Detecting Communities in Time-evolving Proximity Networks

Saurav Pandit*, Yang Yang*, Vikas Kawadia†, Sameet Sreenivasan‡ and Nitesh V. Chawla*
*University of Notre Dame
Notre Dame, IN 46556
Email: [spandit,yyang1,nchawla]@nd.edu
†Raytheon BBN Technologies
Cambridge, MA 02138
Email: vkawadia@bbn.com
‡Rensselaer Polytechnic Institute
Troy, NY 12180
Email: sreens@rpi.edu

Abstract

The pattern of interactions between individuals in a population contains implicitly within them a remarkable amount of information. This information, if extracted, could be of significant importance in several realms such as containing the spread of disease, understanding information flow in social systems and predicting likely future interactions. A popular method of discovering structure in networks is through community detection which attempts to capture the extent to which that network is different from a random network. However, communities are not very well defined for time-varying networks. In this paper, we introduce the notion of spatio-temporal communities that attempts to capture the structure in spatial connections as well as temporal changes in a network. We illustrate the notion via several examples and list the challenges in effectively discovering spatio-temporal communities. For example, such communities are lost if the temporal interactions are aggregated in a single weighted network since the concurrency information is lost. We present an approach that first extracts concurrency information via node-clustering on each snapshot. Each node is then assigned a vector of community memberships over time, which is then used to group nodes into overlapping communities via recently introduced link clustering techniques. However we measure similarity (of nodes and edges) based on concurrence, i.e. when they existed, if they existed together. We call our approach the co-community algorithm. We validate our approach using several real-world data sets spanning multiple contexts.

Index Terms

Data mining, community detection, social network, temporal data, contact graph.

1. Introduction

A distinguishing feature of several real world networks, especially social networks, is the presence of *communities*. Communities are loosely defined as relatively dense subgraphs within the network, and represent shared attributes or shared social relationships between the nodes within. Here we study the problem of detecting spatio-temporal communities in time-varying networks that are formed by the changing interaction patterns between individuals over a period of time. For the datasets considered in this study, the existence of an edge between two nodes implies that the two nodes have been in within a certain pre-defined distance of each other. Thus, a 'contact' or interaction between individuals in our study is exclusively defined by the notion of spatial proximity and does not necessarily imply a personal relationship or direct communication between the individuals. Characterizing and extracting meaningful community information from such proximity based networks is of significant importance in areas ranging from epidemic prevention and control [1], to the design of routing algorithms in pocket-switched networks [2]. The datasets that we use in this study were collected by tracing the contacts between individuals using Bluetooth® enabled phones and RFID badges.

978-1-4577-1051-3/11/$26.00 ©2011 IEEE

1.1. Spatio-temporal communities

In this section, we attempt to define the notion of spatio-temporal communities in time varying spatial networks. Recall that the community structure of a non-time varying network captures the extent to which that network is different from a random network. Spatio-temporal communities attempt to capture the structure not just in spatial connections, but also the structure in temporal changes in a network. They tell us how different a real time-varying network is from a random network that is changing randomly over time. While, we are still working on a quantitative definition of spatio-temporal communities, we illustrate the concept via few examples.

Consider a human proximity network, where an edge represents geographical proximity between the human nodes. This network is time varying due to the human mobility associated with routine activity. Spatio-temporal communities in such a network would be groups of nodes that work together, live together, or do some social activity together. Such communities are not necessarily cliques; a richly connected set of nodes could very well be a community even if not all nodes are in very close proximity of other nodes. Note that node in a spatio-temporal communities do not have to be continuously in proximity to other nodes in the community. A frequent, but not-necessarily contiguous, connectivity pattern could very well define a spatio-temporal community. Spatio-temporal communities in a human proximity network reflects the fact that human mobility and location has some structure and order and does not originate from random movements.

As another example, consider a wireless mobile ad hoc network formed of nodes that are moving over time. Nodes in such a network store and relay packets for each other, thus allowing communication between distant nodes via multi-hop relaying. Communication is feasible between pairs of nodes even if the network is not connected at every instant, since nodes can store packets over time and relay them. Thus, communication is supported by both spatial as well as temporal relaying. Spatio-temporal communities in such a network is a (potentially overlapping) partition of nodes in groups such that intra-group communication has a much higher success probability than inter-group communication. A randomly evolving random network will have no spatio-temporal community structure.

1.2. Challenges

There are several challenges in uncovering meaningful spatio-temporal communities in time-varying interaction data. We elaborate item-wise on these challenges here:

(1) Firstly, spatio-temporal communities are very likely to be overlapping. The natural notion of communities in proximity-based interaction data arises due to the simultaneous presence of individuals within a small spatial area. However, if the data is collected over a significant period of time, then realistically, any observed node would have significant changes in its spatial location over the time period, and therefore could potentially belong to several communities (signified by proximity in distinct spatial locations). For example, if data is collected over a week, an individual may reliably be observed in close spatial proximity to his colleagues during the workday, to a distinct set of friends in the evenings, and to his family members after dinnertime. A competent community detection algorithm must detect all these distinct community memberships for each node.

(2) Secondly, the spatio-temporal community structure may be lost if the temporal snapshots are merged into a single graph. Forming communities by associatively agglomerating pairs of individuals that have a high co-location probability can be misleading. For example, a married couple that live together and share the same workplace have a high co-location probability with both their family members as well as their work colleagues. Simply using co-location between individuals may not be sufficient to disambiguate the couple's 'workplace' community from their 'family' community. A competent algorithm must therefore do more than traditional community detection on a merged weighted network with weights representing co-location probabilities.

(3) Spatio-temporal communities may not always consist of nodes that interact continuously over a certain time period. Certain community memberships may be associated with non-contiguous events that may be predictable but may not be very frequent. For example, members of a book-club may meet once in two weeks, and churchgoers every Sunday. A competent algorithm must prevent such infrequent co-location events from being washed out by more frequent ones.

(4) Finally, note that temporal interactions may have gaps. It is to be expected that during the collection of data, there could be periods in where interactions are absent or escape detection due to the inactivity of individuals (late hours) or due to malfunctioning or switched-off devices, among other reasons. A competent algorithm must be immune to such "gaps" in the data.

In addition to the above, a practical challenge in rigorously studying community detection in proximity

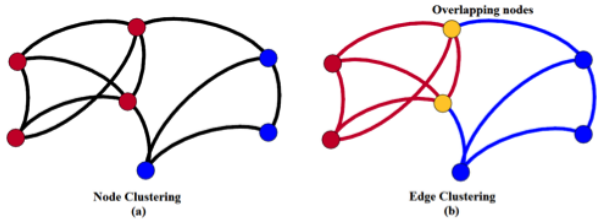

Figure 1. Fig. (a) shows two communities as a result of node-clustering. Each node belongs to a unique cluster. However, the two nodes in the middle can be considered part of two different communities. This is achieved in Fig. (b) using link clustering.

based interaction data has been the paucity of datasets that contain both spatial and temporal information. However with the proliferation of mobile devices and social applications on them, and due to the general interest in understanding human mobility, such spatio-temporal datasets are increasingly being collected and being made accessible to the research community.

Furthermore, a common problem in community detection is the validating whether a given algorithm is detecting "meaningful" communities, since the notion of "meaningful" depends intricately on the type and context of the data. Arguably, the best method to validate the significance of the uncovered communities is commonsense validation, i.e. comparing the output against the *ground truth* embedded in the datasets. Several datasets, including some considered in our study, contain attributes or "tags" for each user that can be helpful in revealing the true nature of such detected communities.

Community detection has been studied heavily in the past decade by computer scientists and physicists. This has resulted in a diverse array of algorithms predominantly designed to detect communities within a single network. Despite the diversity of the approaches, a common feature in many of the algorithms has been the use of *modularity* [3] as an objective function in guiding the search for the best decomposition of the network into communities over the space of all possible partitions (which, when exhaustively performed, is an NP-complete problem). Even a modularity maximizing approach does not guarantee the optimal decomposition of the network into communities due to the large degeneracy of the modularity landscape [4]. An in-depth discussion on the advantages and disadvantages of the various community detection algorithms is beyond the scope of this paper; however, we briefly mention the intuition behind a particular method, *Walktrap* [5], for the clustering of nodes into communities, that we use in one of the stages of our algorithm. Walktrap is based on the idea that a random walk on a network is likely to get trapped in regions where the density of edges is higher. Thus, the first passage times of a random walk between a pair of nodes reflects their cohesiveness, and Walktrap uses an appropriate *similarity* measure derived from these first passage times, to group nodes into communities.

However, as we pointed out in Section 1.2 (1), realistically, a node can belong to several communities, therefore something in addition to simple node-community detection is required to such community *overlaps*. Link clustering [6], [7] has been shown to be an effective method in uncovering overlapping communities, and is based on the idea of classifying links into groups, rather than nodes as in traditional community detection. The basic approach to clustering links (following [6]) is to compute the similarity between pairs of links and group links together based on this similarity. The idea of link clustering has so far been applied only to static graphs, and we show through its incorporation in our algorithm, that it plays an important rule in disambiguating the various social group memberships that appear in datasets collected over an extended period of time, specifically in the context of proximity based data.

We should mention another recent work [8] that has dealt with "multi-slice" community detection. Whereas a "slice" is a more general abstraction than time-interval snapshots, that algorithm is not specific to proximity networks. Also, we found that the introduction of "inter-slice" edges drastically increase the edge density, which is detrimental to obtaining good link clustering.

1.3. Intuitive description of our approach

Next, we provide a brief overview of how node and link clustering, and information on the simultaneity in edge occurrence, can be combined to surmount the challenges presented in Section 1.2. First, we assume that for any dataset one can find a sufficiently small interval of time Δt, over which the co-location of individuals does not vary significantly. Then, we can divide the dataset into distinct *snapshots*, each of which is a network obtained by aggregating the data over a given Δt. According to our assumption, within each of these snapshots, each node will belong with high probability to a single community, since Δt limits the number of distinct spatial positions that the node adopts, and thereby the number of distinct groups of nodes with which it can be spatially co-

located. Thus, it is justifiable to use a node-community detection algorithm within a snapshot to yield each node's singular community membership within that snapshot.All snapshots can similarly be decomposed into node-communities.

The procedure described above yields the participation of nodes in different communities at different points in time. How can we combine these effectively to produce a coherent aggregate measure of communities and the degree of each node's participation in them? For this, we resort to link clustering. If the different network snapshots are merged into a single network, then it is possible to devise a measure of similarity between two links, based on the different snapshot communities that their individual endpoint nodes belong to, and then clustering the links based on similarity. What this measure of similarity captures effectively, is the *concurrency* between the occurrence of links. Thus links are considered similar only if they have occurred simultaneously frequently enough, thereby justifying their grouping into the same community. Taking into account concurrency alleviates several problems pointed out in Section 1.2 : It allows for the disambiguation of communities that a node is a member of at distinct points in time (challenge presented in (2)), while still capturing infrequent but recurring events that reliably cause nodes to be co-located (challenge presented in (3)). Finally, the result of link clustering is to allow each node to have multiple community memberships, thus overcoming the challenge presented in (1).

It is worth explaining why link clustering is such a crucial step towards our goal. First, as shown in Figure 1, link clustering can detect various different memberships of a node. In addition, as noted in [6], it can also detect membership patterns that may not be so apparent in the network. For example, consider a very close group of students in Figure 2. However, some of the students in this group have more in common, as they are in the rowing team, and hence connected to the team coach as well. A link clustering algorithm is likely to detect the difference between two types of edges and extract the information that the students (who are on the team) have more than one thing in common. Also note that our algorithm looks at the time these edges were present in the dataset. Hence the similarity within the black edges and the purple edges will be further reinforced and the difference between a black edge and a pink edge will be further magnified.

Secondly, the structure or shape of a link cluster can give us an idea about the "meaning" of that community. For example, let us consider Figure 2 once more. The purple edges form a community around the coach.

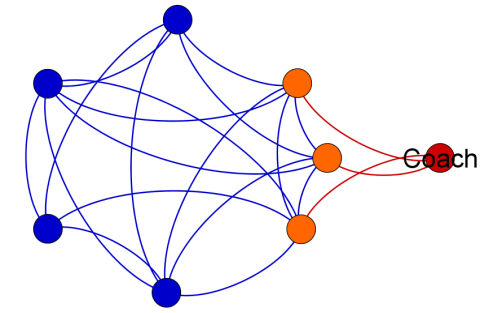

Figure 2. Consider a group of students, some of whom are in the rowing team, and the coach of the team. A node clustering algorithm is likely to put all the nodes in one cluster, or all the students in one cluster with the coach being in a solitary cluster. Due to link clustering, we can discover the additional star-shaped community around the coach.

Even though the edges between the members of the row team are present, they are not part of this community. That means their relationship is predominantly defined by the fact that they are members of that group of students. It is important to note here that if two nodes have multiple memberships in common, the edge between them will belong the community that contains edges that are "more similar" to the said edge. In terms of relationship between the two nodes, it makes sense to define this to be their predominant relationship. This idea easily extends to our temporal version as well - more time spent together is seen as a stronger relationship.

Finally, we show an example of how our algorithm can further enhance the benefits of link clustering. Consider a group of friends, as shown in Figure 3. It is possible that a small subset of those people are much closer to each other or spends more time with each other than the others. Node clustering, or even simple link clustering will not be able to detect this pattern. However, our algorithm will deem the edges a higher similarity value if they appear together more often. Hence it is likely, that we will be pick up such "core" groups within groups.

Our algorithm, as already insinuated, is more than just node clustering or link clustering. It's a combination of both and a few other modules, that is described in the next section.

2. Algorithm

The overall scheme of the algorithm is as follows: divide the data into network snapshots, compute each

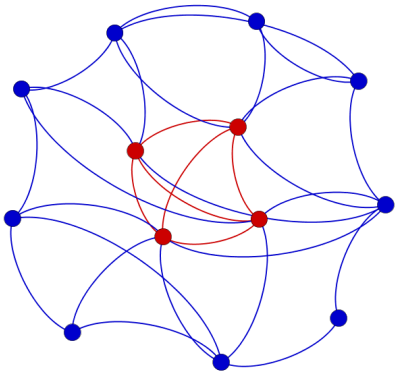

Figure 3. A core group within a group. Our version of link clustering allows making this distinction between two groups of edges. Hence, few members will be detected as part of both the core group and the complete group.

node's membership within snapshots, define similarities between edges based on the membership of their endpoints at various snapshots, and finally perform link-clustering based on those similarity values. Before we give a more formal description of our algorithm, we present a brief overview of the link clustering algorithm in [6] which is an important constituent of the final step of our algorithm.

Suppose we have a static graph $G = (V, E)$. The goal of link clustering is to group the similar edges together. For simplicity and efficiency, only connected pairs of edges (i.e. sharing a node) are considered - since it is unlikely that a pair of disjoint links are more similar to each other than a pair of links that share a node. For a connected pair of edges (k, i) and (k, j), the shared node k is called a *keystone* node and i and j are called impost nodes. Let $N[i]$ denote the closed neighborhood (i.e. including the node itself) of a node i. Similarity between edges (k, i) and (k, j) are defined as

$$S(i,j) = \frac{N[i] \cap N[j]}{N[i] \cup N[j]}$$

An iterative algorithm groups the edges together from higher to lower similarity values, and a hierarchical clustering is obtained. This hierarchical clustering is represented in the form of a *dendrogram*. The dendrogram is a tool to visualize the iterative process from the bottom (when all links are in solitary clusters) to the top (when all links belong to the same cluster). The final output depends on where we "cut" this dendrogram. For better quality of clustering we cut this dendrogram where the partition density is maximized. Please see [6] for more details.

Now, we present a detailed description of our algorithm. First we decide on a time interval Δt, that we call *snapshot interval*. The snapshot interval varies from one dataset to other, but is usually much smaller than the total time that the dataset ranges over. Once this interval is fixed, we divide the event time-line into distinct Δt intervals and aggregate the interactions within each snapshot interval to generate *network snapshots*. Once the network snapshots are generated the remaining modules of the algorithm are as follows:

1) Suppose we have m snapshots: $G_1, \ldots, G_m$. Perform node clustering, using the walktrap [5] algorithm, on each snapshot graph. [1]
2) Each node i has a vector of size m, let us call it $\mathbf{c_i}$. Let $c_i^{(v)}$, i.e. the v^{th} member of $\mathbf{c_i}$, is the *clusterID* of the cluster that i belongs to in G_v. If i does not exist in G_v, then $c_i^{(v)}$ can hold a null value (Φ). Let, m_i denote the number of of elements in $\mathbf{c_i}$ with non-null values, i.e. the number of snapshots where the node has appeared.
3) Using consistent notation from [6], let us fix k to be a keystone node in the *merged graph*, and i and j are two neighbors of k. Let, for each $p \in N[i] \cup N[j]$,

$$n_p(i,j) = |\{v : c_p^{(v)} = c_i^{(v)} = c_j^{(v)} \neq \Phi\}|$$

Measure the similarity of the edges (k, i) and (k, j) as

$$S(i,j) = \frac{\sum_{p \in N[i] \cup N[j]} n_p(i,j)}{\sum_{p \in N[i] \cup N[j]} m_p}$$

4) Perform link-clustering based on the above similarity measure.

The *merged graph* in Step 3 refers to the network containing all the nodes and edges that appear at least once in the dataset. Note that edges in the merged graph are not weighted. However, the information on concurrency is not lost - it is stored as $\mathbf{c_i}$ for each node i.

We emphasize here how the algorithm meets the challenges presented in Section 1.2. Firstly, the use of link clustering in the final stage (Step 4 above), allows us to assign nodes to multiple communities. The numerator of the similarity function ensures a high similarity value if there were many nodes that were present together, at the same time and at the same place. It is assumed that if that was the case, all such

1. Our approach is not tied to Walktrap - other node clustering algorithms can be used as well.

nodes are very likely to be detected in the same node cluster in Step 2. We also ensure we pick up infrequent communities, i.e. if some nodes do not appear a lot, but when they appear, they appear at the same time. Note that in such cases, even though the numerator of the similarity function is smaller, so is the denominator. In other words, $\frac{n_p(i,j)}{m_p}$ will be almost 1, no matter if p appears on 5 snapshots or 50 snapshots. There is no assumption of continuous appearances either. Nodes can appear after a long gap and that does not affect the algorithm.

3. Experimental Results

We present the results of our experimentation on the MIT Reality Mining data [9] and on the COMSNETS data [10]. Whereas we have successfully experimented on other spatio-temporal datasets and there are other experiments that are on-going, in this paper we limit our experimental discussion to the two aforementioned datasets.

3.1. MIT Reality Mining Dataset

100 people were given Bluetooth enabled cellular phones. Of them, seventy-five users were either students or faculty in the MIT Media Laboratory, while the remaining twenty-five were incoming students at the MIT Sloan business school adjacent to the laboratory. The information collected includes call logs, Bluetooth devices in proximity, cell tower IDs, application usage, and phone status (such as charging and idle) etc. However, we are only interested in proximity measures. The data is in a format of a series of *events*. An event occur when two monitored Bluetooth devices (and hence it is to be assumed their users) came in close proximity of one another.

We have sliced this data into snapshots of size 1 hour, 2 hours, 3 hours, 12 hours and 24 hours in order of experimentation. For the 1, 2, and 3 hour snapshots, we chose to use a relatively active week, instead of the whole dataset of about 9 months. For example, if 1 hour snapshots were chosen, then there will be 168 snapshots in a week. The results are consistent. All the regular attendees of the media lab and Sloan business school show up in a large cluster, as expected. In all the experiments a second sizable community is also present that is highly overlapping with the large cluster. This second group mostly consists of Sloan students. See Figure 4 for more details. There are several other smaller, star-shaped clusters that show up at various frequency resolutions which are usually centered around professors or senior graduate students.

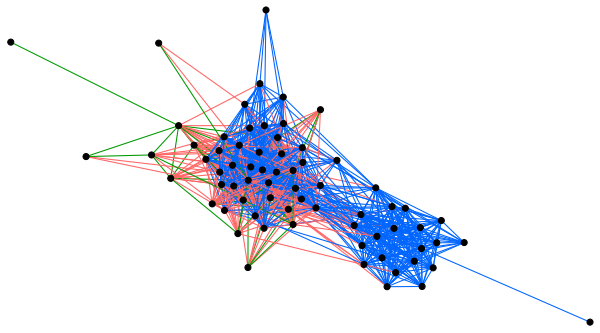

Figure 4. View of the full graph with 1 hour snapshots over a particularly busy week in the Reality Mining dataset. The "blue" community is that of the Media Lab. Some of them are overlapped with the "red" community that is of Sloan students. There are some smaller communities that were detected as well, of which one (the "green" community) is prominent in the picture.

The star-shaped nature of these communities tell us that they are likely to be smaller research groups that, when met, met in the presence of the central node.

3.2. COMSNETS Dataset

The COMSNETS data [10] was collected using RFID badges. The location measures (co-ordinates) were precise. Experiments were run using 30 minutes, 1 hour and 2 hour snapshots. One large cluster was consistently detected due to the large number of mutual contacts during the registration sessions. Four other sizable clusters were detected - these were found to be people of similar interest attending many of the same sessions. See Figure 5 for more details. Additionally, the experiments produced more than a hundred very small clusters, most of them diads and triangles. These are likely friends and colleagues attending the sessions together. For the purpose of clarity, these small clusters are have been removed from Figure 5.

4. Conclusion

Although the early experimental results show remarkable success, this is a work in progress. We are analyzing our algorithm on more spatio-temporal datasets: Cabspotting (536 Taxi Cab locations in San Francisco over 20 days), Microsoft Geolife [11] (GPS trajectories of 165 users over 2 years), AMD Hope Conference [12] (1224 nodes, RFID card proximity data during a conference), Tore Opsahl's social network [13] etc.

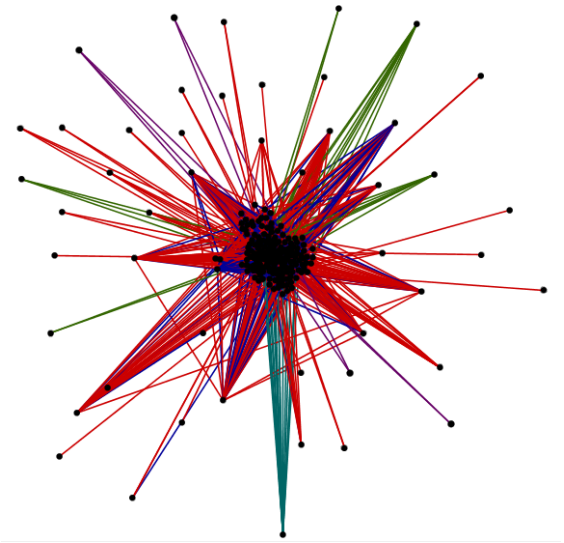

Figure 5. View of the full graph of the COMSNETS dataset using 30 minutes snapshots. The large "red" cluster originated from the registration sessions. Four other clusters are visible that overlap the red cluster.

A bigger goal is to build predictive models based on the output of our algorithm and test the model on the available datasets.

This algorithm was designed with contact/proximity graphs in mind. We pay much attention to concurrency, and spatial concurrency that has some properties that add to the success of this algorithm. For example if a node A is in proximity of B and C at the same time, then we can say B and C are close to each other as well. This would not have been the case if the relationship was based on email or phone communications. However, if the snapshots were long enough, such communications may start making sense. For example, changing friendship patterns over the years. In the future, we want to look into extending this algorithm for non-spatial relationships as well.

Acknowledgment

This research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] M. Salathé and J. H. Jones, "Dynamics and control of diseases in networks with community structure," *PLoS Comput Biol*, vol. 6, no. 4, p. e1000736, 04 2010.

[2] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, ser. MobiArch '07. New York, NY, USA: ACM, 2007, pp. 7:1–7:8.

[3] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[4] B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Phys. Rev. E*, vol. 81, no. 4, p. 046106, Apr 2010.

[5] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *J. of Graph Algorithms and Applications*, vol. 10, pp. 284–293, 2004.

[6] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multi-scale complexity in networks," *Nature*, vol. 466, 2010. [Online]. Available: http://arxiv.org/abs/0903.3178

[7] T. S. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Phys. Rev. E*, vol. 80, no. 1, p. 016105, Jul 2009.

[8] P. Mucha, T. Richardson, K. Macon, M. Porter, and J. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Science*, vol. 328, no. 5980, p. 876, 2010.

[9] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," 2005.

[10] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A Parsimonious Model of Mobile Partitioned Networks with Clustering," in *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*, January 2009. [Online]. Available: http://www.comsnets.org

[11] Y. Zheng, X. Xie, and W. Ma, "GeoLife: A collaborative social networking service among user, location and trajectory," *IEEE Data Engineering Bulletin*, vol. 33, no. 2, pp. 32–40, 2010.

[12] aestetix and C. Petro, "CRAWDAD data set hope/amd (v. 2008-08-07)," Downloaded from http://crawdad.cs.dartmouth.edu/hope/amd, Aug. 2008.

[13] T. Opsahl and P. Panzarasa, "Clustering in weighted networks," *Social networks*, vol. 31, no. 2, pp. 155–163, 2009.