleza-CE, Brasil
[2]Rensselaer Polytechnic Institute
Troy, NY, USA
tiago.marques.mail@gmail.com, vasco@unifor.br, dlm@cs.rpi.edu

Abstract

Our work is aimed at explaining relationships between entities, and in particular, between persons. We note that there is an increasing amount of social network data on the web. This data may be leveraged to provide information about relationships. In the mobile computing ecosystem, with its many mobile devices and software applications, explanations of relationships between potential communicating entities may be critical to managing interconnections. The *SocialXplain* project was initiated to investigate methods of generating explanations of relationships leveraging distributed social networks. We explore social network navigation aimed at finding and describing links between a source and a target person. Once relationships are determined, *SocialXplain* also needs to define and implement strategies for generating abstract relationship explanations.

Introduction

Relationships play an important role in applications involving collaboration. In many situations, collaborators may desire some explanation of a relationship between themselves and a potential agent before they will consider accepting a message, phone call, calendar request, or other interaction. We are investigating issues involved with explaining relationships. In this work, we focus on explanations of relationships between entities, and in particular, between persons. We note that there is an increasing amount of social network data on the web. This data may be leveraged to provide information about relationships. In the mobile computing ecosystem, with its many mobile devices and software applications, explanations of relationships between potential communicating entities may be critical to managing interconnections. The *SocialXplain* project was initiated to investigate methods for generating explanations of relationships leveraging distributed social networks. We explore social network navigation aimed at finding and describing links between a source and a target person. Once relationships are determined, *SocialXplain* also needs to define and implement strategies for generating abstract relationship explanations.

Unlike other work that computes the reputation of a person (e.g., Golbeck and Hendler, 2004), we aim at providing explanations concerning the relationship of a person to the user by analyzing social networks. Focusing on relationships rather than reputation allows our approach to provide explanations relative to any particular user. Our approach could be viewed as describing contextual importance relative to a user rather than reputation relative to a community.

Some examples of how the proposed approach might work in scenarios, including tablets and cell phones follow. Consider a situation where someone may want to filter or prioritize emails according to the relationship between the recipient and sender. In another scenario, cell phone users may want to filter ringing or answering based on their potential relationship to the caller. In the scenarios, we will explore how social network navigation and reasoning may be used to determine relationships and also how our explanation infrastructure may be leveraged to deliver the information in an operational manner. The scenarios include examples with cell phone and web browsing capabilities and access to data sources such as web accessible phone books, personal and public email address books, and friend of a friend (FOAF) files available from social networks. We initially intend to leverage FOAF files available on the web, but *SocialXplain* is open to exploring proprietary social net data such as Orkut's, MySpace's, LinkedIn's, etc.

In this paper, we describe our approach for generating explanations of relationships described in social networks. In particular, we describe a methodology used to explain relationships using "community discovery" in a way that computational effort is minimized.

Social Networks

Social networks are, naturally, represented in the form of graphs, where each person corresponds to a vertex and the edges represent the relations between such persons. The degree of a vertex equals the number of edges that it possesses. Studies show that, in a social network, the degree of a vertex is on the order of hundreds. One of the most interesting and influential studies about social networks was conducted by Milgram (Milgram, 1967). His experiments inspired various others and came to identify some of the foremost characteristics of social networks. Some of these characteristics have been observed in other kinds of networks as well. Among those characteristics, we can point out the following:

- Small diameter and mean path. The first conclusion after Milgram's experiments was that the social networks present a low mean-shortest path length between pairs of vertices.
- Strong group. Social networks tend to possess a high clustering coefficient, reflecting the organization of the individuals in communities.
- Distribution according to a law of power. This distribution reflects the existence of few vertices with a very high degree and many vertices with a low degree.

Currently, the great popularity of virtual social networks such as Orkut, Facebook, and LinkedIn, among others, verifies that people are willing to make their personal information available in exchange for facilities in social relationships via the Internet. Initiatives such as FOAF (friend of a friend) and, more recently, the Google Social Graph API, point toward a strong trend of formation of a single virtual social network, as distributed and accessible as the network formed by Web pages.

Discovery of Communities

Communities are subsets of densely interconnected vertices, but that present few connections among themselves (the subsets). The ability to find communities in a network has several practical applications. In a social network, such communities can represent social groups; in a collaboration network, they can represent articles about a given topic; on the Web, they can represent pages on related matters.

Some methods try to identify the graph edges that are found within the communities, others try to identify the edges that are found between communities. Edge-Betweenness (Girvan and Newman, 2002) is an example of a method that discovers the communities of a graph by progressively removing weak edges of the original graph. Its complexity is on the order $O(m^2n)$, where m is the number of edges and n is the number of vertices. Such complexity makes its application impracticable in large and dense networks such as social networks. For this reason, we have dedicated special attention to another method that makes an analogy between networks and electronic circuits and finds communities in a time that is linearly scaled with the graph size.

An Analogy with Electric Circuits

This method does not use the edge-removal strategy, since it is based on the notion of a decrease in voltage between the nodes of a network, in an analogy with electric circuits. It is capable of discovering communities in a time that is linearly scaled with the graph size (number of vertices and edges). Another interesting quality of this algorithm is the possibility of discovering a community that contains a particular vertex, without the need to extract all the communities from the graph (Wu and Huberman, 2004).

We will illustrate the operation of this algorithm showing how it behaves in the simplest case – dividing a graph into two communities. Let's suppose, also for the sake of simplicity, that we already know that nodes A and B belong to different communities, which we will call C_1 and C_2, respectively. Then we will show how we can get around the fact that we have no knowledge that A and B belong to distinct communities, which is the most common situation. We assume each edge as being a resistor with the same resistance and that the terminals of a battery were connected to *A* and *B*, leaving them with fixed voltages of 0 and 1. In this way, it is possible to see the graph as an electric circuit with a current flowing through its edges. Utilizing the equations of Kirchhoff (Kirchhoff's circuit laws)—well-known in the study of the electricity in physics—it is possible to obtain the voltage value in each node. This value should vary at a pre-defined interval; let's say from 0 to 1. Through the value of its voltage, we can determine if a vertex belongs to C_1 or to C_2. More specifically, we can affirm that a vertex belongs to C_1 if its voltage is higher than a threshold—let's say 0.5—or belongs to C_2 if its voltage is lower. In the case of a tie, some tie-breaking criterion must be used, such as (for instance) opting to classify the vertex as belonging to the larger of the communities.

For example, consider a vertex *C* that is connected to *n* neighbors $D_1,...,D_n$. Kirchhoff's equation states that the current that enters *C* is the same current that leaves it. Therefore the total flow of current in *C* should add up to zero. As shown in Equation 1.

$$\sum_{i=1}^{n} I_i = \sum_{i=1}^{n} \frac{V_{D_i} - V_C}{R} = 0, \qquad (1)$$

The voltage of any vertex is the average of that of its neighbors. If the majority of the neighbors of *C* belong to community *X*, the value of V_C will indicate that *C* belongs to *X*. This algorithm functions by calculating, on an iterative basis, the voltage of all the vertices. Afterwards, by analyzing the spectrum of the calculated voltages, the algorithm seeks the largest gap that is nearest the center of the spectrum. Figure 1 shows the spectrum of voltages of a graph. The dashed green line shows the gap chosen to make the partitioning.

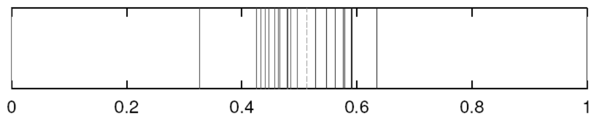

Figure 1. Division of the voltage spectrum in two candidate communities.

As mentioned previously, in the most common case, we do not have the prior knowledge that the vertices that will be connected to the terminals of the battery (A and B) belong to distinct communities; and if two vertices are chosen that belong to the same community, the result of the algorithm will not be correct. This is called the “poles problem.” Wu proposes a probabilistic method to get around this problem. The method consists of randomly choosing two vertices and dividing the graph into community candidates, and repeating this several times. Approximately half of the results will be correct. To improve the method of selection of the poles, just choose vertices that are not neighbors. By doing so, the probability that we will choose vertices of the same community is less than 50%, suggesting that most of the results are correct. It is thus possible to apply a majority vote to determine the communities.

If we need to find the community of a particular vertex *D*, instead of all the communities of the graph, it is possible to make the process even more efficient. To do so, instead of randomly choosing two vertices, we can set vertex *D* as one of the terminals and divide the graph into two communities: the community of the vertices that belong to community *D* and the community of those that do not.

Explanation of Social Relationships

We believe that, in various situations, it is highly useful to consult details about the origin of a piece of information that is being presented to us, if we have access to the data of the social network to which its producer belongs. Knowing such producer’s relationships, habits, characteristics and other production is fundamental for us to measure the degree of reliability of that source of information, which was theretofore unknown. Notice that, among the cited items, social relationships are the most difficult to be forged, assuming that—in a reliable virtual social network—the confirmation of both parties is necessary in order for a relationship to be established. In this situation, explaining the origin of a piece of information based on the social relationships of its producer can be a reliable way to attribute a degree of reliability to such information (Golbeck and Hendler, 2004).

Virtual social networks, in addition to information on relationships, generally possess large quantities of personal data entered by the users themselves. This data can be utilized to supply other forms of explanation: everything from simple retrieval of data (like displaying personal properties of the sender such as surname, city of residence or place of work) to the achievement of comparisons with the properties of the receiver aimed at displaying similarities.

An efficient explanation must provide the receiver with sufficient information to decide, for example, if he/she should trust the email content. We believe that, by analyzing the relationships in common with the sender, in certain situations, it is even possible to infer the intention of the person who is seeking the receiver at that time. The analysis of this explanation can also be utilized to prioritize his/her incoming emails. In this way, a differentiated priority would be assigned to emails from the closest people.

There are situations in which characterizing the relationship between two people is not trivial. Presenting the friends that connect both people can generate an explanation that is hardly informative, or even confusing. This situation occurs when the distance between the vertices is very long (This is José, who knows... who knows... who knows... who knows you). It is necessary to implement alternative strategies of explanation. The alternatives must vary with regard to the cost of processing and quantity of information produced in the explanation. Below, we will present several possible strategies of explanation capable of characterizing an unknown person on the basis of information contained in social networks.

Characterization by Properties

Virtual social networks, in addition to storing and sharing relationships among people, also store (and share) personal information on the participants in such networks. This constitutes one of the main reasons for controversies of virtual networks regarding security and privacy of the participants. On the other hand, some relationship networks utilize such information to induce a meeting between two people with similar characteristics. This is the simplest of the strategies and can be used if all of the others happen to fail or when there is no availability of a computer environment robust enough to perform algorithms on large and complex graphs. It consists, basically, of presenting the visible information (virtual social networks generally use levels of visibility for the attributes contained in the user profiles) for the receiver of the message to examine before accepting the invitation for communication.

Imagine that you have just taken part at an event in the state of Ceará, Brazil, and several days later you receive an email from an unknown sender. Upon requesting information about the sender, you get an explanation saying: This is **José** who lives in **Ceará, Brazil**. This explanation would be sufficient for you to assume that it must be someone you met on your trip. The application responsible for generating the explanation might have a rule saying that, if the sender resides in a location different that that of the receiver, it is informative to display the property that relates to Place of Residence in the explanation.

Characterization by Properties in Common

In this explanation strategy, we make a comparison between the attributes (characteristics) contained in the profiles of the sender and of the receiver and we present the possible similarities. For example, by utilizing this strategy, we could produce the following explanation, based on values of the "Place of Work" property: "This is José who has the same Place of Work as yours." To produce this type of explanation, a prior mapping would be necessary that identifies those properties—modeled by the social network under analysis—that could be considered informative by the receiver for the identification of an unknown sender when the value thereof is equal for both.

Characterization by the Shortest Path (Indirect Relationship)

In situations where the vertices are found at a maximum of three steps away, we consider it informative to show the shortest path (or a subset thereof, when there is more than one) as a way of characterizing a producer of information. This strategy is commonly presented by virtual social networks when we access the page (profile) of a person who is not on our personal list of friends. It is easy to perceive that the explanation produced by this strategy tends to get longer as the shortest distance between the sender and receiver increases. Dijkstra´s shortest path algorithm (Dijkstra, 1959) with a maximum depth was implemented for computing the distance between sender and receiver

Characterization by the Community to which the Sender Belongs

To explain the relationship between two vertices that are more than three "steps" away in a more informative and succinct manner, we can seek a way to characterize the sender by the community to which it belongs. First, we identify that community using the method proposed in (Wu and Huberman, 2004) because it features low complexity and enables the discovery of the community of a predetermined vertex. Having done that, we need to characterize the community that was discovered in order to order to cite it in the explanation. Such a strategy is used whenever the *sender* belongs to a different community as the *receiver*. Based on the small-world property, we can affirm that the chance that we will find a member of this group who has a close relationship with the receiver is quite high. In our tests with a graph generated from FOAF files with average

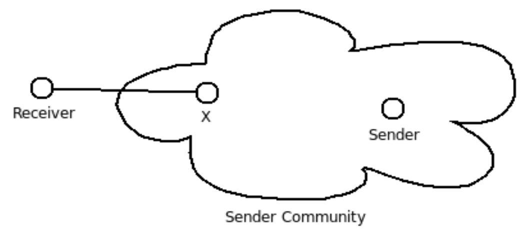

Figure 2. Characterization of the sender's community by a third participant closer to the receiver.

shortest path of 5.6 steps, after community identification the average shortest path between a sender and the receiver community is 2.4 steps. Once that closest member has been identified, we have a way to characterize the discovered community. We can characterize it as "the Community of the member". Then it becomes possible to build the following explanation: "This is **José** who belongs to **the community of Maria** who knows you". Figure 2 illustrate this situation

Capturing the Social Net Evolution

Virtual social networks are dynamic in nature. New members are frequently included while others are removed. Similarly, new relationships are established at a time when others cease to exist. Capturing this dynamic nature is essential when searching for information implicit in the structure of the network. In particular the process of community discovery - which is computationally very expensive - needs to take into account the latest network updates to provide accurate results. However, applying community discovery algorithms to each request for explanation in large networks is computationally impractical. Therefore we propose the implementation of the calculation of voltages in spaced moments. In the initial stage (startup of the application), the calculation of voltages runs n times (say n = 50). Each implementation of the calculation of voltages is a spin. Each round produces a set of m community candidates that, at the end of n rounds, produces a set of S (*n x m*) community candidates. Pairs of random vertices are chosen as poles of the "circuit" in each round. The community candidates resulting from this process reflect the current state of the network, say, at time $t = 0$. On receiving the first request for explanation, whereas the underlying network may have changed since the initial moment, a recalculation of voltage is run by setting up the vertices representing both sender and receiver as poles. As a result of this new round, a new set of community candidates is obtained and embedded in the initial set S. In that

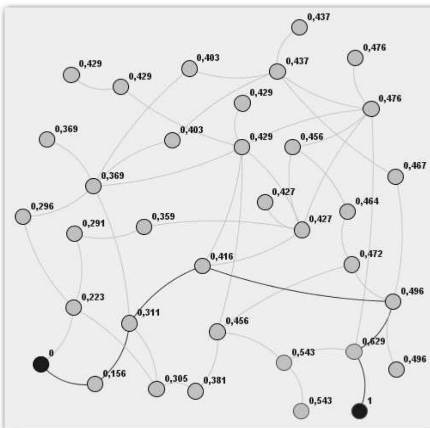

Figure 3. Graph with voltages between a sender(voltage 1) and receiver(voltage 0).

instant, the candidate of the first round (older) is excluded of S. Each new request, the set S is updated with addition of the new candidate and exclusion of the oldest, We thus keep the size of S in $n \times m$ candidates in which the majority vote to identify the communities is done. In this particular case, as we are interested in discovering the community of the sender, we apply majority vote to find out, those that appear most often in the same community of the sender.

Using the Voltages as Heuristics in the Search for the Shortest Path

Another reason for choosing the voltage algorithm to identify the communities is that we are using the voltages computed by the algorithm as a gradient for driving the search of the shortest path between two vertices (see Figure 3 for illustration). We have then used this information as a heuristic for a hill climbing search for finding the shortest-path between two vertices. We have obtained 95% accuracy (in terms of succeeding in finding the shortest path) using such a heuristic. We also use these heuristic values for driving the search of the shortest path between the sender and the receiver community.

If the underlying social network provides support for the creation and maintenance of communities by the users themselves, we can further refine the explanation above. By consulting the user-defined communities, we could find those that most closely resemble the one that was automatically identified. Then it would be possible to build the following explanation: "This is **Joseph** who belongs to the community of **Unifor Professors**." Or we could even combine both of them in the following way: "This is **Joseph** who belongs to the community of **Unifor Professors** and to the community of **Maria**, who knows you."

In the next section, we describe the implementation of the prototype capable of performing the last two explanation strategies presented.

Implementation of the Prototype

We implemented a prototype that is capable—by analyzing the graph of a social network—of producing a combination of the last two strategies presented: characterization by shortest path and characterization by community. We chose Python as the language because it enables quick prototyping and because it is easily integrated with the C programming language. The use of efficient languages such as C is fundamental for applications aimed at manipulating large social graphs. In order for us to represent this graph, we used NetworkX, a Python package designed for the creation, manipulation, and analysis of complex graphs.

We have used FOAF files represented in a graph of 19.396 vertices e 175.034 edges with 5.38 on average of shortest path between two vertices.

Our model of the social network is read from a database and completely loaded in the memory. To avoid excessive consumption of the resources of the application's server, we store only one unique identifier in each node of the model in memory. The loading of the model is done only once, at the startup of the application. Figure 4 depicts the *SocialXplain* architecture where a crawler, periodically, updates the social network file from distributed FOAF files. This dynamic model is then used by the Explanation Module (EM) for elaborating explanations. In order to facilitate reuse, we developed that application as an inter-

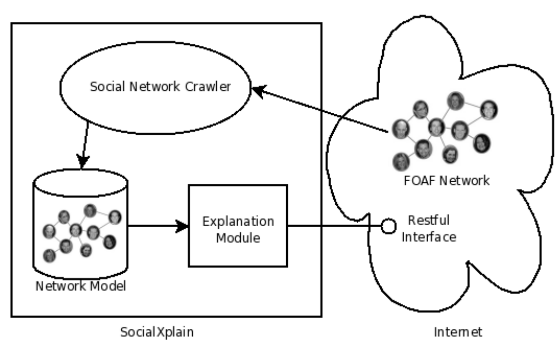

Figure 4. Schema of the Application as a restful service.

face in the restful service model (McMilian, 2003). Applications access the EM by a simple *http* call, thus creating a platform where *SocialXplain's* explanation service may be easily used by applications with many different platforms.

As an application example, we simulated an interface where the user logs on to the system, informing only his/her email. The system presents a list of messages generated automatically, having another participant of the network as the sender. The user has the option of asking the system who that sender is. The system then uses the following strategy to produce the answer:

1. If the receiver (user who is logged on) and the sender are located at most three steps away from one another, the system presents the common friend that unites them. Otherwise go to 2;
2. It identifies the community of the sender and uses the community member closest to the receiver to characterize that community in the explanation.

All the algorithms and strategies are implemented in Python using the networkx.Graph class as a model for the social network. We are also extending email clients for accessing this service from two kinds of Nokia´s devices: the N800 tablet and the S60 cell phones.

Conclusions

Due to their popularity, we can consider virtual social networks as an example of successful Web 2.0 applications. Technologies such as FOAF and Google Social Graph API point out the future existence of a social network that is distributed and accessible to users and applications. In this work, we show that it is possible to use the data contained in those networks to characterize unknown persons who send us some type of message.

Explanations based on social relationships can be considered more reliable due to the difficulty in forging such relationships. We can easily perceive that it is much easier to inform false values to the properties than to forge social relationships, since in the latter case, the participation of other network members is necessary.

Using an incremental approach, we are identifying the community using current information instead of relying on a frozen image of the network, which quickly becomes outdated. Thus, we capture the evolution of the underlying social network, without having to apply the complete calculation of voltages to each new request. Doing so, we are considerably reducing the computational effort in uncovering communities at large social networks.

Our future work will be explaining relationships between documents. It would be possible then to inform the reader in the web about his/her social relationships with the producers of documents accessed by him. Ranking of web pages would take into consideration this kind of social proximity. It would be possible to query documents and artifacts produced by members of a certain community. We also are beginning to generate PML (Proof Markup Language) proofs from explanations of social relationships. Using PML (McGuinness et al, 2007), we can share explanations across many different systems and also can combine explanations generated from diverse sources. We can also leverage the Inference Web (McGuinness and Pinheiro da Silva, 2004) tool suite for manipulating explanations. We also plan to enrich the Inference Web toolkit with some tools specially designed to explain the relationships of the observer of the proof—identified by means of the application login—with the people identified as the origin of information contained in the observed proof.

References

Dijkstra, E.: A note on two problems in connexion with graphs, Numerische Mathematik, 1959.

Girvan, M.; and Newman, M.E.J.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. 99 (2002) 7821–7826.

Golbeck, J.; and Hendler, J.: Accuracy of Metrics for Inferring Trust and Reputation in Semantic Web-based Social Networks. In International Conference on Knowledge Engineering and knowledge Management (EKAW), Northamptonshire, UK, Oct. 2004.

Kirchhoff's circuit laws [on-line]. Available from http://en.wikipedia.org/wiki/**Kirchhoff**'**s**_circuit_**laws**, accessed on 25 Jan 2008.

McGuinness, D.L.; Ding, L.; Pinheiro da Silva, P.; and Chang, C.: PML 2: A Modular Explanation Interlingua. AAAI 2007 Workshop on Explanation-aware Computing, Vancouver, British Columbia, Canada, July 2007. Also Stanford KSL Tech report KSL-07-07.

McGuinness, D.L.; and Pinheiro da Silva, P.: Explaining Answers from the Semantic Web: The Inference Web Approach. Web Semantics: Science, Services and Agents on the World Wide Web Special issue: International Semantic Web Conference 2003 - Edited by K.Sycara and J.Mylopoulis. Volume 1, Issue 4. Journal published Fall, 2004.

McMillan, R.: A RESTful approach to web services. URL: http://www.networkworld.com/ee/2003/eerest.html (2003). 45. IBM: The TSpaces programmer’s guide.

Milgram, S.: The Small World Problem. Psychology Today (1967) 60–67.

Wu, F.; and Huberman, B.A.: Finding communities in linear time: a physics approach. The European Physics Journal B (2004).