Efficient algorithms for analyzing large scale network dynamics: Centrality, community and predictability

Sima Das

EFFICIENT ALGORITHMS FOR ANALYZING LARGE SCALE NETWORK DYNAMICS: CENTRALITY, COMMUNITY AND PREDICTABILITY

by

SIMA DAS

A DISSERTATION

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

2017

Approved by

ABSTRACT

Large scale networks are an indispensable part of our daily life; be it biological network, smart grids, academic collaboration networks, social networks, vehicular networks, or the networks as part of various smart environments, they are fast becoming ubiquitous. The successful realization of applications and services over them depend on efficient solution to their computational challenges that are compounded with network dynamics. The core challenges underlying large scale networks, for example: determining central (influential) nodes (and edges), interactions and contacts among nodes, are the basis behind the success of applications and services. Though at first glance these challenges seem to be trivial, the network characteristics affect their effective and efficient evaluation strategy. We thus propose to leverage large scale network structural characteristics and temporal dynamics in addressing these core conceptual challenges in this dissertation.

We propose a divide and conquer based computationally efficient algorithm that leverages the underlying network community structure for deterministic computation of betweenness centrality indices for all nodes. As an integral part of it, we also propose a computationally efficient agglomerative hierarchical community detection algorithm. Next, we propose a network structure evolution based novel probabilistic link prediction algorithm that predicts set of links occurring over subsequent time periods with higher accuracy. To best capture the evolution process and have higher prediction accuracy we propose multiple time scales with the Markov prediction model. Finally, we propose to capture the multi-periodicity of human mobility pattern with sinusoidal intensity function of a cascaded non-homogeneous Poisson process, to predict the future contacts over mobile networks. We use real data set and benchmarked approaches to validate the better performance of our proposed approaches.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to God Almighty, for her unconditional blessings, kindness and support in helping me complete my Ph.D. In this journey the loving memory of my beloved father and brother always remained as an inspiration and comfort. I am deeply indebted to my advisor Professor Sajal K. Das, without his expertise, encouragement, support, and motivation, it would not have been possible today. I would also like to thank my committee members Professor Jennifer Leopold, Professor Wei Jiang, Professor Dan Lin, Professor Abusayeed Saifullah and Professor Egemen K. Çetinkaya for agreeing to serve in my committee, and for their support and help. Their suggestions and comments greatly helped me improve the dissertation. I am specifically thankful to Professor Jennifer Leopold for her support, opportunity and initiatives. Besides, I am thankful to Professor Bruce McMillin for organizing my research qualifier. I am also thankful to my Professors and advisors in my previous Universities who put their effort and trust in me, and helped me learn at every step. In addition, I am thankful to the Department of Computer Science and Council of Graduate Studies at Missouri Science and Technology, Rolla, for giving me the opportunity and support. I would also like to thank the office staff of Computer Science department, Dawn Davis, Christina Barton, Elaina Manson, Grayson Rhonda, and others for their support and help in the official work, specifically to Dawn Davis for her help in the official process and their successful execution. I also thank my friends for their support and encouragement. Besides the support in the professional circle, I remain deeply indebted to my family, my mother for always being there for me; for her continuous support, patience, encouragement, and inspiration to turn me to be a good human being. I thank National Science Foundation for financial support under award numbers CCF-1533918, CCF-1725755, CBET-1609642 and Intelligent Systems Center, MST, Rolla.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

We are living in an era of large scale networks [71]. With immense growth and use of the Internet and computing technologies in the last two decades, the large scale networks have profound impact on the society with application domains spanning from the Internet, transportation networks, social and opportunistic networks, smart grids, biological networks, (scientific and academic) collaboration networks, to mention a few; justifying active and profound research in large scale network analysis. To better analyze these networks, a substantial amount of research has emerged in the domain of modeling network evolution [95], analytical tools for visualization [16, 76], determination of empirical, quantitative indices and clustering of social networks [112].

More so, decreasing cost of mobile devices, and wide spread adaptability of technology further emphasizes the existence of large scale networks as an an integral part of our daily life, with much diverse application domains [6, 77, 121]. Large scale networks, though gives rise to the possibility of many applications and services at its disposal; the successful realization of these applications depend on efficient solution to various computational challenges originating with large network size, network dynamics, to mention a few.

These computational challenges though specific to applications, are all stemmed from the underlying large scale networks. Thus, the network characteristics and behavior should be analyzed in order to propose efficient algorithms to resolve the intrinsic computational problems over the networks, and result in efficient and successful applications.

The remainder of this discussion is organized as follows. Subsection 1.1 describes the applications and challenges over large scale networks, while Subsection 1.2 presents the motivations for the work in this dissertation. Subsection 1.3 states the main problems that are addressed in this dissertation and discusses our major contributions. Finally, the organization of this dissertation is presented in Subsection 1.4.

1.1. LARGE SCALE NETWORK APPLICATIONS AND CHALLENGES

Large scale networks, with its diverse network domains result in a wide range of applications, starting with routing, information dissemination, viral marketing (also determining market trends), disease propagation, scientific/ academic collaborations, new friendship in social networks, movies and songs a user will listen to, hidden links in anti-social activities, to maintaining a reliable and secure network at its disposal. More so, the explosive growth of Internet-related communication, decreasing cost of mobile devices like smart phones and tablets etc., along with their ever growing density, and the large scale deployment of sensors, led to the class of mobility induced dynamic networks. More so, with realistic inclusion of vehicular networks [6], smart health care, smart management of physical resources with realization of smart cities [77, 121], mobile social networks [74], and mobile sensor networks [72] over the past decade, the ubiquitous smart devices result in a large scale mobile, ad-hoc wireless network. These devices not only capable of sensing vital environmental and application specific data, but are also equipped with storage, computational, connectivity capabilities for successful realization of the applications, which depend on the real-time, efficient collaboration among the devices for data offloading [67], forming virtual cloud [47] etc.

The success and efficiency of the applications depend on various aspects like, availability of contact and link between node pairs (e.g. for collaboration, information spreading, routing etc.), the set of critical nodes or influence of a node (e.g. for maintaining reliable and secure networks and checking the spread of virus, in fast routing, fast information dissemination, to have effective vaccination strategies to check disease outbreak, to have guided spread of advertisements for effective marketing). For example, in a good opportunistic routing algorithm the challenge is to which node and when a node should pass message to, for the minimal latency, maximal coverage and is dependent on knowing the timely contacts ahead of time, and the effective (central) nodes that needs to act as major carriers.

These computational problems, such as determining influential nodes, finding links and contacts between node pairs are in fact affected by the type of influential metric that is considered (that causes possible increase in computational cost), the underlying network structure, the intermittent and dynamic nature of networks that affect connectivity, to mention a few. These computational challenges are further escalated with large size of the underlying network.

Obviously underlying every large scale networks, there is a network graph. The network elements are the nodes and the contacts or interactions represent the edges. In stationary graphs, the connectivity or edge remains (or is assumed to remain) unchanged and is the aggregated graph of the contacts over the whole time domain. Whereas, the network dynamics is best reflected over time domain is usually known as dynamic networks.

The inherent dynamic nature of large scale networks, stems with underlying interactions (as in biological, bio-informatics, social networks) or caused due to mobility or environmental factors (as in ad-hoc wireless networks, mobile-social networks, vehicular networks, opportunistic networks) [27, 107, 126]. The dynamic nature of these networks as a result of their structural changes over time. Change in network structure results from the inclusion of new or exclusion of existing nodes, and presence or absence of links over time. In reality the exclusion/ deletion is a rare event whereas, inclusion of new nodes is prominent, resulting in denser network over time. Over such evolving networks predicting future link between node pairs is an important challenge. To best overcome the intermittent connectivity or disconnection and variable delays that are an integral part of these network it is wise that the problems be studied over temporal domain [70, 75, 103, 137].

Associated with large size also comes the sparsity of large scale networks. The prominent challenge for the significant research interest in link prediction problem stems from the sparsity of the real world large-scale networks [8, 48, 91], implying the existing links at any instant are only a very small fraction of all potential links in the network. Lets consider a network $G(V,E)$ with $V \times V - E$ probable links to choose from. For a dense graph $E = O(V^2) \approx V^2 - c$, c a constant in the range $[1,V]$. Thus, the size of probable link set is $O(1)$, with probability of $O(\frac{1}{c})$ to select a link correctly at random. If existing link set is sparse then the probable link set is $O(V^2)$, with $O(\frac{1}{V^2})$ probability of correct random selection of an future link. Thus, it is challenging to select the future edges accurately in sparse graphs, along with the added dynamic nature of networks.

1.2. MOTIVATION OF THE DISSERTATION

As mentioned earlier, influential metric and knowledge of the availability of links and contacts are critical to success and efficiency of applications. More so, betweenness influential metric is based on dependency, in addition to being a just distance based and being a critical factor in many applications. Efficient solutions to address these problems will intern effect the success of diverse applications over law scale networks. So, in this dissertation we consider proposing efficient solutions for *betweenness centrality* evaluation, *link prediction* over dynamic networks and *contact prediction* over mobile networks.

Further, the usefulness of betweenness centrality lies in its computational cost. Therefore, finding novel and computationally efficient algorithms for exact betweenness centrality evaluation is at the heart of large scale network analysis and our work is devoted to it. More so, the computational challenges are sourced from the underlying network structure. So, we are motivated to leverage network structural characteristics in proposing our novel and efficient algorithms.

Static network representation is the basis of existing research, except a few that consider temporal graphs. Here, link prediction consider evaluating prediction accuracy of new links, e.g. whether two authors will collaborate with each other, or two persons will become friends at a certain time. In relatively stationary network like academic collaboration network (say, DBLP), the assumption may not give worst prediction accuracy, but in dynamic networks, say online social networks, e-mail communications etc., where the link or recurrent-link occurrence is very frequent, it significantly undermines the prediction accuracy. Link prediction, though mostly addresses the possibility of new links between node pairs, it is also significant to address the recurring-links/ interactions over an existing link with examples like: interactions between two existing friends, collaboration between pair of earlier collaborators in an academic network. These time stamped recurring-links press the need for an incremental design model.

Existing techniques lack well defined *time period* for single snapshot of the temporal graph that undermines an efficient and effective prediction model. For example, in co-authorship network (DBLP network) this can be monthly, whereas in email, or online social network communications the time period has to be small enough to capture the underlying dependency (say, hour:minute format) for structural changes. Thus, the prediction model needs to take care of both macroscopic and microscopic temporal evolution and associated correlation over the respective time domains (i.e temporal graphs) for more accurate prediction accuracy. Further, given the observation till time T or a static aggregate snapshot graph or a single snapshot, existing models limit their link prediction for the next time interval (say, $T + 1$).

The conventional link prediction addresses whether two specific nodes indeed interact at a specific time, in contrast the generative models describe evolution on large scale using global characteristics (and latent parameters) without regard to which node interacts with which one. Among the temporal graph based approach,the prediction models either on network structure based [73, 130] or generative model based [68, 79], but what we really need is integration of both for most accurate and elaborate prediction model to determine the set of links at any future time. Further, the dynamic models restrict themselves to the effect of only local neighborhood on link prediction. The challenge is to incorporate the temporally evolving local and global structure in link prediction for improved prediction accuracy.

Further, the dynamic models restrict themselves to the effect of only local neighborhood on link prediction. We consider both local and global structural evolution in link prediction. The challenge here is to incorporate the temporally evolving structure in our dynamic link prediction approach for improved prediction accuracy. In addition, none of the existing work consider rate of evolution in predicting future links.

Keeping aside the underlying dynamics over dynamic networks, the mobile networks and sensor networks are paged with additional factors like intermittent connectivity and variable delays. Thus, exact estimation of the future contacts between node pairs, along with the time at which contacts happen, help us make intelligent forwarding decisions dynamically. As future contact probability of node pairs changes over temporal domain, it emphasizes the need to analyze past contacts to have accurate future contact predictions. Even if it is tempting to assume that possibly contacts between node pairs occur at uniform intervals, it is in reality completely opposite. In reality, contacts (interactions) have non-uniform periods and rate of occurrence of interactions is no longer constant thus, assuming regular contact pattern and associated homogeneous Poisson process model to analyze and predict contacts is not suitable as in [28].

Owing to this non-homogeneity that is the rate occurrence of interactions varying significantly over time, we are motivated to consider the non-homogeneous Poisson process ($NHPP$) model [24, 44, 63, 86, 87] as the basis of our proposed model.

Further, to capture the recurrent nature, the existing models use doubly periodic property considering the applications to be only of short term and log term. Whereas, human mobility and hence contacts over mobile networks are beyond simply doubly periodic, rather they are multi-recurrent. For example, we may go to school (lab) every day, to library every weekend, attend a seminar talk bi-monthly, go to shopping mall every six months, watch movies bi-monthly etc., and so are tasks of data offloading, mobile advertisement etc. are beyond doubly periodic. More so, not only the probability and time of contacts, but also duration, inter-contact period and number of contacts play a major role in efficient and timely completion of applications.

1.3. CONTRIBUTIONS OF THIS DISSERTATION

We address three main problems in this dissertation, namely: exact betweenness centrality evaluation, link prediction over dynamic networks, contact prediction over mobile networks. These problems correspond to the core computational challenges over large scale networks and are critical to applications' success. We present an overview of our contributions in each of the problems in Subsection 1.3.1, Subsection 1.3.2 and Subsection 1.3.3 respectively.

1.3.1. Exact Betweenness Centrality Evaluation. What is exact betweenness centrality indices of network nodes to determine on critical nodes for the network applications?

We propose to leverage the underlying community structure exhibited by large scale networks in proposing the divide and conquer based exact betweenness centrality algorithm. As we leverage the community structure, we thus propose a novel community detection algorithm that leverages network structural features like power law degree distribution, before proposing centrality algorithm. In contrast to considering the entire network for evaluating betweenness centrality metric as in existing exact algorithms, we use community structured network, over which the divide and conquer based centrality algorithm executes, resulting in reduced computational cost. Specifically, the proposed community detection algorithm in this dissertation captures the inherent community structure underlying the large scale network with predefined upper bound on the number and size of communities ($O(\sqrt{|V|})$), where $|V| = n$ is the number of nodes in the network. The resulting communities of the community detection algorithm are used in centrality evaluation algorithm. Thus, we propose a community detection algorithm as an integral part of our proposed centrality evaluation algorithm.

The proposed community detection algorithm (based on agglomerative approach) uses network structural property of right-skewed nature of degree distribution (power-law degree distribution exhibited over large scale networks [31]), coupled with incremental accumulation and semi-local optimal node selection giving computational cost $O(|V|^2 - m|V|k^2)$, where k, $|V|$ and m represent average degree, number of vertices and modularity respectively. For any given network the modularity index quantifies its underlying modular structure.

To proceed with centrality evaluation, the proposed approach inherits the set of communities ($O(\sqrt{|V|})$) resulting from community detection phase. Here, centrality indices are based on shortest path weight measure, where path weight is a reflection of various cost measures. In evaluating exact betweenness centrality, the approach first leverages intra-connections within communities (dense intra-modular), then leverages inter-connections across communities (sparse inter-modular) to compute the centrality indices. Finally, the two centrality scores corresponding to each node are summed up to obtain the final exact betweenness centrality score for that node. The procedure costs $O(|V|^2 + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$ time with $|E| = m' \simeq O(|V|^2)$ edges.

The computational cost incurred due to community detection and betweenness centrality evaluation holds irrespective of graph density and out performs existing exact algorithms. To the best of my knowledge our work is the first work to leverage structural properties in community detection and exact betweenness centrality evaluation over large scale networks. We validate our approach over four large scale network data sets relative to benchmarked approaches.

1.3.2. Link Prediction over Dynamic Networks. How to know the link occurrences ahead of time over dynamic networks, so that we can have efficient application services that are minimally affected by underlying network dynamics?

We propose a probabilistic model, with temporal graph as its basis for link prediction over dynamic networks. Here, the link prediction probability is the transition probabilities in the corresponding Markovian transition matrix. We consider the state space of our Markov model to reflect the past s states during the evolution process and the model is built bottom up. Nonetheless, we view the temporal evolution of links is a cascaded process, with nodes acting as controller or potential barrier effecting which subsequent links will appear; we use this concept to evaluate prediction probability for both source and destination nodes for future link prediction.

Further, we propose integration of different timescales over network evolution for most accurate prediction model. Thus, the analysis considers both local and global network structure evolution with corresponding two-way temporal dimension; that is local structural evolution (link evolution) is analyzed over microscopic time domain, whereas global structural evolution (evolution of clusters) is analyzed over macroscopic time domain. Since, evolution of link gives rise to evolution of underlying communities/clusters, we have an intrinsically dynamic community structure. Besides, we also consider temporally correlated evolution, rate of evolution and evolution dynamics over the dynamic (temporal) network for model construction. Thus, our model is able to capture very distinct evolution profiles during the observation period.

We predict future links over both microscopic and macroscopic time domain. Further, we also investigate prediction success beyond simply next time interval. Finally, we experimentally evaluate the prediction accuracy of our approach using four real world data sets with three dynamic network e.g. Twitter, Enron E-mail, Facebook and a lesser dynamic structure such as DBLP, and compare the accuracy with existing baseline static and dynamic measures. We use ROC-curve (Receiver Operating Characteristic), AUC- value (Area Under ROC -curve) metrics to reflect the model accuracy, both graphically and quantitatively.

To the best of our knowledge ours is the first work to consider Markovian matrix model considering temporal graphs with integration of fine-grained and coarse-grained temporal analysis for best accuracy. Further, we incorporate corresponding local and global structural evolution, correlated evolution, and evolution dynamics at the start and end node of a future link from the intuition of cascaded evolution process.

1.3.3. Contact Prediction over Mobile Networks. How to determine future contacts between node pairs, specifically the number and inter-contact period of future contacts associated with nodes, to efficiently perform tasks over intermittently connected mobile networks?

To leverage the multiple recurrent and dependent nature of events, we propose a cascaded non-homogeneous Poisson process (cascaded $NHPP$) model [45, 84, 114] in analyzing the history of interactions. Unlike many existing prediction models based on static (say, regression models) or discrete time models, we consider continuous time stochastic model to analyze the recurrent, dependent contact pattern and predict the future traits in contact patterns, as this is the most natural way of describing discrete contacts, occurring over continuous time.

In order to capture the multiple recurrent nature of contacts into our proposed stochastic model, we propose using sinusoidal function as the intensity function. Further, we also capture the direct dependency among recurrent contacts. As a result, we propose cascaded non-homogeneous Poisson process model to replicate interactions and predict future contacts.

We also consider the possibility of random occurring contact events (without any periodicity) and propose to intercept these contacts and predict their future occurrences. The Markov modulated Poisson process is proposed for this case. Thus, our contact prediction integrates both proposed models to address prediction accuracy. We have considered simulation only for the recurrent contact patterns, and hence prediction accuracy is validated only over cascaded non-homogeneous Poisson process. The simulation while considering random contact patterns into account and hence the prediction accuracy of the integrated model is still going on.

Experimentally, we analyze number of contacts relative to an user and over all users, over different time intervals. We also analyze future contact time per user and per user pair and conclude more accuracy in later case. Then, we predict aggregate inter-contact time between node pairs over different time intervals and compare their accuracy. To evaluate the performance and validate our model, we use MIT reality mining dataset [108], and compare our model with doubly recurrent and homogeneous Poisson process model obtaining better prediction accuracy, Further, we consider aggregate pairwise inter-contact period over the whole observation period. To validate our model accuracy we use empirical measures like, F score, ROC-curve (Receiver Operating Characteristic), AUC- value (Area Under ROC -curve).

Part of our work is published in [37, 38, 39, 40, 41, 42, 43].

An overall graphical representation of the problems we consider, and the motivation behind them is presented in Figure 1.1.

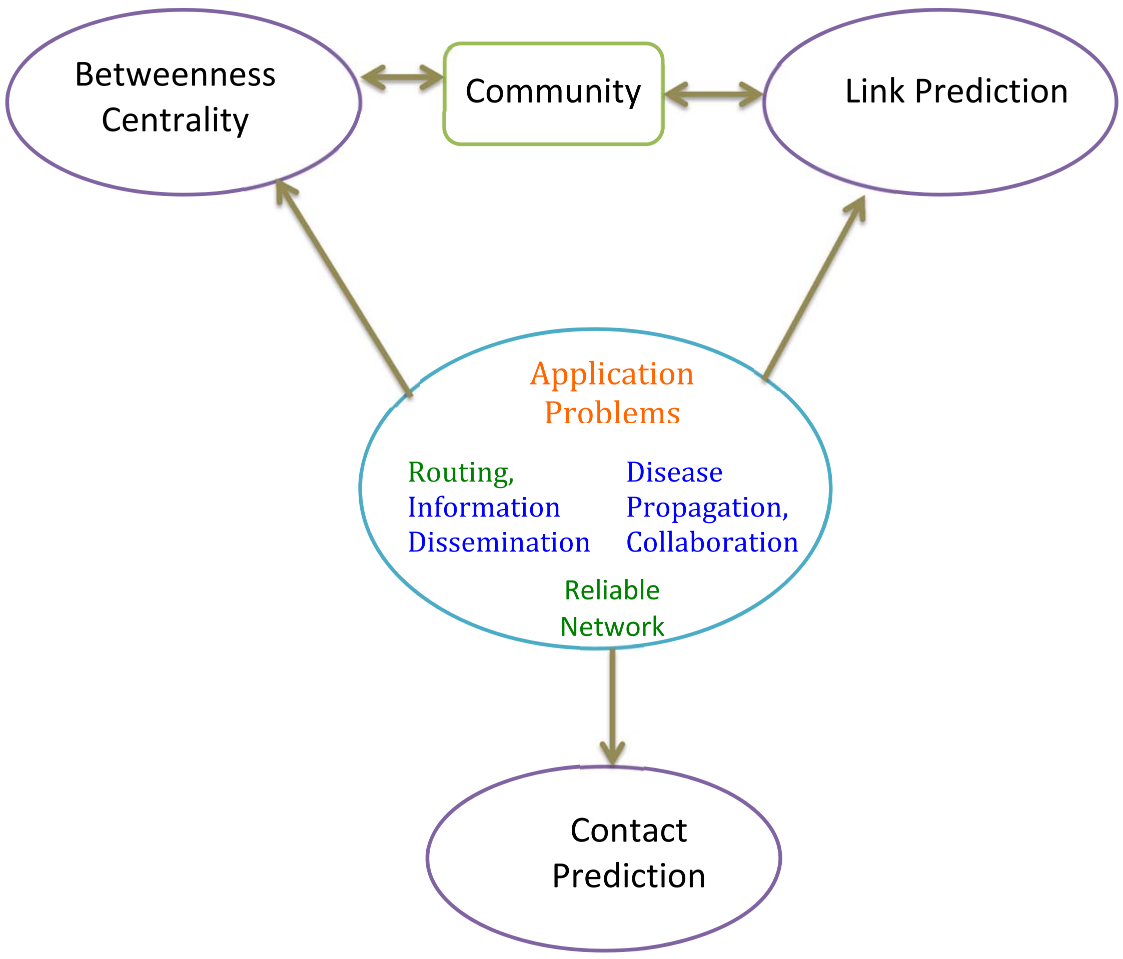


Figure 1.1. A Schematic View of Our Problems and Motivation

1.4. ORGANIZATION OF THIS DISSERTATION

The remainder of this dissertation is organized as follows.

Section Two reviews the related work in betweenness centrality evaluation, community detection, link prediction and contact prediction problem. In the centrality evaluation subsection we start with the various existing centrality metrics, then discuss the existing exact and approximation algorithms for betweenness centrality evaluation. As communities are an integral part of our proposed centrality algorithm and we propose a novel algorithm for community detection, we also discuss existing algorithms on community detection.

Section Three presents the divide and conquer based deterministic (exact) betweenness centrality evaluation algorithm. It emphasizes the use of underlying community structure and the sparse inter-community edges as the key points for reduced computational cost. It also provides the network structure based community detection algorithm. We present the correctness of our proposed algorithm, the improvement in computational cost for both algorithms and their experimental validation in this Section.

Section Four presents the temporal network structure based probabilistic link prediction model over dynamic networks. It presents the temporal model with multiple timelines to better reflect the network evolution process and discuss its integration in selecting start and end nodes, and the link occurrence probabilities. We then discuss the procedure for populating the Markovian matrix and considering links for subsequent time periods. We also present information theoretic prediction procedure and then derive upper bound on number of history states needed for the prediction model. Finally, the detailed experimental validation of the approach is presented.

Section Five presents the cascaded non-homogeneous Poisson process ($NHPP$) model that leverage multi-recurrent contact pattern to predict future contacts over mobile networks. It also integrates the Markov modulated Poisson process to capture the random (non-periodic) events. The contact prediction model is finally an integration of both models. We then present the parameter estimation and experimental validation for our proposed recurrent model.

Section Six Summarizes our main contributions and discuss future work of this dissertation.

2. LITERATURE REVIEW

Large scale networks with its diverse applications is fast becoming an integral part of our daily lives. More so, advances in the Internet and mobile technology, along with reduced cost of smart devices, are now generating omnipresent smart and mobile environments. Over these networks, the application problems like routing, information dissemination, viral marketing, maintaining network reliability, addressing diffusion minimization and maximization problems over temporal domain to disease propagation, though seem significantly interesting, the underlying research challenges are quite complex; more so, over time-varying or temporal networks. To best overcome the challenges for efficient and effective realization of these applications, there is considerable research attempt to better analyze these networks. For example, modeling network evolution [95], identifying influential nodes, effective routing strategies [36, 50, 89], improving resilience [105], security[133], privacy [18, 66], and clustering of (social) networks [16, 76, 112].

Core to the complexity and challenges in successful realization of these applications, are the challenges in the underlying large-scale networks, over which the applications are made to be a reality. Addressing these challenges are part of network analysis problem. As the applications are realized over networks, the network structural challenges for example, centrality (more so, betweenness centrality), occurrence of links and contacts between node pairs are core to the success of these applications. Though there are several algorithms and approaches that address these challenges, and attempt to make the applications more effective and efficient, most of them lack to integrate network structural characteristics and temporal characteristics in their design. Considering the dynamic or time-varying nature of the underlying network, and the structural changes here with, it is obvious to consider these aspects in the design of efficient, effective and successful framework (strategy) to address the intrinsic challenges in network. In this dissertation, we propose to integrate network structural and temporal characteristics to design novel, effective and efficient solutions to address these challenges.

Underlying every network lies the associated network graph. So it is imperative that network graphs form the backbone of the analysis process. In this dissertation we represent the underlying network graph of a network as $G(V, E)$, where V is the set of nodes and E is the set of edges with number of nodes and edges are given by $|V| = n$ and $|E| = m'$ respectively.

The remainder of our discussion here is organized as follows: Subsection 2.1 reviews the various centrality measures concisely and then focus on the existing betweenness centrality evaluation strategies. As we leverage the community structure underlying large scale networks, for our centrality evaluation, and propose a novel and computationally efficient community detection algorithm, we also discuss the existing community detection algorithms as part of it. Subsection 2.2 describes related work in link prediction. As the underlying network is temporal (time-varying) in nature, and we incorporate this characteristic for better prediction accuracy, we start with a description of temporal networks and their representation. Then we discuss the existing class of link prediction models, along with the specific versions of link prediction problems considered in literature. Subsection 2.3 describes the approaches for contact prediction over mobile networks across diverse application domains, the existing models and also define some important models that are the basis behind the contact prediction models. Finally, a summary of this Section is presented in Subsection 2.4.

2.1. CENTRALITY: DEFINITION AND CLASSIFICATION

The key to maintaining a connected and reliable (safe) network is addressing its vulnerable nodes. Vulnerability is proportional to the centrality or influential index of any node. In fact, of the most important aspect of the analysis of large scale networks is the computation of *centrality indices* that measure the importance or influence of a node or edge [17]. Depending on the application several measures of influential metrics are proposed in the literature.

- *Degree centrality*: The degree centrality measures the total number of contacts incident upon a node, with importance in the design of forwarding algorithms. Evaluation of degree centrality is of the order of $O(|V|^2)$ over an underlying network graph $G(V,E)$, that reduces to the order of $O(|E|)$ for sparse graphs. Concisely we can say that degree centrality reflects importance of a node from local perspective, as it only takes into account number of incident (direct) edges or adjacent nodes. In some application domain degree centrality has a more meaningful implication. For example, in a co-citation network the paper ranked higher with degree centrality index has many papers citing it; indicating its influence or prestige in that specific category. But, it may not reflect the papers importance across specializations (global influence), unless papers from other specialization also cite it. In general a node with high degree tend to communicate with many other nodes. In power law degree distribution that is a case with few nodes. But when high degree is approximately equal to average degree, communications are more evenly distributed. It does not consider indirect contacts and their influence in the network. For example, two nodes in a network with equal degree centrality may not be equally important considering the network topology, similar to the scenario where someone may have small but really important neighbors). To accommodate this eigenvector centrality come up. In addition, only adjacencies are counted not the distance of the adjacent nodes; so, in a weighted graph any change in distance of adjacent nodes is not reflected in the centrality metric. To accommodate this closeness centrality has come up.

- *Closeness centrality* [19]: The closeness centrality measures how close a vertex is, to all other vertices in the graph and has applications in determining broadcast points. The closeness measure using the Single Source Shortest Path ($SSSP$)

based exact algorithm of $Dijkstra$ [34] over $|V|$ source nodes is of the order of $O(|V||E| + |V|^2 log|V|)$. A drawback in computing closeness centrality of a node relative to the component in which the node lies in a disconnected network is that a node in a smaller component will have higher centrality index than a node in a larger component with much larger number of nodes, even if the node in the later case is closest to all the nodes in that component. It is because of the larger number of nodes in the larger component and when the nodes are scattered across that component the distance among nodes is larger than smaller number of nodes condensed inside a smaller component. A node with higher closeness centrality index is closer to all nodes than the nodes with lesser closeness centrality indices. For both weighted and un-weighted networks the node with highest closeness centrality has significant importance in many application domain for example which one needs to be a broadcast center, or where information needs to be kept to be accessed by all nodes with least delay (where edge weights reflect delay/traffic). In other words these nodes can spread information very productively through the network. A node with high closeness is usually restricted to larger component of the network. In case of disconnected (components of a) network the closeness centrality measure to becomes 0. In an unweighted n/w a node with highest degree centrality may not be the node with highest closeness centrality index.

- *Eigenvector centrality* [110]: The eigenvector centrality measures the centrality of a vertex as the highest value from the normalized eigenvector that corresponds to the principle eigenvalue of the adjacency matrix for underlying application domain. Existing algorithms for eigenvector centrality indices employ power method that have a computational complexity of $O(|V| + |E|)$, although actual time depends on the spectral gap of the adjacency matrix.

- *Betweenness centrality* [56]: The betweenness centrality measures the importance of a node from the perspective of being located in the shortest paths that connect other peer nodes, thus captures both its load and significance or, critical factor in information flow over the network. In essence betweenness centrality index quantifies the number of times a node acts as a bridge along shortest paths between other nodes in the network w.r.t. all possible shortest paths between all pairs nodes in the network. Formally *betweenness centrality*, a relative measure of a node v is the percentage of the number of shortest paths that node v is part of between all pairs of nodes, over all possible shortest paths between them. This metric is of significance in a wide range of application domains: the node through which maximum traffic or flow go through (rate of flow uniform across all geodesic paths), the node whose vulnerability/safety/failure affects the most in a global scale (in contrast to the mostly localized disruption in case of degree centrality based central node). Thus, the node becomes the most critical. The nodes with higher betweenness centrality index are considered to be influential as they help spread information across networks and acts as connectors. In a network, a vertex can have low degree, with neighbors too having low degree, even have higher mean distance to other nodes but still can have higher betweenness centrality value

- *Stress centrality* [120]: The stress centrality measures the stress index of a node by computing number of times the node being located in all pairs of least stress paths over a network and is known as another variant of betweenness centrality.

Of the above approaches, the closeness, betweenness and stress centrality measures are distance based metric. Further, though closeness centrality is directly based on the shortest path algorithms and their computational cost, the betweenness centrality and stress centrality measures have the added dependency factor that compromises the computational cost purely based on the performance of shortest path algorithms. Of these centrality metric, research over betweenness centrality evaluation has garnered more interest due to its wide range of applicability and the higher computational cost incurred due to indirect dependency. The attempt is to propose approaches to reduce the computational cost i.e. to compute the metric efficiently, so that it can be effectively used over wide range of application domains. The approach and computation cost for betweenness centrality evaluation, applies equally to the stress centrality evaluation too.

2.1.1. Betweenness Centrality Evaluation. We are specifically interested in the *betweenness centrality* (BC), a relative measure of occurrence of node v over all pairs of shortest paths. The usefulness of betweenness centrality lies in its computational cost. Therefore, finding novel and efficient algorithms for centrality evaluation is at the heart of large scale network analysis and hence much research has been devoted to the design and analysis of efficient algorithms for evaluating the betweenness centrality which falls into the distance based centrality measures. Thus, algorithms used for shortest path evaluation are equally applicable in computing distance based centrality metrics. But, the issue lies in the computational cost incurred in exact use of shortest path algorithms for betweenness centrality evaluation. Using either of the Floyd-Warshall algorithm [34] or the Dijkstra's algorithm [34] (including BFS algorithm for unweighted graphs) in evaluating betweenness centrality incurs computational cost of $O(|V|^3)$, and space utilization of $O(|V|^2)$, where $|V|$ is the number of nodes in the network. The $O(|V|^3)$ computational cost is incurred due to the explicit addition of centrality indices in direct use of existing shortest path algorithm for betweenness centrality evaluation. The work in [15] defines *geodetic semiring*, a closed semiring generalization for shortest path problems, which results in $\theta(|V|^3)$ algorithm for betweenness centrality evaluation by augmenting the Floyd/ Warshall algorithm for all pairs shortest paths problem with path counting.

Motivated by the need for efficient computation of betweenness centrality, Brandes [21] proposed an exact algorithm based on Dijkstra's shortest path algorithm that takes $O(|V||E| + |V|^2 log|V|)$ time over weighted networks and $O(|V||E|)$ time for unweighted networks. $O(|V|+|E)$ memory space is required in both cases. Baglioni et al. [10] proposed an improved version of Brandes algorithm to achieve a speed up of 2-to-5 times relative to the Brandes approach.

In order to reduce computational cost approximate betweenness measure is proposed by Bader et al. [9], Geisberger et al. [60], Riondato et al. [113] that use adaptive sampling technique, linear and bisection scaling, $VC-$dimension. The drawback is that it is valid for only high centrality nodes with certain probability bound and depends on sample/pivot selection, number of iteration etc. As exact evaluation of BC metric is critical to applications and is computationally challenging, we focus on its exact evaluation.

The summary of the existing centrality evaluation approaches in literature can be presented in a graphical form as in Figure 2.1.

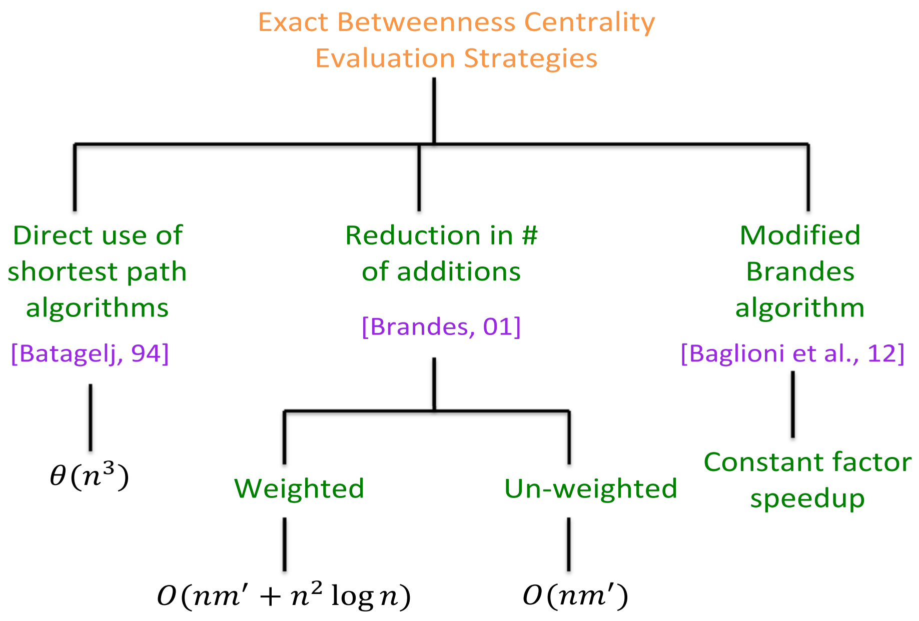

Exact Betweenness Centrality Evaluation Strategies
Direct use of shortest path algorithms
[Batagelj, 94]
$\theta(n^3)$
Reduction in # of additions
[Brandes, 01]
Weighted
$O(nm' + n^2 \log n)$
Un-weighted
$O(nm')$
Modified Brandes algorithm
[Baglioni et al., 12]
Constant factor speedup

Figure 2.1. Schematic View of Existing BC Evaluation Schemes

2.1.2. Community Detection Strategies. A *community* is a set of nodes (network elements) that are bonded together with higher connectivity than these nodes connectivity with rest of the network. Determining inherent communities (clusters or modules) in large scale networks is one of the fundamental structural problems and is widely addressed by the research community [7, 62, 99, 125], reflecting dense intra-module and sparse-inter module connection. In this dissertation, to refer to a densely connected group of nodes, I use cluster or community or module interchangeably.

There are three distinct strategies for community evaluation (detection), i.e., the *agglomerative*, the *devisive* and the *spectral* approach. In the *agglomerative hierarchical* clustering algorithms, the procedure starts with a set of nodes and gradually accumulated with other nodes to form suitable community. The techniques vary with how the set of nodes are selected initially and how they are accumulated with other nodes to form/detect communities. In the *divisive* based clustering algorithms, the procedure starts with the whole (complete) network at hand and then partitions it to reach at the suitable communities. The procedures vary by how the network is partitioned or divided in each stage. Usually the betweenness centrality index of a node is used to help in partitioning task. More so, a quantitative measure called *modularity* is used to reflect/ measure the bindingness or the bonding of the nodes in a community and help establish community boundaries. In order to evaluate the effectiveness of community structure the most widely used measure is *modularity* [30, 102], which is defined as $Q = \sum_i (\frac{e_{ii}}{2m'} - (\frac{a_i}{2m'})^2)$, where e_{ii} denotes the set of edges within community i, $\frac{a_i}{2m'}$ denotes the expected fraction of edges connected to community i in case of a random graph, over an undirected-unweighted network.

The larger the modularity, the better the corresponding partition or community. In the *spectral* community formation schemes, the eigenvectors corresponding to the weighted connectivity matrix of the network reflects the underlying communities. Below we present some significant contributions from the literature in community formation strategies.

Since, finding optimal community structure is known to be NP-hard [59], heuristic and approximation solutions have been proposed based on agglomerative hierarchical clustering [30, 100], divisive clustering based on betweenness centrality [102], or spectral partitioning [135]. The two agglomerative clustering costs $O(dm' logn)$ time, d as the depth of the hierarchical clustering, and $O(n^2)$ time respectively; divisive clustering costs $O(n^3)$ time; whereas spectral partitioning costs $O(n)$ time, all these approaches are over very sparse graphs ($m' \simeq n$, with $|V| = n$ nodes and $|E| = m'$ edges).

A graphical presentation of the classification of these existing approaches is presented in 2.2.

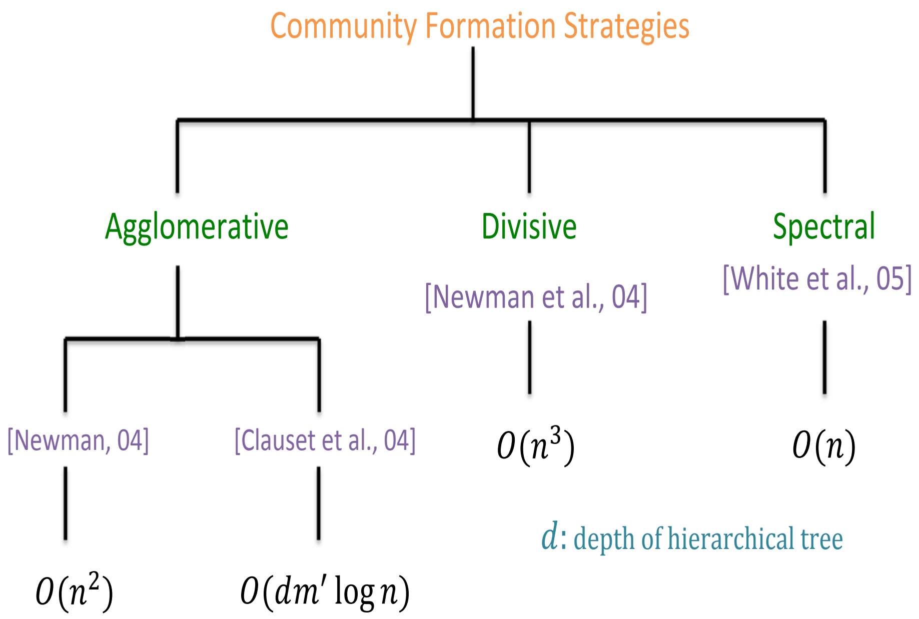


Figure 2.2. Schematic View of Existing Community Evaluation Schemes

2.2. COMPUTING FUTURE LINKS: NETWORK MODELS

Large-scale networks are inherently dynamic and and ubiquitous in nature. The dynamic nature of these networks as a result of their structural changes over time is represented using time varying graphs, and is viewed as a sequence of discrete snapshots of an evolving graph [2, 70, 75, 103]. Change in network structure results from the inclusion of new or exclusion of existing nodes, and presence or absence of links over time. In reality the exclusion/ deletion is a rare event whereas, inclusion of new nodes is prominent, resulting in denser network over time. Over such evolving networks predicting future link between node pairs is an important challenge.

A natural strategy to analyze this structural evolution in dynamic networks is to observe it over the temporal dimension. The study and modeling of the dynamics in the network structure are a significant research domain and is addressed in many research papers [13, 23, 82, 85, 122]. Core to these structural evolution is the evolution of links or formation of interactions or associations over the network that implies not only the presence or absence of a node but also its influence. This has wide range of applications including but not limited to biology, medicine, social and opportunistic networks, to mention a few, generating massive interest in link prediction problem. Irrespective of the diverse application domains, common to all these networks is the representation of network evolution using time varying graphs [70, 75, 103, 137]. Though, network evolution or dynamics and hence the underlying time varying graph is caused due to mobility of the participating nodes relative to time in vehicular, opportunistic and mobile-social networks, in most large scale networks mobility is not the influencing factor. In biological, social, and economic networks, it is the underlying dynamics of interactions that result in system dynamics.

As most of the existing link prediction strategies are actually based on the aggregated graph of the underlying dynamic network, we discuss this stationary representation and then its variants to reflect the network. Still it is unable to reflect the network dynamics, hence not suitable to be considered as the basis for prediction models over dynamic networks. Thus, comes the temporal or time-varying network (graph) consideration. Before going into the details of existing prediction models, we discuss the two distinct interpretation of the underlying dynamic network, i.e., *static* and *temporal* network graphs.

2.2.1. Stationary and Temporal Interpretation of Dynamic Networks. Stationary graphs assume appearance of all edges and existence of all nodes at the same time, do not capture significant temporal characteristics such as duration of exchange, inter-contact time, recurrent contacts and time order of contacts along a path. In contrast the underlying large scale networks are inherently dynamic, i.e., they vary over time. By considering static interpretation of these inherently dynamic networks, we loose the underlying behavior over the network. We explain it below through a simple example. Let us consider a four nodes A, B, C, D. We present their interaction time and duration over a time axis in Figure 2.3.

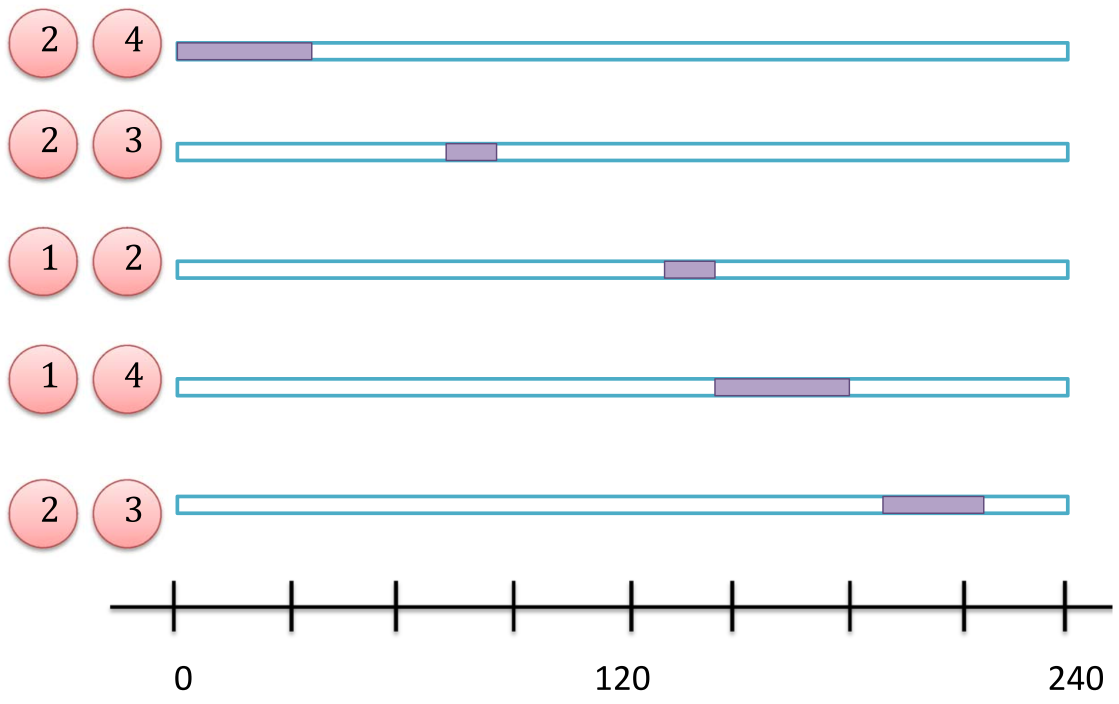


Figure 2.3. Time of Events

The corresponding static graph representation of a dynamic graph aggregates the contacts over the time and it can either be unweighted or weighted. Considering our example the widely used unweighted static aggregated graph is as shown in the first graph of Figure 2.4. The second graph represents the weighted static aggregated graph, where edge weights represent the frequency of contacts between node pairs.

In yet other representation of weighted aggregated static graphs the edges are weighted according to the duration of interaction and order of interaction duration over any single pair of nodes. We present the corresponding graphs associated with our example in the following Figure 2.5. The graph in the left shows the edges annotated with edge weights whereas, the second graph shows how nodes $2, 3$ interacted over two distinct intervals.

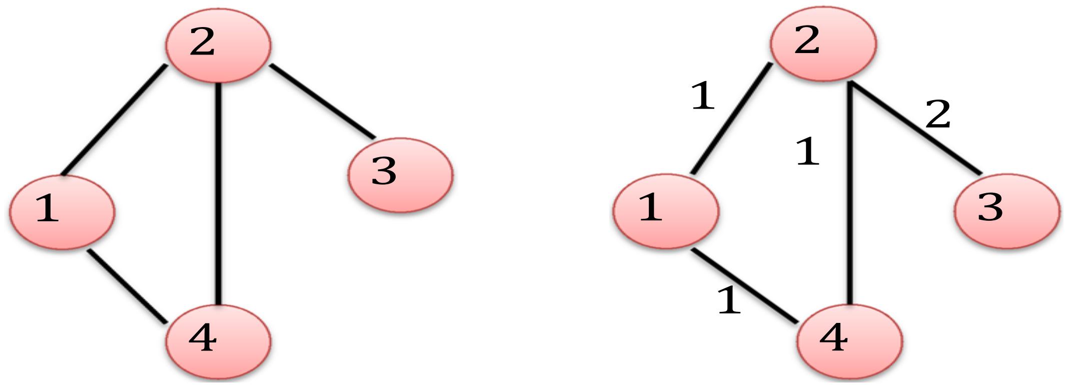


Figure 2.4. Static Aggregated Graph: Unweighted and Weighted According to Frequency of Contacts

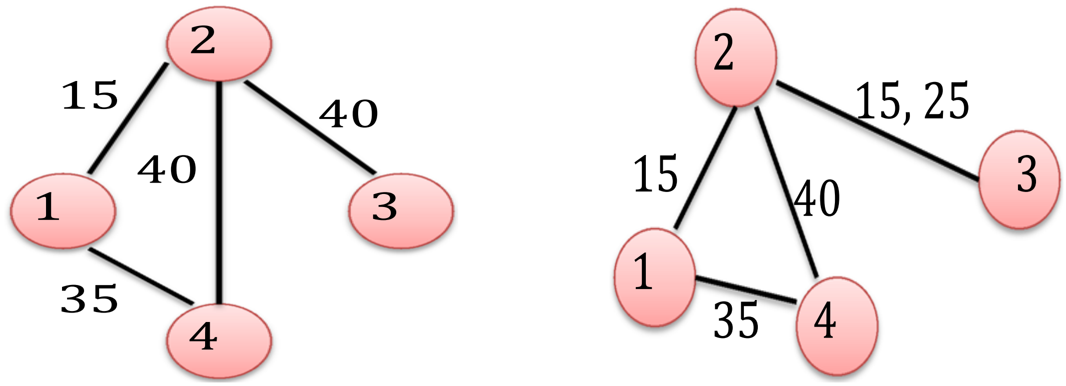


Figure 2.5. Static Aggregated Graph: Weighted According to Contact Duration and Order of Contacts

It is evident that the above static representation overestimates the contacts and does not capture the structural evolution (in terms of edge creation/interaction) and order of events/interaction in a global scale.

This necessitates the formation of *Temporal graphs* or *Time varying graphs* (or dynamic graphs) instead of static graphs to represent networks and to better reflect its underlying dynamics. It is first proposed by [75].

In order to incorporate temporal parameter existing approaches discretize the time domain into a sequence of T time slots, each with duration/window size w. The instances of the resulting graph called *Time series graph* or *Time varying graph* or *Dynamic graph* or *Temporal graph* [75, 81, 104, 128], shown in Figure 2.6. It shows the evolution of the network structure in the underlying graph over discrete and consecutive time intervals. The Time series graph shown here is based on the example we considered earlier.

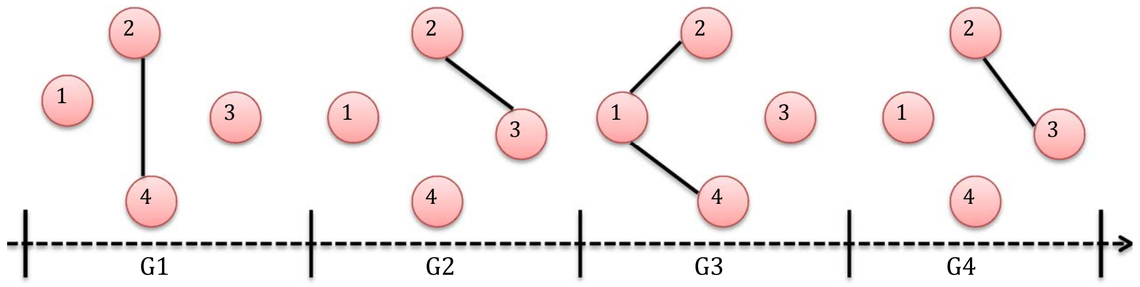


Figure 2.6. Time Series Graph

In our proposed prediction model, we consider the underlying temporal network as an integral part.

2.2.2. Link Prediction. Link prediction, though mostly addresses the possibility of new links between node pairs, it is also significant to address the recurring-links/ interactions over an existing link with examples like: interactions between two existing friends, collaboration between pair of earlier collaborators in an academic network. These time stamped recurring-links press the need for an incremental design model. As we mentioned earlier, the set of links at any future time are crucial to success of many widely used applications, for example: next collaboration between authors, new friendship in social networks, market trends, movies and songs a user will listen to, hidden links in anti-social activities, and so on. This has led to significant research interest in accurate link prediction strategy [1, 4, 5, 20, 88, 90, 111, 119, 129]. Accurate link prediction has also applications in efficient routing in opportunistic, delay tolerant and mobile-social networks in order to minimize delay and maximize coverage area [25, 26, 109].

Most of the literature baring a few consider static networks or static interpretation of a time-varying networks, where the prediction probability of links at time T is derived over a single snapshot of the network. The snapshot is a single aggregated graph representation of all temporal graphs from start of the observation till time $T-1$ or a single snapshot itself [1, 5, 20, 88, 90, 111, 119]. All these approaches incorporate network structure in some way or other. In fact, preferential attachment model [13] is one prominent network structure based link prediction model that considers connecting only pairs of high degree nodes with higher probability. Another approach consider the statistical relational models for link prediction [61, 69, 129], these are restricted to relational models and are designed only for static networks. Yet another approach consider Markov chains for link prediction [117] over website, but assume Markov property that limits it from capturing structural evolution, temporal dependency and thus, making it un-suitable over dynamic networks.

The underlying large scale networks and their behavior are intrinsically dynamic. Examples of some such dynamic instances over networks are, friendship or amount (degree) of interaction among individuals in a social network is usually transient and recurrent [80, 106]; collaboration among academic scholars changes with time [123]; different intellectual tasks activate different voxels in brain causing different neural activity [54, 131]; communication among agents in a telecommunication system is usually bursty and fluctuate over time [12, 97, 124], to mention a few. Increasing availability of microscopic data over these intrinsically dynamic networks along with data from sensors, online resource networks where the participating node or its participation may be changing (e.g. twitter, inter-organizational collaboration), paves the way for temporal network and its analysis.

A natural strategy to analyze this structural evolution in dynamic networks is to observe it over the temporal dimension. The study and modeling of the dynamics in the network structure are a significant research domain and is addressed in many research papers [13, 23, 82, 85, 122]. Core to these structural evolution is the evolution of links or formation of interactions or associations over the network that implies not only the presence or absence of a node but also its influence. This has wide range of applications including but not limited to biology, medicine, social and opportunistic networks, to mention a few, generating massive interest in link prediction problem. Irrespective of the diverse application domains, common to all these networks is the representation of network evolution using time varying graphs [70, 75, 103, 137]. Though, network evolution or dynamics and hence the underlying time varying graph is caused due to mobility of the participating nodes relative to time in vehicular, opportunistic and mobile-social networks, in most large scale networks mobility is not the influencing factor. In biological, social, and economic networks, it is the underlying dynamics of interactions that result in system dynamics.

Among the very few dynamic approaches, only [73, 116, 130] consider time series data. Both models consider temporal graphs from start of observation till time T to evaluate the prediction probability of links for time $T + 1$. In [73] the authors consider temporal graphs over time periods of one month and frequency of communication between node pairs as the corresponding edge weight. Here the time series is integrated with a structural feature of static network i.e. common neighbors, to obtain the temporal prediction model, similar is the case for the model in [130]. Whereas, in [116] the authors integrate time series with common neighbors and last occurrence of a link for their temporal prediction model and use real world sensor network and co-authorship network. These conventional link prediction strategies focus on finding links over only the immediately next time slot/ time period. The conventional link prediction addresses whether two specific nodes indeed interact at a specific time, in contrast the generative models describe evolution on large scale using global characteristics (and latent parameters) without regard to which node interacts with which one.

A schematic representation of the class of existing link prediction strategies are given in Figure 2.7.

2.3. CONTACT PREDICTION OVER MOBILE NETWORKS

As discussed earlier, over dynamic pervasive networks the solution to contact prediction or link prediction and its variants like mobility prediction, contact time prediction, inter-contact time prediction are estimated and used to propose efficient techniques for routing, information dissemination, smart management etc. In literature, in order to predict the

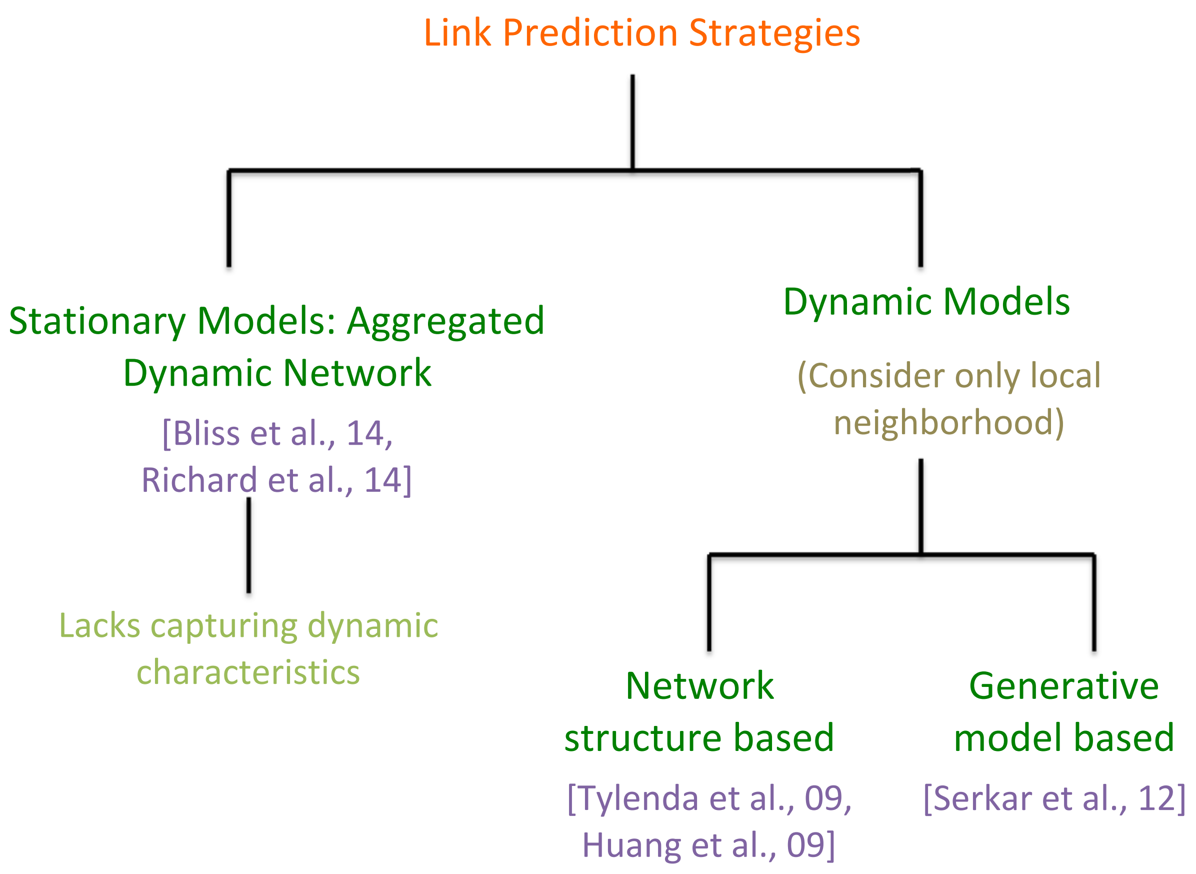

Link Prediction Strategies
Stationary Models: Aggregated Dynamic Network
[Bliss et al., 14, Richard et al., 14]
Lacks capturing dynamic characteristics
Dynamic Models
(Consider only local neighborhood)
Network structure based
[Tylenda et al., 09, Huang et al., 09]
Generative model based
[Serkar et al., 12]

Figure 2.7. Schematic Overview of Link Prediction Strategies

above, human walk traces are analyzed to discover several significant statistical patterns of human mobility like establishing the individuals'/mobile devices' return to a few highly frequented locations [64, 83], which implies that they come in contact with other mobile devices frequenting there. Analysis of contact patterns, inter-contact period over DTN [32] is used to show their exponential and power law distribution, to the analysis of contact time using the fact that topology expands and shrinks with time, thus influencing connection times and opportunities between participants. In contact processes over dynamic networks, the inter-contact time between any pair of nodes is exponentially distributed as is supported through numerical simulations based on synthetic mobility models. In [65] the authors used the Random Waypoint mobility simulation to prove that the inter-contact time is mutually independent and exponentially distributed.

2.3.1. Contact Prediction Models. The links (or interactions or contacts or edges) over a dynamic network are recurrent, that is exhibit some pattern over temporal dimension, thus it is worth using this underlying pattern of interaction to predict future interactions. A common notion to model the inherently regular contact pattern is to use *Poisson* process models [114], that is also known as counting process model as is used to count the number of contacts in any time interval. Assuming the contact patterns to be regular or homogeneous, Ciobanu et al., [28] proposed a simple homogeneous Poisson process model (HPP) to predict future contacts.

In contrast Chaintreau et al. [26] considered the fact that there might be heterogeneity about the inter-contact time distributions of different pairs of devices. Further the research work by Conan et al. [32] investigated three real human contact data sets and found that most of the pair-wise inter-contact time distributions tend to well fit log-normal curves, while some distributions may also fit the exponential distributions. In [118] the authors found that the inter-contact times exhibit strong time-of-day non-stationary property, whereas the cyclic rhythm of the inter-contact time distribution is found in [58]. This generates curiosity to explore models reflecting time non-stationarity, varying contact rate, correlation among contacts and periodic rhythms of contact patterns, hence predicting future contacts.

The regularity of interaction pattern in designing prediction model for future interactions is not suitable as the interactions have non-uniform periods and rate of occurrence of interactions is no longer constant. Owing to this non-homogeneity that is the rate occurrence of interactions varying significantly over time, several variants of non-homogeneous Poisson process models ($NHPP$) are considered [24, 63]. Further, to capture the recurrent nature, the existing models use doubly periodic property considering the applications to be only of short term and log term. One such doubly periodic non-homogeneous Poisson process model is proposed over hurricane data [92], though it is also applicable to seismic activity. In a similar line of study the authors in [93, 138] used the concept of doubly periodic contacts as short time interval (daily) and long time interval (weekly) contact events. Thus, in order to study the pairwise contacts the authors use non-homogeneous Poisson process along with homogeneous Poisson process. The authors in [93, 138] consider doubly periodic stochastic double chain hidden Markov model to capture the recurrent pattern in analyzing communication/contact over node pairs; for example, daily and weekly recurring communication pattern, though over distinct application domain, i.e. over email and MIT [108], UCSD [94] data sets respectively.

The class of existing approaches can be represented schematically as in Figure 2.8.

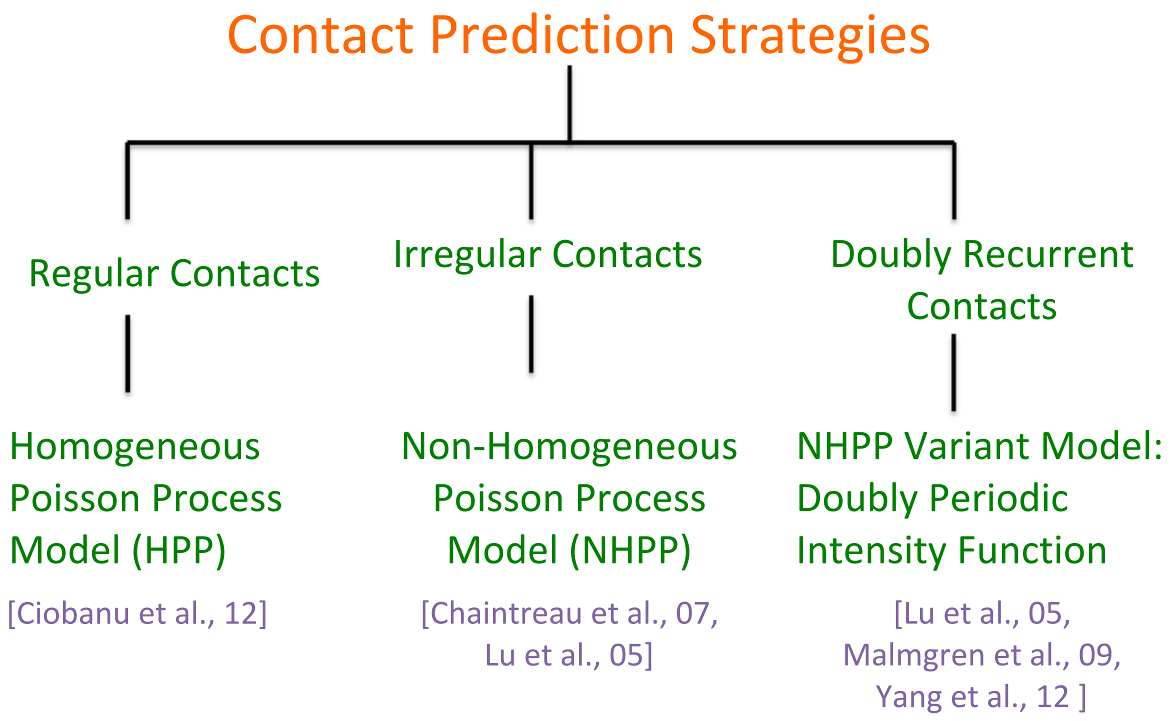

Contact Prediction Strategies
Regular Contacts
Irregular Contacts
Doubly Recurrent Contacts
Homogeneous Poisson Process Model (HPP)
[Ciobanu et al., 12]
Non-Homogeneous Poisson Process Model (NHPP)
[Chaintreau et al., 07, Lu et al., 05]
NHPP Variant Model: Doubly Periodic Intensity Function
[Lu et al., 05, Malmgren et al., 09, Yang et al., 12]

Figure 2.8. Schematic Overview of Contact Pattern Prediction Strategies

Since, HPP and $NHPP$ models act as the basis behind existing prediction models, and a variant of $NHPP$ model that is the cascaded $NHPP$ model acts as the basis of our proposed model, we here define these models before concluding this Section.

2.3.2. Basic Model Definitions.

Definition 2.3.1 *The* Homogeneous Poisson *process (HPP) model has independent and identically distributed inter-contact time as per the exponential distribution with parameter/contact rate λ. The number of contacts between time interval $[0,t)$ given by the probability mass function $P\{N(t) = k\} = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$ and the expected number of contacts during the same period is given by $E[N(t)] = \lambda t$. Incase of HPP the contact rate or the intensity function remain homogeneous or constant over intervals i.e. $\lambda(t) = \lambda, \forall t \geq 0$. In realty the contact rate or the intensity function instead is a function of time, represented as $\lambda(t)$, hence non-homogeneous.*

Definition 2.3.2 *The* non-homogeneous Poisson Process $(NHPP)$ *[114] with $\{N(t) : t \geq 0\}$ is a counting process with intensity function $\lambda(t), t \geq 0$ as an integrable function of time, where $N(t)$ be the number of interactions taking place in the interval $[0,t)$. The mean value function of the NHPP is defined as $\mu(t) \equiv E[N(t)] = \int_0^t \lambda(q)dq$, $\forall t \geq 0$. Further, a non-homogeneous Poisson process is a Markov process, with $0 \leq s < t$, $N(t) - N(s)$ has a Poisson distribution with mean $m(t) - m(s) = \int_s^t \lambda(q)dq$. We propose to use the intensity function $(\lambda(t))$ of a NHPP to predict the occurrence of interactions. As we study the interactions over temporal domain, it is natural to visualize the interaction set as time stamp of their occurrence along the time axis.*

Definition 2.3.3 *A cascaded non-homogeneous Poisson process model, is a two state point process model initially proposed by Malmgren et al. [93]. During the normal state of operation the events occur at rate $\lambda = \lambda_0$. Once an even has occurred at rate λ_0, the event rate is changed to $\lambda = \lambda_0 + \delta\lambda$, where $\delta\lambda > 0$ represent the excited state. Thus, this point process model is self exciting in the sense that an event occurrence in the normal state excites the process to generate a cascade of events at a higher rate. More so, the cascaded non-homogeneous Poisson process (cascaded NHPP) assumes that in normal state the event rate is modulated periodically.*

This justifies cascaded NHPP models being used as basis in cyclic or periodic behavior analysis.

2.4. SUMMARY

Here, we discussed a variety of approaches that address the core challenges in large scale networks that also effect the success of various applications over the networks. However, the existing approaches lack integrating network structural characteristics and time-varying properties properly in solving the network challenges. In contrast, as applications are realized over large scale networks or an integral part of it. Our intuition points at leveraging the structural and temporal characteristics in resolving the network challenges in a novel and efficient manner and thus helping to make applications a successful reality. We propose to integrate network structural characteristics in providing novel and efficient algorithms for community formation and betweenness centrality evaluation. Similarly, we propose to integrate time varying nature of networks and the associated structural variations along temporal domain as basis for link prediction model. Further, we integrate the multi-periodic temporal contact pattern as basis of our proposed cascaded $NHPP$ model for contact prediction over mobile networks.

3. COMMUNITY AND CENTRALITY EVALUATION OVER LARGE SCALE NETWORKS: LEVERAGING NETWORK STRUCTURE

Betweenness centrality (BC), a distance based metric, measures the influence of a node from how many times it lies between indirectly connected node pairs. Higher the betweenness centrality index of a node, the more it lies between pairs of other indirectly connected node pairs in the network and is core to the success of many important application problems like diffusion maximization and diffusion minimization, reliability of a network, determining influential spreaders, determining community boundaries to mention a few. Motivated by the need for efficient, and exact evaluation of betweenness centrality, the formal problem definition and the approach for the network structure based between centrality evaluation algorithm is presented here. As we propose a distributed evaluation of the betweenness centrality evaluation and leverage the divide and conquer based algorithmic approach, we consider a network with communities as input to the centrality evaluation algorithm. Thus, determining communities over the application network, is an integral step of our proposed strategy. Here, we first propose a community detection strategy and then evaluate the betweenness centrality.

The following discussion is organized as follows. In Subsection 3.1 we present preliminary concepts and definitions used in centrality and community evaluation strategy. Subsection 3.2 describes our approach for community detection, while Subsection 3.3 discusses our approach for evaluating exact betweenness centrality indices of all nodes in the network. Subsection 3.4 reports experimental results and finally summary of our approach and the obtained results are offered in Subsection 3.5.

3.1. PRELIMINARY CONCEPTS: CENTRALITY EVALUATION

Let $G(V,E,W)$ represent the underlying graph of the directed-weighted application network, where V, E and W denote the set of vertices, edges and edge weights respectively. The number of vertices is considered to be $|V| = n$. For any pair of nodes $u, v \in V$, directed edge from u to v is represented as $e_{uv} \in E$ and a weighted edge is represented as $w(e_{uv}) \in W$, representing variable edge weights, where as unweighted network has uniform edge weights and edges are not annotated with weights. Further, we consider that the edge weights in the weighted network are normalized as $w(e_{uv}) = w(e_{uv}) / \sum_{\forall e_{uv} \in E}(w(e_{uv}))$ such that $\sum_{e_{uv} \in E} w(e_{uv}) = 1$. In absence of an edge between a pair of vertices, the corresponding normalized edge weight is assumed to be 0. The set of communities in the network is represented by $C = \{C_1, C_2, \ldots\}$ where $C_i(V_{C_i}, E_{C_i}) \in C$ represents a community, $C = \cup_i C_i$ and $C_i \cap C_j = \phi, \forall C_i, C_j \in C$ and $i \neq j$; V_{C_i}, E_{C_i} are respectively the set of vertices and edges in C_i. To describe the proposed centrality approach in Subsection 3.3, the related definitions are introduced here.

- *Internal Edges*: In any $C_i(V_{C_i}, E_{C_i}), \forall e_i(v_k, v_l)$, such that $v_k, v_l \in V_{C_i}$, then $e_i \in E_{C_i}$ is called an *Internal edge* of C_i.

- *External Edges*: Across any pair of communities $C_i, C_j, \forall e_{ij}(v_k, v_l)$, such that $v_k \in V_{C_i}, v_l \in V_{C_j}$, then $e_{ij} \in E$ is called *External edge* between C_i and C_j.

- *External (Boundary) Node*: This is internal to a community with at least one incident external edge.

- *Internal Node*: It is internal to a community with no incident external edge.

- *Internal (Local) Centrality*: In any community the Internal (local) betweenness centrality index associated with node v_i, is denoted as $\delta_{Int}[v_i]$ and computed over local topology for C_i, taking into account only internal edges. The node $v_i \in V_{C_i}$ having maximum local centrality index $\delta_{Int}[C_i] = max\{\delta_{Int}[v_i]\}$ is called local or internal central node for that community.

- *External Centrality*: For any set of communities external betweenness centrality index for node ($\delta_{ext}[v_i]$) is defined as the number of times the node acts as a connector or bridge across distinct communities.

- *Global Centrality*: For graph $G(V, E)$, global centrality of a vertex v_i considers both internal and external edges over graph G_R. If $\delta[v_i]$ is the global centrality index of vertex v_i, then the node with highest global centrality index in C $\delta[C] = max\{\delta[v_i]\}$ is called the global central node.

- *Virtual Cluster*: For any external node in a community ($v_{ext_i} \in C_i$), virtual cluster $VC_{v_{ext_i}}$ is the subset of nodes in C_i that have the shortest path to v_{ext_i} over all $v_{ext_j} \in C_i, v_{ext_i} \neq v_{ext_j}$.

- *Mutual Virtual Cluster*: Over any pair of virtual clusters in a community, the mutual virtual cluster ($MVC_{v_{ext_i}, v_{ext_j}}, v_{ext_i}, v_{ext_j} \in C_i, v_{ext_i} \neq v_{ext_j}$) is the set of nodes in $VC_{v_{ext_i}}$ whose shortest path to v_{ext_j} is not via any $v_{ext_i} \in C_i$.

- *Complemented Mutual Virtual Cluster*: Over any pair of virtual clusters in a community, complemented mutual virtual cluster ($CMVC_{v_{ext_i}, v_{ext_j}}$) is the set of nodes in $VC_{v_{ext_i}}$ whose shortest path to v_{ext_j} is via $v_{ext_i} \in C_i$.

For brevity we present the associated diagram explaining the definitions in Figure 3.3, Subsection 3.3.

Before going into the details of our proposed approach, we first formally define our problem. Here, we are specifically interested in the *betweenness centrality* (BC), a relative measure of occurrence of node v over all pairs of shortest paths, defined as: $BC[v] = \frac{\sigma_{st}[v]}{\sigma_{st}}$, where $s, v \in V, t \in V \setminus \{s\}$, V is the total set of nodes, $\sigma_{st}[v]$ is the number of shortest $s \rightsquigarrow t$ paths from s to t going through node v, and σ_{st} is the total number of shortest $s \rightsquigarrow t$ paths in the network. We compute the exact BC indices of all nodes in the network.

3.2. COMMUNITY STRUCTURE DETECTION

The proposed approach presents an algorithm to detect $O(\sqrt{n})$ communities, each of size $O(\sqrt{n})$, $|V| = n$ with computational cost $O(n^2 - mk^2 n)$. The details over undirected-weighted network are discussed and then expanded it over to directed-weighted network.

3.2.1. Initial Community Selection. We consider the set of nodes with degree one as our initial set of unit size communities. This is in contrast to the usual consideration in agglomerative hierarchical community detection schemes of either starting with each node in the network as a distinct community, or starting with top r nodes ranked as per their highest degree. Due to the right skewed-ness in degree distribution of large scale application networks, there are nodes (though few) with very high degree, as large as $O(n)$. To include subsequent neighbors to the respective communities, starting at these top ranked nodes would have incurred higher computational cost of $O(n^2)$ per node in modularity computation. In contrast, due to connected network, power-law degree distribution, coupled with our techniques, the initial community selection needs at most $O(k^2)$ time per node, where k is the average degree. A combination of proposed subsequent techniques ensure computational cost reduction. Before discussing rest of the techniques, they are first explained in an example in Figure 3.1 over a directed network.

Illustration1: The example weighted, directed network with non-normalized edge weights has 3 nodes A, D, I, each of degree one. Irrespective of their affinity with their neighbors they have higher priority than virtual nodes (here, J, with self-edge weight

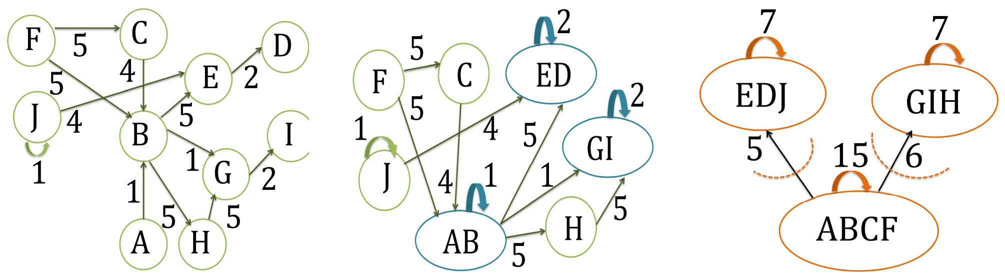


Figure 3.1. Community Detection Steps and Final Outcome

1) and are self-accumulated, getting virtual nodes AB, ED, GI and respective self-edge weights $1,2,2$. From each virtual node we check if there is any node / virtual node (J) that is adjacent to only the virtual node. Node J is self-accumulated with virtual node ED, resulting in virtual node EDJ and self-edge weight 7. No other self-accumulation takes place at this stage, as EDJ has higher self-edge weight than connecting edge with AB and AB has neighbor nodes (F, H) with same edge weight; nodes (virtual nodes) with higher connecting edge weights get precedence (F, H, EDJ over C, GI). Between nodes and virtual nodes of same connecting edge weight, nodes get precedence. Now, nodes F, H get precedence over C and EDJ. Instead of randomly accumulating them with AB, we check their list of neighbors to find the number and weight of the associated tie of common neighbors with AB. Precisely, we are checking triads ($\{F, C, AB\}, \{H, GI, AB\}$) and their connecting edge weight ($(AB, C), (AB, GI)$). The one with higher weight takes precedence such as (AB, C). In case these weights are same, the triads with higher total weight, i.e., (F, C) versus (H, GI) or $\{(F, AB) + (C, AB) + (F, C)\}$ versus $\{(H, AB) + (H, GI) + (AB, GI)\}$ takes precedence. The resulting accumulated virtual nodes created at this stage is $ABFC$ and GIH with self-edge weights 15 and 7 respectively. Virtual node GIH is connected to AB with edge weight 6 which is less than its self-edge weight, similar to EDJ; so they are not accumulated with $ABFC$. Our approach optimizes modularity at every stage of the accumulation irrespective of the constraint. In fact, our final outcome just fits fine

for a maximum limit of 4 on size and number of communities. The resulting set of final communities is shown in the extreme right of Figure 3.1. In the following subsection the techniques are formally discussed.

3.2.2. Incremental Accumulation. In an attempt to reduce the associated computational cost, the vertices in the same community are merged to a single node, called the accumulated virtual node. This process intrinsically merges incident edges on nodes, reducing the number of edges to be scanned in subsequent stages and helping single adjacency nodes to be self-accumulated without modularity gain computation. Note that any node (or virtual node) pair i and j to be merged have at most three types of incident edges. The set of edges ($e_{V\setminus\{i\}}$) incident on i from remaining nodes and similarly $e_{V\setminus\{j\}}$ set of edges incident on j, edges between i and j (e_{ij}), and the set of edges within community i (e_{ii}) or community j (e_{jj}). In creating an accumulated virtual node ($k' = i \cup j$), the incident external edges to k' are edges of $1st$ type above ($e_{V\setminus\{k'\}}$), whereas the internal/self edges in k' ($e_{k'k'}$) are the remaining two types of edges from above. Further, self-edge weight is given by $w(e_{k'k'}) = \sum_{\forall i,j \in k'}(w(e_{ii}) + w(e_{jj}) + 2w(e_{ij}))$ over undirected network that is intrinsically bi-directional. Similarly, external edge weight $w(e_{V\setminus k'}) = \sum_{i',j'} w(e_{ii'}) + w(e_{jj'}), i' \in V \setminus \{i\}, j' \in V \setminus \{j\}$. As the approach will use the modularity function [30, 102] in order to validate the resulting community structure, it can be extended to modularity over undirected-weighted networks and can be defined as follows.

$$Q = \sum_i \left(\frac{w(e_{ii})}{\sum_{e_{m'n} \in E} 2w(e_{m'n})} - \left(\frac{w(a_i)}{\sum_{e_{m'n} \in E} 2w(e_{m'n})}\right)^2\right) \tag{3.1}$$

where $w(e_{ii})$ and $w(a_i)$ are edge weights within and across community i respectively, and the denominator in the fractions represent total edge weight over the graph. With normalized edge weights as defined in Subsection 3.1, Equation (3.1) reduces to:

$$Q = \sum_i \left(\frac{w(e_{ii})}{2} - \frac{(w(a_i))^2}{4}\right) \tag{3.2}$$

To define modularity gain over accumulated virtual node we proceed as follows. Let i and j be two prospective nodes to be merged. The edge weight between i and j is $w(e_{ij})$. Let the neighbors of a node (say i) be represented as $N(i)$. Node $j \in N(i)$ is considered as a candidate for merging with node i when $w(e_{ij}) = MAXw(e_{iN(i)})$ and $w(e_{ij}) = MAXw(e_{jN(j)})$, where $w(e_{iN(i)})$, $w(e_{jN(j)})$ represents the set of edge weights between node i, $N(i)$ and j, $N(j)$ respectively. In other words $\forall_{j' \in N(i) \setminus \{j\}}(w(e_{ij}) - w(e_{ij'})) > 0$ and $\forall_{i' \in N(j) \setminus \{i\}}(w(e_{ij}) - w(e_{ji'})) > 0$. Modularity gain over accumulated virtual node k' ($\delta Q'_k = \delta Q_{ij}$) is obtained as:

$$\delta Q_{ij} = (\frac{w(e_{ij})}{\sum_{e_{m'n} \in E} 2w(e_{m'n})} - \frac{w(e_{iN(i)\setminus\{j\}})w(e_{jN(j)\setminus\{i\}})}{(\sum_{e_{m'n} \in E} 2w(e_{m'n}))^2}). \quad (3.3)$$

With normalized edge weights, Equation (3.3) reduces to:

$$\delta Q_{ij} = \delta Q_k = (\frac{w(e_{ij})}{2} - \frac{w(e_{iN(i)\setminus\{j\}})w(e_{jN(j)\setminus\{i\}})}{4}). \quad (3.4)$$

Now, we define the accumulated virtual node creation in relation with modularity.

Definition 3.2.1 ***(Accumulated virtual node creation)*** *Let the neighbors of node i be $N(i)$. Let $j \in N(i)$ that maximizes modularity, then nodes i and j are merged to create accumulated virtual node k'. Any neighbor j of i resulting in $\delta Q_{ij} > 0$, i.e., positive modularity gain becomes a candidate for merging, while for any $j \in N(i)$ with $\delta Q_{ij} < 0$, i.e., negative modularity gain is rejected from being merged to i.*

From the earlier definition of $e_{k'k'}$ we can define modularity of the virtual node k' with normalized edge weights as:

$$Q_{k'} = \frac{w(e_{k'k'})}{2} - \frac{w(e_{k'N(k')\setminus\{k'\}})^2}{4}$$
$$= \frac{w(e_{ii})+w(e_{jj})+2w(e_{ij})}{2} - \frac{(w(e_{iN(i)\setminus\{j\}})+w(e_{jN(j)\setminus\{i\}}))^2}{4}$$
$$= Q(i,j) = \frac{w(e_{k'k'})}{2} - \frac{w(a_{k'})^2}{4}$$
$$= \frac{[w(e_{ii})+w(e_{jj})+2w(e_{ij})]}{2} - \frac{[(w(a_i)+w(a_j))^2]}{4}.$$

From Equation 3.2 $Q_{k'}$ can be represented in simpler form as:

$$Q_{k'} = \frac{w(e_{k'k'})}{2} - \frac{w(a_{k'})^2}{4}$$
$$= \frac{[w(e_{ii})+w(e_{jj})+2w(e_{ij})]}{2} - \frac{[(w(a_i)+w(a_j))^2]}{4} = Q(i,j)$$

As accumulated node k' has same modularity as that of nodes i, j jointly, the accumulation of vertices (or edges) does not affect the modularity of the vertices (or edges) being accumulated. Let us further consider nodes $j, p, l \in N(i)$. Let $\delta Q_{ij} > 0$, $\delta Q_{ip} < 0$ and $\delta Q_{il} < 0$. Let us also consider that $N(i) \setminus \{j, p, l\} \neq \phi$. Let j, p, l be accumulated as part of the same community and form a virtual node. Further, the edges between nodes i and j, p, l will be accumulated to a single virtual edge. Now, node i will be a candidate for accumulation with virtual node containing j, p, l iff $\delta Q_{ij} > \delta Q_{ip} + \delta Q_{il}$. Moreover, a node having negative modularity with its neighbors before laters are accumulated, is refrained from further consideration to aggregate it with that virtual node.

3.2.3. Incremental Elimination. During incremental accumulation two types of nodes are accumulated irrespective of the resulting modularity gain, as they always result in positive modularity gain. Let C_i represent the community containing any node $i \in V$. The self accumulated nodes (SA) at any iteration t of the accumulation phase is defined as:

$$SA_t = \begin{cases} \phi & \text{if t=0} \\ \{i : |N(i)| = 1\} \cup & \\ \{i : \forall j, l \in N(i), C_j = C_l\} & t > 0 \end{cases}$$

Thus, a node is in the set of self accumulated nodes if either it has degree one or all its neighbors are in the same community.

Lemma 3.2.1 ***(Modularity gain in self accumulated node)*** *For any node $i \in SA_t$, at any iteration (say t) of accumulation phase, the modularity gain of i over $N(i)$ is $\delta Q_{iN(i)} > 0$.*

Proof 3.2.2 *For nodes (say i) with single degree, let $j \in N(i)$. $\frac{w(e_{ij})}{\sum_{j \in N(i)} w(e_{iN(i)})} = 1$, $\frac{w(e_{iN(i)\setminus\{j\}})}{\sum_{j \in N(i)} w(e_{iN(i)})} = 0$ and $0 < w(e_{jN(j)\setminus\{i\}}) < 1$. Thus, $\delta Q_{ij} = 1 > 0$. For nodes (say i) with $|N(i)| > 1$ and $\forall j, k' \in N(i) : C_j = C'_k$, then $\frac{\sum_{\forall N(i) \in C_j} w(e_{iN(i)})}{\sum_{N(i)} w(e_{iN(i)})} = 1$, $\frac{\sum_{\forall l, j \in N(i): C_l \neq C_j} w(e_{il})}{\sum_{N(i)} w(e_{iN(i)})} = 0$. So, δQ_{ij} and hence modularity gain satisfies $\delta Q_{iN(i)} > 0$.*

It is thus clear from above that the self accumulated vertices do not compromise modularity gain. The question now is how to select these nodes, so as to reduce the computational cost.

3.2.4. Semi-local Optimal Node Selection. Our objective is to optimize computational cost along with the selection of most deserving nodes into prospective community. Because of the $O(\sqrt{n})$ limit on the number and size of communities, the optimal node order selection is more significant here. We propose a semi-local approach for node order selection. A node $j \in N(i)$ resulting in highest modularity gain among all the neighbors of i is a candidate to be part of the community. Instead of accumulating it, we also look for its neighbors that collectively with i and $N(i)$ produces highest modularity gain and collectively accumulate them. Let $j \in N(i)$ be the node with maximum modularity at any stage t. Let $MAX(\delta Q_{iN(i)})$ be the maximum modularity gain between nodes i and $j \in N(i)$. For the node with highest modularity gain, semi-local modularity optimization is used to select the most suitable set of nodes at once. So, $\forall j \in N(i) : \delta Q_{ij} = MAX(\delta Q_{iN(i)})$, we find $\{N(i)\setminus\{j\}\} \cap \{N(j)\}$. Now, $\forall j' \in \{N(i)\setminus\{j\}\} \cap \{N(j)\}$ we find $\delta Q_j^{triad} = \delta Q_{ij} + \delta Q_{ij'} + \delta Q_{jj'}$. For positive modularity gain i.e., $\delta Q_j^{triad} > 0$, and $\forall j' : \delta Q_j^{triad} = MAX(\delta Q_j^{triad})$, we accumulate j, j', i to form a single virtual node and accumulate the corresponding edges. When more than one node has highest modularity gain, semi-local modularity optimization is used to determine the order of selection of a suitable set of nodes at once. Let the number of neighbors with maximum modularity at any stage t be greater than 1. The approach is as follows:

$\forall j_1, j_2 \in N(i) : \delta Q_{ij_1} = \delta Q_{ij_2}$, we find $\{N(i) \setminus \{j_1\}\} \cap \{N(j_1)\}$ and $\{N(i) \setminus \{j_2\}\} \cap \{N(j_2)\}$. Now, $\forall j'_1 \in \{N(i) \setminus \{j_1\}\} \cap \{N(j_1)\}$ and $\forall j'_2 \in \{N(i) \setminus \{j_2\}\} \cap \{N(j_2)\}$ we find total modularity gain involving i, j_1, j'_1 and i, j_2, j'_2 as:

$\delta Q_{j_1}^{triad} = \delta Q_{ij_1} + \delta Q_{ij_1'} + \delta Q_{j_1 j_1'}$ and

$\delta Q_{j_2}^{triad} = \delta Q_{ij_2} + \delta Q_{ij_2'} + \delta Q_{j_2 j_2'}$.

Now, $\forall j_1, j_2 \in N(i) : \delta Q_{j_1}^{triad}, \delta Q_{j_2}^{triad} > 0$ and the node triplets have maximum modularity gain.

The corresponding triads j_1, j_1', i and (or) j_2, j_2', i are accumulated, creating a virtual node and virtual edge(s) incident on this node. In a directed graph a node with higher in-degree is considered to be more influential, relative to a node with higher out-degree. Let the corresponding impact factor (that determines relative importance of either degree) for in-degree and out-degree be $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$ respectively, such that $\alpha + \beta = 1$. Let the directed-weighted edge from any node j to node i and node i to node j be represented as $w(e_{ij})$ and $w(e_{ji})$ respectively. The total degree of a node i in the weighted-directed graph is $\alpha w(e_{iN(i)}) + \beta w(e_{N(i)i})$. Now, Equation (3.4) can be defined over directed-weighted graph with normalized edge weights as in the following equation.

$$\begin{aligned} \delta Q_{ij}^{Directed} &= (\alpha w(e_{ij}) + \beta w(e_{ji})) \\ &- (\alpha w(e_{iN(i)\backslash\{i,j\}}) + \beta w(e_{N(i)i\backslash\{i,j\}})) \\ &(\alpha w(e_{jN(j)\backslash\{j,i\}}) + \beta w(e_{N(j)j\backslash\{j,i\}})). \end{aligned}$$

Over the directed-weighted graph, $\delta Q_{ij}^{Directed}$ is used as the basic step in semi-local optimal node selection and the techniques described above are applied.

We give the schematic presentation of the steps of community formation algorithm in Figure3.2.

Algorithm 1 for community detection is equally applicable to undirected and directed-weighted network. It starts with finding degree one nodes and forming unit size communities, followed by self-accumulation. The set of remaining vertices including the set of virtual nodes, CommunitySize and NumberOfCommunity are updated and smallest degree nodes are selected from the nodes that are not self-accumulated at the current iteration. The

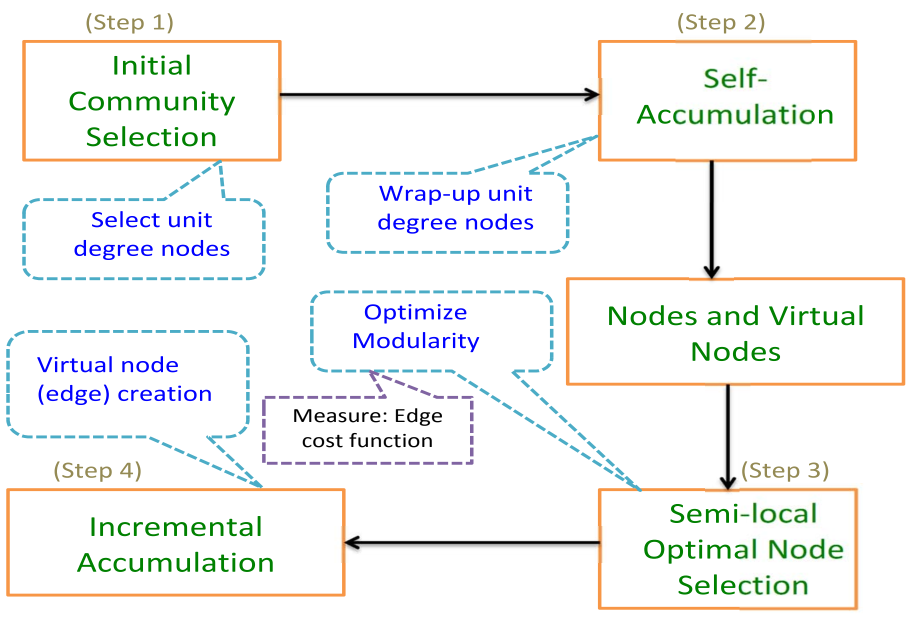

(Step 1)
Initial Community Selection
(Step 2)
Self-Accumulation
Select unit degree nodes
Wrap-up unit degree nodes
Nodes and Virtual Nodes
Optimize Modularity
Virtual node (edge) creation
Measure: Edge cost function
(Step 4)
(Step 3)
Incremental Accumulation
Semi-local Optimal Node Selection

Figure 3.2. Steps of Community Formation Algorithm

semi-local optimal node selection strategy is followed and nodes are accumulated to produce virtual node and virtual edge(s), size and number of communities are updated. Finally, the set of vertices (V) are updated using the set of to be accumulated nodes in that iteration. At the beginning of each iteration, the set of single adjacency nodes are self-accumulated with their respective adjacent nodes. The algorithm repeats itself until the node set $|V|$ is empty, and the limit on the size and number of communities does not exceed.

3.2.5. Computational Cost. The proposed algorithm has major reduction in computational cost from the incremental aggregation coupled with semi-local optimal node selection, though self-accumulation also effects. It outperforms existing best known agglomerative hierarchical and decisive community detection algorithm having time complexity of $d|V|^2 log|V|$, where $d \sim lg|V|$ [30] and $O(|V|^3)$ [102] respectively. Further, the spectral algorithm [135] incurs cost $O(|V|)$ over sparse graphs with $|E| \sim |V|$, whereas our algorithm costs $O(|V| - m|V|), 0 < m < 1, k \sim 1$ in the same domain. Though, over dense graphs the fastest version of [135] takes $O(|V|K^2e)$, where K, and e are respectively the number

Algorithm 1 Community Detection Algorithm

```
Require: Network graph $G_R(V, E, W), |V| = n$
Ensure: $O(\sqrt{n})$ Communities each of size $O(\sqrt{n})$
  $t = 0, SA_0 = \{\}, t = t + 1, SA_t = \{i : |N(i)| = 1\}$;
  Initialize $|SA_t|$ communities each of size one
  while $|V| > 0$ or $(CommunitySize \not> O(\sqrt{n})$ or $NumberOfCommunity \not> O(\sqrt{n}))$ do
    $t = t + 1$; $SA_t = \{i : |N(i)| = 1\}$;
    for $\forall i \in SA_t$: do
      Accumulate node $i$ with its adjacent node; Update $CommunitySize$,
      $NumberOfCommunity$;
    end for
    $|V| = |V| - SA_t$; Initialize $NodesToBeAccumulated$;
    Select node with smallest degree and maximum $\delta Q_{ij}$;
    Create Virtual node and edges , Update CommunitySize
    Update NumberOfCommunity, NodesToBeAccumulated;
  end while
```

of iterations of K-means algorithm and number of iterations of the proposed algorithm in [135]. On the other hand our algorithm over dense graphs cost $O(|V|^2 - mk^2|V|)$, where $|E| \simeq |V|^2$.

Theorem 3.2.3 Computation Cost in Community Detection: *The proposed agglomerative hierarchical community detection approach incurs computation cost of $O(|V|^2 - mk^2|V|)$, where $m, k, |V|$ are the modularity, average degree and number of nodes of the network, respectively.*

Proof 3.2.4 *Let us consider the graph with $O(|V|^2)$ edges. The number of modularity computation is done over every pair of adjacent vertices, thus requiring at most $O(|V|^2)$ computations. Semi-local optimal node selection strategy selects a set of triads (node triplets) with highest modularity gain. Let k be the average degree. For any node, the selection strategy checks at least on its neighbors and also neighbor's neighbors. Thus, the number of modularity computations are $O(|V|k^2)$. Reduction in the number of computations for next stage is proportional to the number of nodes (hence edges) accumulated in the current* *stage, which is proportional to* $|V|k^2$. *Furthermore, two nodes being accumulated is also dependent on the modularity m, resulting in the number of accumulated edges at any stage as* $m|V|k^2$. *This leads to, computation cost of* $O(|V|^2 - m|V|k^2)$.

Now that we have $O(\sqrt{n})$ modules (communities) of size $O(\sqrt{n})$ each, we discuss our approach for exact betweenness centrality evaluation in the following section.

3.3. EXACT BETWEENNESS CENTRALITY EVALUATION

The following proposed approach to betweenness centrality evaluation is also equally applicable to other distance based measures, such as *Stress Centrality* [120] and *Closeness centrality* [19].

3.3.1. Internal Betweenness Centrality With Example. Community C_i is represented by a graph $G_{C_i}(V_{C_i}, E_{C_i}, W)$ where, V_{C_i}, E_{C_i}, W correspond to the set of nodes, and edges within community C_i and W is the set of edge weights. In Algorithm 2, internal centrality indices of nodes are computed over each community C_i with only internal edges. The approach makes use of the list of predecessors for any shortest path, recursive dependency computation intrinsic to shortest paths and Brandes [21] centrality aggregation function as the basis behind Algorithm 2. Concisely, for predecessor node v in predecessor set ($P_s[t]$) of shortest path $s \rightsquigarrow v \rightarrow t$, the dependency $\delta_{st}[v] = \frac{\sigma_{st}[v]}{\sigma_{st}}$. This can also be stated recursively as:
$\delta_{st}[v] = \frac{\sigma_{sv}}{\sigma_{su_1}} \times \frac{\sigma_{su_1}}{\sigma_{su_2}} \times \ldots \times \frac{\sigma_{su_k}}{\sigma_{st}}$, where the shortest path $s \rightsquigarrow v \rightsquigarrow t$ has nodes $u_1, u_2, \ldots, u_k$ with v as their predecessors, i.e., the path $s \rightsquigarrow v \rightarrow u_1 \rightarrow u_2 \rightsquigarrow \ldots \rightsquigarrow u_k \rightarrow t$ exists. Now, when computing dependency on node v for all shortest paths $s \rightsquigarrow t$, $\forall t \in V \setminus \{s\}$, the dependency index are recurssively accumulated not only for $s \rightsquigarrow v \rightsquigarrow t$, but also for all nodes u in $s \rightsquigarrow v \rightsquigarrow u \rightsquigarrow t$ such that $v \in P_s[u]$. Thus, in the above mentioned shortest path $s \rightsquigarrow v \rightarrow u_1 \rightarrow u_2 \rightsquigarrow \ldots \rightsquigarrow u_k \rightarrow t$, the dependency index on node v is given by: $\delta_s[v] = \frac{\sigma_{sv}}{\sigma_{su_1}} + \frac{\sigma_{sv}}{\sigma_{su_2}} + \ldots + \frac{\sigma_{sv}}{\sigma_{su_k}} + \frac{\sigma_{sv}}{\sigma_{st}} = \frac{\sigma_{sv}}{\sigma_{su_1}} + \frac{\sigma_{sv}}{\sigma_{su_1}} \times \frac{\sigma_{su_1}}{\sigma_{su_2}} + \ldots + \frac{\sigma_{sv}}{\sigma_{su_1}} \times \frac{\sigma_{su_1}}{\sigma_{su_2}} \times \ldots \times \frac{\sigma_{su_k}}{\sigma_{st}}$. The

above formulation reduces to the aggregation function defined in [21]. The dependency index on node v, over all shortest paths $s \rightsquigarrow t$, for $t \in V \setminus \{s\}$, starting at node s is obtained as: $\delta_s[v] = \sum_{\forall t \in V \setminus \{s\}} \{ \sum_{\forall v : v \in P_s[u]} \frac{\sigma_{sv}}{\sigma_{su}} (1 + \delta_s[u]) \}$.

The, betweenness centrality index of any node v is its overall dependency index with respect to all pairs of shortest paths $s \rightsquigarrow t$, $\forall s \in V, \forall t \in V \setminus \{s\}$, is given by $\delta[v] = \sum_{\forall s \in V} \delta_s[v] = \sum_{\forall s \in V} \{ \sum_{\forall t \in V \setminus \{s\}} \{ \sum_{\forall v : v \in P_s[u]} \frac{\sigma_{sv}}{\sigma_{su}} (1 + \delta_s[u]) \} \}$.

Algorithm 2 presents the pseudocode to evaluate the exact internal betweenness centrality indices of all nodes over directed-weighted graph.

Algorithm 2 Calculate Internal Betweenness Centrality over Weighted-Directed Network Graph

```
Require: Sub-region $C_i = G_{C_i}(V_{C_i}, E_{C_i}, W)$ a directed-weighted sub-graph, $|V_{C_i}| = O(\sqrt{n})$ nodes and edge set $e_{(v_i,v_j)} \in E_{C_i}$ and edge weight $w(e_{(v_i,v_j)}) \in W$
Ensure: Internal Betweenness centrality indices $\delta_{Int}[v], \forall v \in V_{C_i}$.
  $StartNode = \{\}, \forall v \in V_{C_i} : \delta_{Int}[v] = 0$;
  while $StartNode \neq V_{C_i}$ do
    SelectSelect StartNode $s$, evaluate shortest path to $v \in \{V\} \setminus \{s\}$, update immediate predecessors list. Count the number of shortest path to node $\sigma_s[v], v \in \{V\} \setminus \{s\}$. Compute $\delta_{Int}[v]$.
  end while
  Construct Virtual, Mutual Virtual and Complemented Mutual Virtual clusters and find $\sigma_{v_{ext_i}}[u_i], v_{ext_i}, u_i$ are external node and internal node in respective cluster.
  $\forall v, v_{ext} \in V_{C_i}$: Return $\delta_{Int}[v]$, $\sigma_{v_{ext_i}}[v]$, $VC_{v_{ext}}$, $MVC_{v_{ext_i}, v_{ext_j}}$, $InternalAndExternalEdgeSet$, $CMVC_{v_{ext_i}, v_{ext_j}}$.
```

From internal centrality evaluation, we have shortest weighted path from every internal node to the external nodes within each community and vice versa. The shortest paths are used to form the set of virtual clusters, mutual virtual clusters and complemented mutual virtual clusters during this stage. We also have the list of predecessors, and the number of successors of a node within the respective clusters. Internal nodes within the respective clusters are ordered in the increasing order of their number of successors. Further, we keep track of the number of shortest paths from the external node to the internal nodes in the respective clusters. Without loss of generality we can assume that within any community the shortest distance between an external node in one virtual cluster and any other virtual cluster is from the external node in the later cluster. Before presenting Algorithm 3 for external centrality evaluation, an example (Figure 3.3) is presented for simplicity.

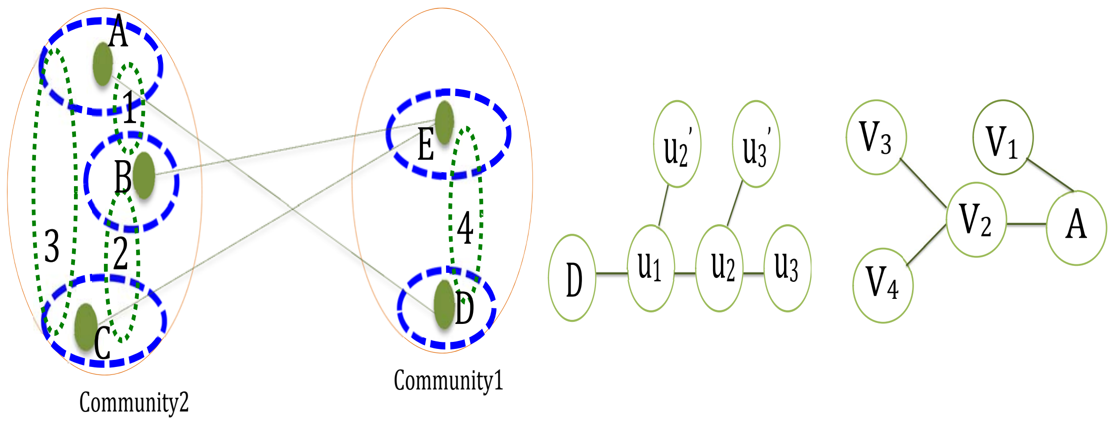


Figure 3.3. Community with Virtual Clusters and External Edges

Illustration2: Let us consider two communities $Community1$, $Community2$. Consider virtual clusters $A_{vc}, B_{vc}, C_{vc}, D_{vc}, E_{vc}$ over external nodes A, B, C, D, E respectively, mutual virtual clusters between $A_{vc}B_{vc}, B_{vc}C_{vc}, A_{vc}C_{vc}$ as $1, 2, 3$ respectively, and between $D_{vc}E_{vc}$ as 4. Similarly, their complemented mutual virtual clusters $1' = \{A_{vc}\} \setminus \{1\}$ and so on. We also have the number of shortest paths a node has from the respective external node (e.g., $\sigma_D[u], \forall u \in D_{vc}$). For external centrality evaluation, we first find shortest paths between every pair of external nodes; we also keep track of the number of shortest paths between respective pairs and the external centrality indices the external nodes get at this stage. External centrality indices of all nodes in the respective clusters are initialized to 0 at the beginning (say $\delta_{ext_D}[u_i] = 0, \forall u_i \in D_{vc}$); nodes other than leaf nodes are evaluated and updated. Let us assume that we have shortest paths $A \rightarrow D$ and $A \rightarrow B \rightarrow E$. Let the internal nodes within D_{vc} are as shown in the middle graph in Figure 3.3. We find external centrality index of nodes in D_{vc} caused due to nodes in A_{vc}, i.e., $\forall v_i \in A_{vc}$ acting as the source node s and due to the $A \rightarrow D$ path. Nodes are considered in increasing order of successors, with external centrality index of its successor nodes known. So, external centrality of u_2 is computed, due to the path to u_3 and u_3'. For u_2 we compute

$|A_{vc}|\frac{\sigma_D[u_2]}{\sigma_D[u_3]}$, as for each node in A_{vc} this path is explored ($\frac{\sigma_s[u_2]}{\sigma_s[u_3]} = \frac{\sigma_s[D]\sigma_D[u_2]}{\sigma_s[D]\sigma_D[u_3} = \frac{\sigma_D[u_2]}{\sigma_D[u_3]}$). Similarly, due to u'_3, $|A_{vc}|\frac{\sigma_D[u_2]}{\sigma_D[u'_3]}$. Let $\delta_{ext_D}[u_2]$ be the external centrality index of u_2 due to interface node D. Thus, $\delta_{ext_D}[u_2] = \delta_{ext_D}[u_2] + |A_{vc}|(\frac{\sigma_D[u_2]}{\sigma_D[u_3]} + \frac{\sigma_D[u_2]}{\sigma_D[u'_3]})$. Similarly, for u_1 we evaluate $\delta_{ext_D}[u_1] = \delta_{ext_D}[u_1] + \frac{\sigma_D[u_1]}{\sigma_D[u_2]}(|A_{vc}| + \delta_{ext_D}[u_2]) + \frac{\sigma_D[u_1]}{\sigma_D[u'_2]}(|A_{vc}| + \delta_{ext_D}[u'_2])$. Now that we have external centrality indices for all internal nodes in D_{vc}, we compute the same for D as $\sum_{u_i}(\frac{1}{\sigma_D[u_i]}), u_i \in D_{vc}$, ($\sum_{u_i}\frac{\sigma_s[D]}{\sigma_s[u_i]} = \sum_{u_i}\frac{\sigma_s[D]}{\sigma_s[D]\sigma_D[u_i]}, u_i \in D_{vc}$). So, $\delta_{ext}D = \delta_{ext}D + |A_{vc}|(\sum_{u_i}(\frac{1}{\sigma_D[u_i]}))$. Similarly, external centrality of A will be evaluated as $|A_{vc}|\frac{1}{\sigma_A[D]}\sum_{u_i}(\frac{1}{\sigma_D[u_i]}), \forall u_i \in D_{vc}$. Thus, $\delta_{ext}[A] = \delta_{ext}[A] + |A_{vc}|\frac{1}{\sigma_A[D]}\sum_{u_i}(\frac{1}{\sigma_D[u_i]})$. Now, for nodes in A_{vc} with internal centrality 0, we do not need to compute external centrality index. So, for nodes having nonzero number of predecessors ($\forall v_i \in A_{vc}, P[v_i] \neq 0$), let $P_{total}[v_i]$ denotes the total number of predecessors of v_i. Considering the rightmost graph in Figure 3.3, since v_2 has $P_{total}[v_2] > 0$, the external centrality $\delta_{ext}[v_2] = \delta_{ext}[v_2] + \frac{\sigma_s[v_2]}{\sigma_s[A]} + \frac{\sigma_s[v_2]}{\sigma_s[A]}\frac{1}{\sigma_A[D]}(\sum_{u_i}\frac{1}{\sigma_A[D]}), \forall u_i \in D_{vc}$. We know $\frac{1}{\sigma_A[D]}(\sum_{u_i}\frac{1}{\sigma_A[D]})$ from external centrality evaluation for A, which we substitute in evaluating centrality indices for nodes in A_{vc}. Now, that we are done with $A \rightarrow D$ path, let us assume that the $A \rightsquigarrow E$ path exists via B. Here the computations are performed as in the earlier case except that instead of considering A_{vc}, mutual and complemented mutual virtual clusters 1 and 1′ are considered along with B_{vc}. For a given cluster, the external centrality indices of its nodes are computed only once. Subsequent, reference to their interface external node (hence cluster) will only use these values, thus minimizing the computation cost.

3.3.2. External and Global Betweenness Centrality Evaluation. We present the steps of the external centrality evaluation in Algorithm 3. As we explained the basic steps in the example, we omit the details for brevity. Now, that we have internal and external betweenness centrality ($\delta_{Int}[v], \delta_{ext}[v]$), $\forall v \in V$ indices, Algorithm 4 gives the final step for computing global centrality index. Centrality being an additive measure, summing the internal and external centralities gives a node's global betweenness centrality.

Algorithm 3 Calculate External Centrality Indices over Weighted-Directed Network Graph

```
Require: Output from Algorithm 2
Ensure: External Betweenness centrality $\forall v \in V : \delta_{ext}[v]$
  1. Compute shortest path, record ImmediatePredecessors.
  2. $\forall v_{ext_i}, v_{ext_j}$, Record the number of shortest paths between pairs of vertices $(v_{ext_i}, v_{ext_j}), i \neq j$.
  3. Compute centrality indices of external nodes over with only external edges.
  4. Evaluate external centrality of all nodes. Consider Virtual cluster or Mutual and ComplementedMutual Virtual cluster based on whether the shortest path across external vertices excludes external node of the same cluster or includes them.
  5. Reurn $\forall v \in V : \delta_{ext}[v]$.
```

Algorithm 4 Compute Overall (global) Betweenness Centrality over Weighted-Directed Network

```
Require: Internal and External Centrality indices for each node in $G_R$ i.e., $\forall v \in V : \delta_{Int}[v], \delta_{ext}[v]$
Ensure: Betweenness centrality indices $\delta[v], \forall v \in V$.
  for $(v \in V)$ do
    $\delta[v] = \delta_{Int}[v] + \delta_{ext}[v]$; $BC(v) = \delta[v]$;
  end for
```

A schematic representation of the steps of betweenness centrality evaluation procedure is shown in the following block diagram in Figure 3.4.

Theorem 3.3.1 Correctness Proof: *The proposed approach correctly and effectively evaluates the centrality indices over all nodes.*

Proof 3.3.2 *Algorithm 2 uses internal edges and dependency among nodes in each community to evaluate* $\delta_{Int}[v]$ *with* $s \rightsquigarrow t$, $\forall s \in V_{C_i}, \forall t \in V_{C_i} \setminus \{s\}$. *It is given by* $\delta_{Int}[v] = \sum_{\forall s \in V} \delta_s[v] = \sum_{\forall s \in V_{C_i}} \{\sum_{\forall t \in V_{C_i} \setminus \{s\}} \{\sum_{\forall v : v \in P_s[u]} \frac{\sigma_{sv}}{\sigma_{su}} (1 + \delta_s[u])\}\}$, *for nodes in respective communities. Now, the dependency among nodes due to connections across communities is left to be considered next. This first evaluates* $\delta_{ext}[v_{ext}], \forall v_{ext} \in V$ *by considering shortest paths among external nodes in the same community (from internal centrality evaluation) and external edges (e.g.,* $A \rightarrow B \rightarrow E$ *paths). Computing external centrality indices due to node-to-node path across communities is the most trickier one. There can be two types*

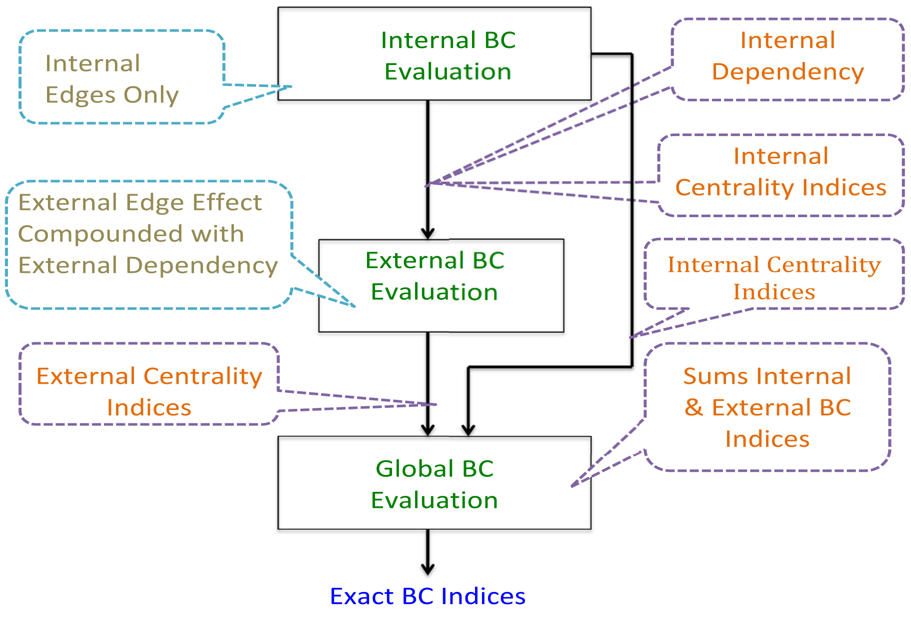

Internal BC Evaluation
Internal Edges Only
Internal Dependency
Internal Centrality Indices
External Edge Effect Compounded with External Dependency
External BC Evaluation
Internal Centrality Indices
External Centrality Indices
Sums Internal & External BC Indices
Global BC Evaluation
Exact BC Indices

Figure 3.4. Steps of Betweenness Centrality Evaluation

of paths: one without having any external vertices of the same community and the other one having them. Here our major source of reduced computation is from three facts shown in the earlier example where, $\delta_{ext_D}[u_2] = \delta_{ext_D}[u_2] + |A_{vc}|(\frac{\sigma_D[u_2]}{\sigma_D[u_3]} + \frac{\sigma_D[u_2]}{\sigma_D[u_3']})$*. It is possible* $\forall s \in A_{vc}$ *and for* $A \to D$ *shortest path,* s *has* A *as the only outgoing interface, else it would not have been in* A_{vc}*, by definition. Additionally, every* $s \in A_{vc}$ *will cause the same computation, hence the multiplication by* $|A_{vc}|$ *(dependent on source clusters). In addition, other incident shortest paths to* D *will use this computed* $\delta_{ext_D}[u_i], \forall u_i \in D_{vc}$ *for further computation. When the shortest path goes through more than one external vertex in the same community, for every such additional vertex, mutual and complemented mutual virtual clusters are considered along with the virtual cluster (for example in* $A \to B \to E$*,* $B_{vc}, 1, 1'$*). With both* $\delta_{Int}[v]$ *and* $\delta_{ext}[v]$ *for every node, Algorithm 4 evaluates the betweenness centrality index for them over the network.*

Theorem 3.3.3 Computation Cost in Centrality Evaluation: *The proposed approach for exact betweenness centrality evaluation incurs $O(|V|^2 + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$ computational cost, where $|V| = n$ is the number of nodes in the network.*

Proof 3.3.4 *The proposed exact algorithm operates in three main phases. It first evaluates the internal centrality of each of the $O(\sqrt{n})$ nodes in each of $O(\sqrt{n})$ communities. As the communities are dense, we assume each with $O(n)$ edges. Evaluating internal centrality in each community takes time $O(n^{\frac{3}{2}} + nlog\sqrt{n})$, as shortest path computation is bounded by Dijkstra's Algorithm and computation for centrality indices is bounded by $O(n^2)$. Repeating the process over $O(\sqrt{n})$ communities incurs $O(n^2 + n^{\frac{3}{2}} log\sqrt{n})$ time. Forming virtual clusters takes $O(c\sqrt{n})$ time, where c is a very small constant representing the number of external nodes in a community. The computation of clusters, the list of successors, predecessors, and the number of shortest paths from the external node to each of the internal nodes in their respective virtual clusters are embedded during the shortest path computation of internal centrality evaluation phase. Then it is a fact that total number of external nodes over all communities be $c\sqrt{n}$. Since the number of external edges across communities is very sparse, we consider it to be of the order of $K_1\sqrt{n}, c < K_1 < \sqrt{n}$. The second phase of the algorithm first finds shortest paths and associated external centrality indices over all pair of external nodes and takes $O(cKn + c^2nlog\sqrt{n})$ time. Finally, computation of external centrality indices needs $O(1)$ time per node in either cluster. There is c virtual clusters, 4 mutual and complemented mutual virtual clusters over a pair of virtual clusters. In the worst case $4c^2$ clusters will be involved in a community, resulting in total time of $O(4c^2\sqrt{n}) \sim O(\sqrt{n})$ over all communities. Let us assume that the size of clusters are not uniform to analyze the worst case, and the size of a cluster can be at most $O(\sqrt{n})$. This implies $O(n^{\frac{3}{2}})$ computations for a single community to others and $O(n^2)$ computations over all the clusters across all communities. In the third stage of computing total/global exact centrality indices, there is just one addition per node, with the resulting computation cost of $O(n)$. The total computation cost over all phases is thus bounded* *by* $O(|V|^2 + |V|^{\frac{3}{2}} log\sqrt{|V|})$. *Similarly, the community detection requires*$O(|V|^2 - mk^2|V|)$ *time. For both cases, this is the worst case over dense graphs where* $|E| \simeq |V|^2$. *Thus, computational cost of our proposed approach is bounded by* $O(|V|^2 + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$.

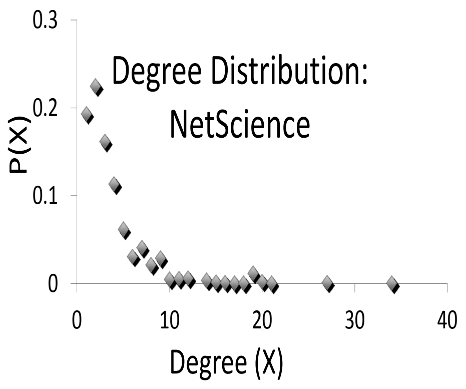
3.4. EXPERIMENTAL RESULTS

To validate our theoretical approach and to draw practical insight, we experimented with four networks: NetScience, twitter mention, power grid, and Internet [136]. We perform all our experiments using 2.3GHz Intel Core i5, 4 GB RAM, MAC10.7.3. From Table 3.1 twitter mention is the most dense and NetScience is the least dense networks. Further, they have large number of nodes of degree one ($\sim O(n)$ in most cases).

Table 3.1. Network Statistics for Four Networks

	NetScience	Twitter Mention	Power Grid	Internet
Number of Nodes	1859	3656	4941	22963
Number of Edges	2742	157727	6594	48436
Network Density	1.329	43.14	1.334	2.1
Average degree	3	86	2	4
Number of Nodes of Degree one	307	168	1226	7840

The degree distribution of the four application networks in Figure 3.5 and Figure 3.6 justifies power-law from linearity of the long tails and the straight line nature of the plots; thus, validates our approach, to start with degree one nodes and use right skewed property to form self and incremental accumulation. This also implies that we can use these four networks to study the performance of our algorithms.



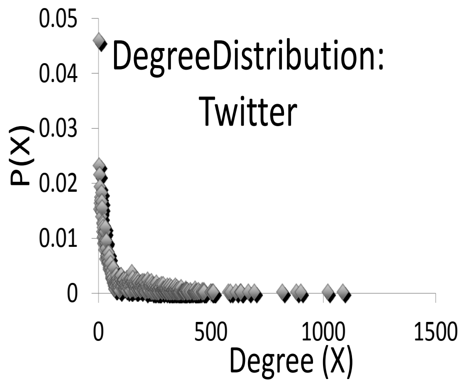
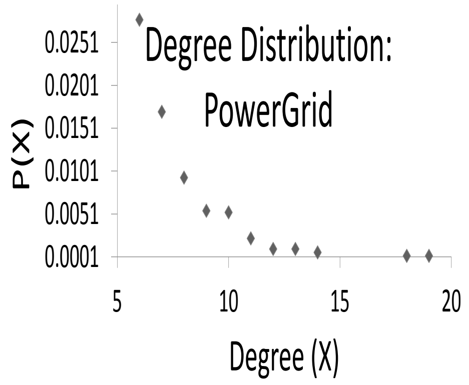


Figure 3.5. Degree Distribution of Two Networks



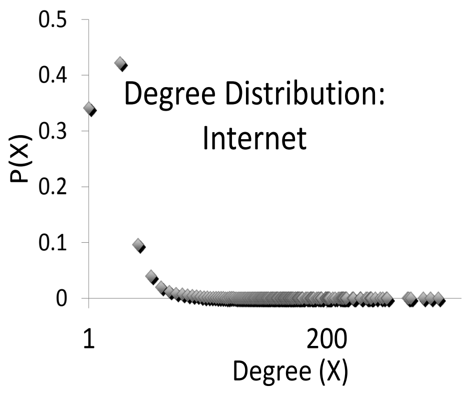
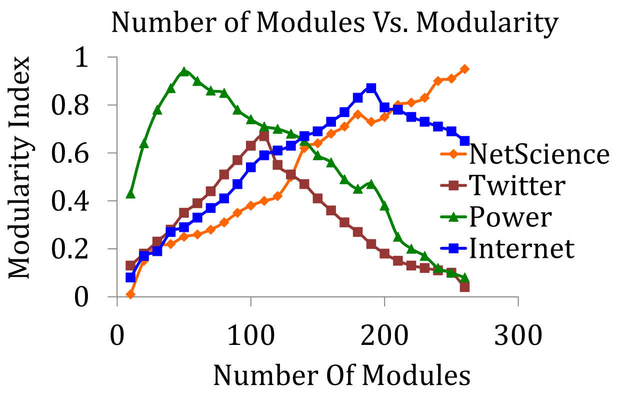


Figure 3.6. Degree Distribution of Two Networks

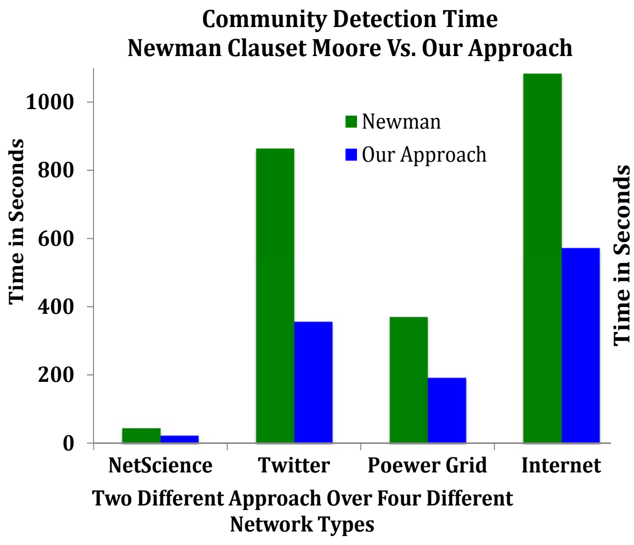
We use the community detection algorithm with and without constraint to evaluate the effectiveness and modularity index, as shown in Figure 3.7. The modularity in selecting $O(\sqrt{n})$ communities in either case remains unchanged. Further, we are not selecting community structure with maximum modularity, so the plot shows how effective they are to the best obtained index. Modularity index above 0.3 are considered good, justifying our community consideration. The Table in Figure 3.7 demonstrates the modularity index with the number of considered communities (say 150 in NetScience) in our and Newman et al. approach [30]. Our approach fares better and gives better modular structure.

The run time of the proposed approach and Newman et al.'s approach is shown in the first graph of Figure 3.8, demonstrating that the algorithm out performs the algorithm in [30]. For directed networks the proposed approach consider the impact factor α, β to



Number Of Community/ Modularity Index	Net Science (150)	Twitter (110)	Power (75)	Internet (250)
Newman Approach	0.61	0.41	0.73	0.54
Our Approach	0.62	0.58	0.85	0.69

Figure 3.7. Number of Modules Vs. Modularity in Four Networks and Number of Selected Modules Vs. Modularity Index



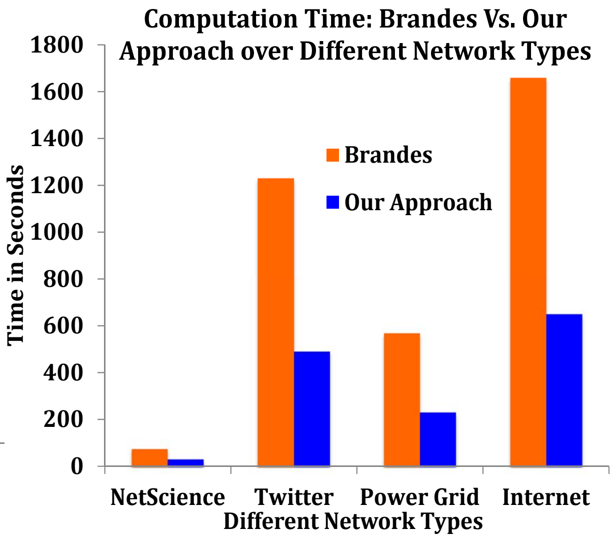


Figure 3.8. Runtime: Community Detection, Centrality Evaluation

be 0.85, 0.25 respectively, because at this value the best community structure with optimal modularity is obtained. The proposed exact betweenness centrality algorithm is also ran over the obtained structure and compared it with Brandes approach [21] approach. It also reflects the variation in computational cost relative to the modularity index, number of communities and network density. The closer the number and size of communities to $\sqrt{|V|}$ the better the performance (e.g. power grid network). Further, performance is also dependent on density of the network as evident between the Twitter network and the Internet.

3.5. SUMMARY

To sum up our work here, the key to the effectiveness of our approach is not only integration of structural properties, but also the fact that we are not putting any strict limit on the number and size of communities. Instead we limit it to $c'n$, where $c' << n$, thus giving liberty to explore the best possible community structure, as well as creating the opportunity to not let the computational cost evaluations in later stage being bottlenecked by any community. The proposed approach to evaluate exact betweenness centrality indices for all nodes incurs the computational cost of $O(|V|^2 + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$ over dense graphs in which $|E| \simeq |V|^2$, thus outperforming the existing exact algorithm in this domain. Similarly, our community detection approach integrates power law property along with techniques like incremental accumulation, self-accumulation, semi-local optimal node order selection, resulting in $O(|V|^2 - mk^2|V|)$ computation cost over dense graphs. Whereas, over sparse graphs the betweenness centrality evaluation will incur computational cost of $O(|V|^{\frac{3}{2}} + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$, where the number of edges in any community is $|E_{C_i}| \simeq \sqrt{|V|}$. Our theoretical and experimental analysis validates the better performance of our approach over existing approach.

4. PROBABILISTIC LINK PREDICTION OVER LARGE SCALE DYNAMIC NETWORKS

One of the most intriguing aspects, or the core challenges of network analysis is how links or interactions occur over time between node pairs. More so, whether we can design to have a model to accurately predict the occurrence of these links ahead of time, and with what accuracy. In contrast to the existing approaches, here we proposes a novel Markov prediction model that leverages the underlying network structure evolution over the time-varying graph of a large scale network. To incorporate the network structure evolution over microscopic and macroscopic time frames, the model considers the effect of multiple time scales in leveraging temporal analysis for link prediction. The analysis considers microscopic (fine-grained) and macroscopic (coarse-grained) time scales, along with associated local (links) and semi-global (clusters) structural evolution, respectively. The model takes into account correlated evolution and rate of evolution in selecting start and end nodes, and the corresponding interaction probability. We discuss the details in the following Subsections.

The following discussion is organized as follows. In Subsection 4.1, we present preliminary concepts and definitions used in link prediction strategy along with the formal problem definition. Subsection 4.2 describes our proposed Markovian model, the use of multiple time scales along with, correlated structural evolution and rate of evolution in link prediction, while Subsection 4.3 presents the proof that our edge selections approach leads to power-law degree distribution. Subsection 4.6 reports experimental results and finally a summary of our proposed approach and the results are offered in Subsection 4.7.

4.1. PRELIMINARY DEFINITIONS AND CONCEPTS: LINK PREDICTION

We represent the underlying application network graph $G(V,E)$ with communities and the observation time frame as, $G(V,E,C,[0,T+1])$, where V,E and C represent the set of vertices, edges and communities (clusters) respectively, $[0,T]$ is the training period and $[T,T+1]$ is the test interval. At the beginning, in absence of any edge, $|C| = |V|$. As stated earlier, in a temporal network, the network interactions are viewed as a set of discretized events over time axis. To have two-way temporal analysis, we visualize two temporal domains over time axis. Thus, we divide the time interval $[0,T]$ into uniform length divisions, and further, divide each of these uniform divisions into uniform length subdivisions. We visualize these as two distinct temporal domains over $[0,T]$, referred to as coarse-grained and fine-grained time domain, respectively.

Let the interval $[0,T+1]$ be divided into macroscopic time interval, called time slots $T_1,T_2,\ldots,T_i,T_{i+1},\ldots T_k$. Each of the time slot T_i is further divided into microscopic time interval, called time stamps $t_{1'},t_{2'},\ldots t_{i'},t_{i'+1},\ldots t_{k'}$, where time stamp reflects the smallest interaction time unit (say minute, hour, day or month) considered for an application. Let the corresponding subgraph over any time T_i and $t_{i'}$ be represented as $G_{T_i}(V_{T_i},E_{T_i},C_{T_i})$ and $G_{t_{i'}}(V_{t_{i'}},E_{t_{i'}},C_{t_{i'}})$ respectively, such that $G_{T_i},G_{t_{i'}} \subset G$ and $G_{t_{i'}} \subset G_{T_i}$; V_{T_i},E_{T_i},C_{T_i} and $V_{t_{i'}},E_{t_{i'}},C_{t_{i'}}$ are the respective set of vertices, edges and clusters, with $C^j_{T_i} \cap C^k_{T_i} = \emptyset$. For notational simplicity we represent the clusters $C^j_{T_i},C^k_{T_i} \in C_{T_i}$ as C^j_i and C^k_i respectively. We consider directed network graph, where any undirected application network is incorporated as having bidirectional edges between any pair of nodes. From here onwards, we use link or interaction or edge or contact interchangeably.

From our observation of the structural evolution in real world networks, we make the following model assumptions.

- Network evolution process is a sequence of states.
- The start state has interactions to reflect reality.

- Every subsequent state is based on the past states (called history states).

Formal Problem Formulation:

We formally define the link prediction problem as the mapping $f : (G_{T_i}, C_{T_i}, \forall i \in [0,T]) \rightarrow \{L_{i+1}\}$, where G_{T_i}, C_{T_i} are respectively the subgraphs and clusters over corresponding time slots T_is, and $\{L_{i+1}\}$ is the set of links in the subsequent time slot $T + 1$ (i.e. interval $[T, T+1]$). As we consider two distinct temporal axis, we consider $T+1$ as both subsequent time slot T_{k+1} and time stamp $t_{1'}$ in T_{k+1}.

We represent some specific notations used in solving link prediction problem in Table 4.1.

4.2. MODEL FOR LINK PREDICTION: MARKOV MODEL

Here, we discuss the details of our proposed stochastic prediction model, the steps involved and how they work in tandem in computing prediction probabilities, i.e. how to formulate the initial prediction probabilities (i.e., the elements of the transition probability matrix), and then proceed to describe the steps to populate the elements of $M(T_i)$ (also, $M(t_i)$) in subsequent steps.

4.2.1. General Model Definition and Initialization. This Subsection discusses high-level integration of the temporal network with our proposed model, our model architecture, and its initialization process.

We consider dynamic network evolution as a transition process over sequence of network states, going by the intrinsic evolution process in real world networks, which are referred as the corresponding temporal graphs over time domain.

In Figure 4.1, the temporal graphs $G_{t_1}, G_{t_2}, G_{T_i}$ represent network states over time stamps t_1, t_2 and time slot T_i respectively, where time slot T_i is divided into two time stamps t_1 and t_2. The green, blue and orange dots represent nodes; where as the directed and bi-directed arrows reflect interaction between node pairs.

Table 4.1. Notation Explanation

Notation	Description
s	Set of states of the evolution process that are kept track
$[0,T]$	Observation period
T_i	Time slots in $[0,T]$
$t_{i'}$	Time stamps in T_i
$t_{k'}$	Last time stamp over all $t_{i'}$ w.r.t. each T_i
$t_{1'}$	First time stamp over all $t_{i'}$ w.r.t. each T_i
d_i	Number links for node v_i
d_j	Number links for node v_j
$d_{in}(v)$	In-degree of node v at time stamp $t_{i'}$
$d_{out}(v)$	Out-degree of node v at time stamp $t_{i'}$
$d(v)$	Total degree of node v at time stamp $t_{i'}$
C_i	Set of clusters in time slot T_i
C_i^j	A cluster j, with $C_i^j \subset C_i$
$MAXd(v)$	Maximum degree of node v
$MIN(a,b)$	Returns minimum of the two parameters a and b

The network state corresponding to any time interval, t_1, t_2 *or* T_i (as shown within green, blue or orange rectangles) reflects the interactions, i.e. the edges between node pairs during that specific period. Further, the subgraph of the temporal graph that is inside each dotted quadrilateral, corresponds to the clusters formed over that time slot (here, T_i). The network evolution is interpreted as a transition process from one network state to another. The correct subsequent state depends on the accuracy of the state transition probabilities that in turn is dependent on the past states. From the temporal graph representation underlying a network state, the transition probability over subsequent temporal graphs is the probability of links over each node pair. Thus, the resulting transition probability between states is a transition probability matrix over set of links.

4.2.2. General Model Definition and Initialization. This Subsection discusses high-level integration of the temporal network with our proposed model, our model architecture, and its initialization process.

We consider dynamic network evolution as a transition process over sequence of network states, going by the intrinsic evolution process in real world networks, which are referred as the corresponding temporal graphs over time domain.

In Figure 4.1, the temporal graphs $G_{t_1}, G_{t_2}, G_{T_i}$ represent network states over time stamps t_1, t_2 and time slot T_i respectively, where time slot T_i is divided into two time stamps t_1 and t_2. The green, blue and orange dots represent nodes; where as the directed and bi-directed arrows reflect interaction between node pairs. The network state corresponding to any time interval, $t_1, t_2 \; or \; T_i$ (as shown within green, blue or orange rectangles) reflects the interactions, i.e. the edges between node pairs during that specific period. Further, the subgraph of the temporal graph that is inside each dotted quadrilateral, corresponds to the clusters formed over that time slot (here, T_i). The network evolution is interpreted as a transition process from one network state to another. The correct subsequent state depends on the accuracy of the state transition probabilities that in turn is dependent on the past states. From the temporal graph representation underlying a network state, the transition probability over subsequent temporal graphs is the probability of links over each node pair. Thus, the resulting transition probability between states is a transition probability matrix over set of links. Thus, we propose a Markov model, with its states referring to network states and the inter-state transition probability represented by the Markovian transition matrix over corresponding temporal graphs . For most accurate prediction probability, the state space of the Markov model encapsulates the past s states within it.

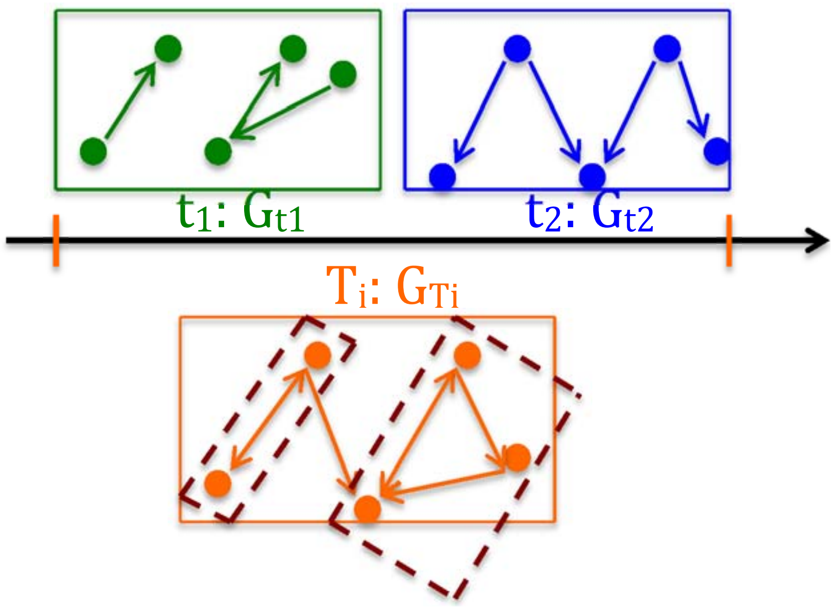


Figure 4.1. Time-Varying Graph of The Dynamic Network States

Over the two distinct temporal domains we consider local and semi-global structural evolution (i.e. link and clusters respectively). Let $M(T_i)$ represent the Markovian transition matrix model. In transition from T_i to T_{i+1}, called time slot T_{i+1}, for each $t_j \in T_{i+1}$, we evaluate the corresponding renewed transition probability, owing to the changes in link structure. At each time stamp t_i, for temporal graph G_{t_i}, we have the associated transition matrix M_{t_i}. Over time stamp t_i, we consider and compute the local structural evolution or link evolution. Whereas, we consider global structural evolution or evolution of clusters w.r.t. time slots. The clusters and cluster evolution probability is computed once per time slot , at the end of its respective last time stamp t_k. For example, in Figure 4.1 the clusters are computed over T_i once, at the end of its last time stamp t_2. Though, the initial time slot T_1 is a diversion from this. We consider edge weight, based on the number of interactions (say, number of papers they co-authored etc.) in that slot (initial time slot), to compute the clusters. The transition probabilities are evaluated based on the procedures in the following sub-sections, and the ij^{th} entry of the matrix $M(T_i)$ represent the conditional probability that a node with d_i links will interact with a node with d_j links. Let us represent the link probability between a node pair with degree d_i and d_j in cluster C_i^j and C_i^k respectively as $p_{d_i d_j}^{C_i^j C_i^k}$.

In Figure 4.2, $C_i^1, C_i^2, \ldots, C_i^k$ are the row and column sub-matrices, representing the corresponding clusters in the underlying time-varying network graph. Each sub-matrix has number of rows and columns equal to the number of vertices in the corresponding cluster. Elements in the main diagonal are the prediction probability for future interactions between pair of nodes within any cluster, whereas probabilities in the upper and lower-diagonal matrix elements correspond to pair of nodes in distinct clusters, and are called intra- and inter-cluster link prediction probabilities respectively.

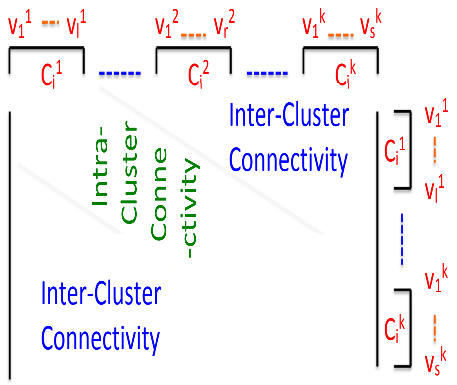


Figure 4.2. Markovian Prediction Matrix Model

Besides the proposed model framework, the mathematical formulae underlying the model, the approach for model parameter evaluation and interaction probability computation, is inspired by intrinsic operations over real world networks. For example, over academic collaboration network, a CS researcher can collaborate with Physicists to have an one time paper in Quantum Computing area, or have a set of papers over time slot(s). Here, the individual-one time collaboration corresponds to links over different time stamps, whereas the later one reflects changed community structure over time slot. Thus, comes the evolution of links over time stamps and evolution of clusters over time slots, as described in Subsection 4.2.3.

Further, it is evident from online social networks (e.g. Facebook, Twitter etc.) that in any interaction, nodes that reply, re-tweet etc. are considered initiator of the action. In response, receiving nodes may reply, re-tweet (to others), like a post etc. This type of cause-effect-based interactions or occurrence of pairs of interactions, gives rise to the concept of temporally correlated evolution over both time scales, as is used in Subsections 4.2.3.1, 4.2.3.2 and subsequently in 4.2.4. More so, a person with more collaborators or friends gets opportunity for more interactions; similarly persons' active duration attracts others to interact with. In addition, receiver communication with others depends on its set of associates (degree), its duration and interaction frequency. In fact these basic attributes are used in the detailed formulation of Subsections in 4.2.3, and 4.2.4.

4.2.2.1. Effect of node degree in real world networks. Here we give a simple degree based prediction probability computation. In real world application networks, nodes of high degree prefer to interact with other high degree nodes. Though, there are also high degree nodes interacting with other low degree nodes, justifying the right-skewed degree distribution [31] in them. So, to some extent interaction between pair of nodes is dependent on both source and destination node degree or how much they communicate. In fact, the Pearson correlation coefficient of the degrees of the end points of an edge is given by [101]. In order to reflect the explicit correlations we need the distribution of interaction $p(d_i)$ and the conditional probability that a node of degree d_i interacts with a node of degree d_j, i. e., $p(d_j|d_i)$, $\sum_{d_i} p(d_i) = 1$ and $\sum_{d_j} p(d_j|d_i) = 1$. In the simplest Markovian model, each of the matrix elements can be represented as $p(d_i) \times p(d_j|d_i)$.

The following Subsection presents our approach to initialize the matrix elements of $M(T_i)$ (also, $M(t_i)$).

4.2.2.2. Initializing Markov matrix elements. Over time stamp t_i, the changes in the elements of M_{T_i} (also M_{t_i}) are effect of local evolution, where as over time slot T_i the effect of global evolution is reflected in the elements of M_{T_i}. If we start with the network without any interactions (say at $t = 0$), then interactions between pairs of nodes occur uniformly at random and can be selected likewise. Where as, in presence of interactions from the beginning, we describe the initialization of matrix elements below.

Initializing $M(t_1)$ in Presence of Interactions:

At time stamp t_1 corresponding to T_1, we have the first temporal graph and is the start of our observation. At this point we do not have any past interaction information to derive edge weights. So, we use node attributes (say, authors in same research field, same University, and people in the same geographic location, affiliation to same interest/club) to form the initial tentative clusters, and the resulting network graph has now both intra- and inter-cluster interactions.

We now compute the probabilities within cluster $p_{d_i d_j}^{C_1^j C_1^j}(t_{1'})$ and across cluster $p_{d_i d_j}^{C_1^j C_1^k}(t_{1'})$. Let v_i, v_j be the vertices with d_i, d_j number of links respectively. When $v_i, v_j \in C_1{}^j$, let $p(d_i)$ and $p(d_j)$ be their associated degree distributions w.r.t. $C_1{}^j$. Thus, we define intra-cluster initial link probabilities as:

$$p_{d_i d_j}^{C_1^j C_1^j} = p(d_i)p(d_j)\frac{d_{in}(v_i)}{\sum_{v \in C_1^j} d_{in}(v)} \times \frac{d(v_j)}{\sum_{v \in C_1^j} d(v)}. \tag{4.1}$$

In a similar manner, we initialize the initial inter-cluster link probabilities given by the following Equation:

$$p_{d_i d_j}^{C_1^j C_1^k} = p(d_i)p(d_j)\frac{d_{in}(v_i)}{\sum_{v \in C_1^j} d_{in}(v)} \times \frac{d(v_j)}{\sum_{v \in C_1^k} d(v)}, \tag{4.2}$$

where nodes $v_i \in C_1^j$, $v_j \in C_1^k$; corresponding degree distribution $p(d_i)$ and $p(d_j)$ over modules C_1^j and C_1^k respectively.

The weighted in degree reflects a node's position as a start node, whereas weighted total degree on end node reflects that a leaf node may not have any out degree, but may still be connected. In case the underlying clusters are ignored at the start of observation, the node degrees need to be weighted over the whole network in contrast to their respective cluster.

Now, we describe the correlation associated with local and global structural evolution and their associated rate of evolution, as they effect link prediction and are parameters in our subsequent steps. Subsequently, we discuss ways to populate the elements of transition probability matrix $M(T_i)$ (also, $M(t_i)$).

4.2.3. Markov Model: Evolution Metric Computation. In this Subsection we discuss the procedures for evaluating parameters that are in turn used in computing link prediction probabilities in Subsection 4.2.4.

4.2.3.1. Correlated evolution of links: local correlation. We consider the set of time-varying graphs over time stamps $t_{i'} \in T_j, \forall T_j \in [0,T]$. We compute the correlated evolution of links w.r.t. the set of time-varying graphs over each time stamp. Over subsequent time stamp pairs $t_{i'}, t_{j'} \in T_j$, we have $L(t_{i'})$ and $L(t_{j'})$ as the corresponding link sets over time-varying graphs $G_{t_{i'}}$, $G_{t_{j'}}$ respectively. We determine correlated link evolution over successive time stamps as follows:

$$Corr_{Link}(t_{j'}, t_{i'}) = \frac{|L(t_{i'}) \cap L(t_{j'})|}{|L(t_{i'}) \cup L(t_{j'})|}. \tag{4.3}$$

Similarly, over pairs of successive time stamps in each time slot we consider average correlated evolution of the corresponding pairs of time-varying graphs. We evaluate average correlated evolution of links over each slot T_j as given by the following Equation.

$$\zeta_{link} = \frac{\sum_{t=t_{1'}}^{t_{k'}} Corr(t, t+\delta t)}{t_{k'} - t_{1'}}, \tag{4.4}$$

where $\delta t = t_{j'} - t_{i'}$, with $t_{i'}$ *and* $t_{j'}$ being the two successive time stamps over any time slot T_i. $1 - \zeta_{link}$ is the average number of links changed over pairs of successive temporal graphs in any time slot T_i. Likewise, the recursive correlated evolution of links over pair of time-varying graphs corresponding to respective time stamps in any time slot is given by Equation 4.5.

$$Corr_{Link}^{recursive_r}(t_j, t_i) = \frac{|L(t_i) \cap L(t_i + \delta t^r)|}{|L(t_i) \cup L(t_i + \delta t^r)|}, \tag{4.5}$$

where r represents the number of time intervals between considered time-varying graph pairs. in other words time stamp intervals between time stamps t_i and t_j.
We define average recursive correlated evolution of links over any pair of time-varying graphs in any time slot T_i as:

$$\zeta_{link}^{recursive_r} = \frac{\sum_{t=t_1}^{t_k} Corr_{Link}(t, t + \delta t^r)}{t_k - t_1}, \tag{4.6}$$

where $1 - \zeta_{link}^{recursive_r}$ is the average number of links changed over those time-varying graph pairs. Over successive time-varying graphs we have $r = 1$.

As link evolution can be visualized as a cascaded process that takes place in number of successive steps, we consider pairwise link occurrence rate. Likewise, for each link in a time-varying graph we determine the number of times a link l_j follows a link l_i in $t_{i'}$, i.e. $F_{(l_i,l_j)}(t_{i'})$. For example, let A, B, C be three network nodes. Let, l_i is the contact between A and B; l_j is the contact between B and C. (Say, over Twitter application network, let A tweets a message to B and B tweets to C.) Thus, $F_{(l_i,l_j)}(t_{i'})$ is the number of times B communicates with C, following As' communication with B in $t_{i'}$ (say, a day). The weighted measure of the pairwise link occurrence over a single time-varying graph $G_{t'_i}$ is given by:

$$F_{(l_i,l_j)}^{weighted}(t_{i'}) = \frac{F_{(l_i,l_j)}(t_{i'})}{\sum_{\forall l_{i'} \neq l_i} F_{(l_i,l_{i'})}(t_{i'})}, \tag{4.7}$$

Relative to successive time stamps, the correlated pairwise link occurrence rate over successive time-varying graphs $G_{t'_i}, G_{t'_j}$ is denoted as $F_{Corr}(t'_i, t'_j)$. We define $F_{Corr}(t_{i'}, t_{j'})$ as:

$$F_{Corr}(t_{i'}, t_{j'}) = MIN\{F_{(l_i,l_j)}^{weighted}(t_{i'}), F_{(l_i,l_j)}^{weighted}(t_{j'})\}, \tag{4.8}$$

Similarly, with $\delta t = t_{j'} - t_{i'}$, average correlated pairwise link occurrence is defined as:

$$F_{Corr}^{Avg} = \frac{\sum_{t=t_{1'}}^{t_{k'}} F_{Corr}(t, t + \delta t)}{t_{k'} - t_{1'}}. \tag{4.9}$$

4.2.3.2. Correlated evolution over clusters: semi-global correlation. We determine the clusters corresponding to each $G_{T_i}, \forall T_i \in \{T_1, \ldots, T_k\}$, based on the edge weight over T_i. As mentioned earlier, we create the initial cluster at time stamp t_1, corresponding to time slot T_1. We compute the cluster over time slot T_1 i. e. over interval $[0, T_1]$, after its last time stamp say, t_k occurs. We use the community formation algorithm in [37] for this and the resulting clusters refers to the underlying graph structure in G_{T_1}.

After 1st time slot T_1, the subsequent clusters are created over T_i (instead of t_is), for which the incremental variant of the community formation algorithm in [37] is used over respective G_{T_i}s.

In the incremental version, for the nodes involved in interaction over a slot, we compute the relative measure of the interaction weight of a node with other nodes within its cluster, to nodes outside its cluster for that slot. To be more precise, the relative weight measure is computed with the nodes interaction weight within its cluster to interaction weight with every other cluster, taken the other clusters one at a time, of which the interacting nodes are part of. If the interaction weight for both the compared clusters is same, then the node belongs to the cluster with which it has higher number of interacting nodes for that slot; else, it joins the cluster with higher interaction weight with it, in the current time slot. Incase both weight and number of nodes that it interacted with are same over different clusters, it remains part of the current cluster. This way the clusters in the successive time slots are formed. We present the steps for our incremental algorithm in Algorithm 5.

Clusters in the successive time slots are formed as discussed above, but what is important for our work here is, how the clusters are correlated over successive time slots.

We compute correlation coefficient for both within and across cluster. For a cluster C_i^k in time T_i, we evaluate its correlation with two successive clusters C_i^k and C_{i+1}^k in respective time slots T_i and T_{i+1} as, $\frac{|C_{T_i}^k \cap C_{T_{i+1}}^k|}{|C_{T_i}^k \cup C_{T_{i+1}}^k|}$. If more than one cluster merges into a single cluster over successive intervals, or a cluster divides into two or more clusters over successive intervals we compute their joint correlation effect as $\prod_{\forall j} \frac{|C_{T_i}^k \cap C_{T_{i+1}}^j|}{|C_{T_i}^k \cup C_{T_{i+1}}^j|}$, where cluster

C_i^k evolves into clusters C_{i+1}^js or clusters C_{i+1}^js merge into cluster C_i^k over successive time interval. For any pair of clusters that do not either evolve from or to each other over successive intervals it is simply computed as, $\frac{|C_{T_i}^k \cap C_{T_{i+1}}^j|}{|C_{T_i}^k \cup C_{T_{i+1}}^j|}$. Thus, the total correlated evolution among all pairs of successive clusters is given by:

$$Corr_{Cluster}(T_i, T_{i+1}) = \sum_{k,j} \Big[\frac{|C_{T_i}^k \cap C_{T_{i+1}}^j|}{|C_{T_i}^k \cup C_{T_{i+1}}^j|} + \prod_{\forall j} \frac{|C_{T_i}^k \cap C_{T_{i+1}}^j|}{|C_{T_i}^k \cup C_{T_{i+1}}^j|} + \frac{|C_{T_i}^k \cap C_{T_{i+1}}^k|}{|C_{T_i}^k \cup C_{T_{i+1}}^k|} \Big]. \tag{4.10}$$

We evaluate average correlated evolution over successive time slots is given by:

$$\zeta_{cluster} = \frac{\sum_{T_i=T_1}^{T_k} Corr_{Cluster}(T_i, T_{i+1})}{T_k - T_1}. \tag{4.11}$$

In the above, $Corr_{Cluster}(T_i, T_{i+1})$ is evaluated over successive time slots.

As in recursive correlated evolution of links, we can compute $\zeta_{cluster}^{recursive_r}$ w.r.t clusters, where r is the temporal interval between pairs of temporal graphs G_{T_i}, G_{T_k}, we omit it here for brevity. In the above, $Corr_{Cluster}(T_i, T_{i+1})$ is evaluated over successive time slots and has $r = 1$.

Similar, to the correlated evolution of individual links we determine the correlated evolution of the same cluster over pairs of G_{T_i}s and the average correlated evolution of clusters over all pairs of G_{T_i}s (let it be, $\zeta_{cluster}$). Likewise, we can determine $\zeta_{cluster}^{recursive_r}$, where r is the temporal interval between pairs of temporal graphs G_{T_i}, G_{T_k}.

For a cluster C_i^k we evaluates its correlation over successive time slots T_i and T_{i+1}, given by $\frac{|C_i^k \cap C_{i+1}^k|}{|C_i^k \cup C_{i+1}^k|}$, representing ratio of common interactions to all interactions for the cluster over successive time slots. If more than one cluster merge into a single cluster over successive intervals, or a cluster divides into two or more clusters over successive intervals we compute their joint correlation effect as $\prod_{\forall j} \frac{|C_i^k \cap C_{i+1}^j|}{|C_i^k \cap C_{i+1}^j|}$, where cluster C_i^k evolves into clusters C_{i+1}^js or clusters C_{i+1}^js merge into cluster C_i^k over successive time interval. For any pair of clusters that do not either evolve from or to each other over successive intervals, it is simply computed as $\frac{|C_i^k \cap C_{i+1}^j|}{|C_i^k \cap C_{i+1}^j|}$. Thus, the total correlated evolution of a cluster C_i^k over successive intervals is given by, $Corr_{Cluster}(T_i, T_{i+1}) = \sum_j \Big[\frac{|C_i^k \cap C_{i+1}^j|}{|C_i^k \cap C_{i+1}^j|} + \prod_{\forall j} \frac{|C_i^k \cap C_{i+1}^j|}{|C_i^k \cap C_{i+1}^j|} + \frac{|C_i^k \cap C_{i+1}^k|}{|C_i^k \cup C_{i+1}^k|}\Big]$.

Algorithm 5 Incremental Community Formation Algorithm (ICF)

```
Require: Graph (initial clusters formed using [37]) in $t_a$ of $T_1$. Interaction weights (accumulated over node pairs) over the time slots ($T_i$s). Graph with clusters from the last time slot (say, $G_{T_i}$).
Ensure: Clusters over current slot ($G_{T_{i+1}}$) with best modularity.
  Form initial clusters over $G_{T_1}$ with interaction weights over $T_1$ (using the community formation algorithm [37]).
  for $\forall$ $T_i$s: do
    for $\forall$ Interacting node $T_i$: do
      Compute Relative Interaction Weight.
      Compute Maximum Relative Interaction Weight relative to all such clusters in $T_i$.
    end for
  end for
  if Maximum Relative Interaction Weight $= 1$ and number of nodes that it interacts with in these clusters are unequal: then
    Node is assigned to the cluster having fewer number of nodes it interacted with.
  else if Maximum Relative Interaction Weight $= 1$: then
    Node is assigned to the cluster with fewer number of nodes.
  else if Maximum Relative Interaction Weight $< 1$: then
    Node is assigned to the cluster having higher interaction weight with it.
  else if Maximum Relative Interaction Weight $> 1$: then
    Node remains unchanged.
  end if
```

We evaluate average correlated evolution of a cluster over successive time slots as, $\zeta_{cluster} = \frac{\sum_{T_i=T_1}^{T_k} Corr_{Cluster}(T_i, T_i+\Delta T)}{T_k - T_1}$, where $\Delta T = T_{i+1} - T_i$.

In a similar manner, we can compute $\zeta_{cluster}^{recursive_r}$, where r is the time interval between pairs of temporal graphs G_{T_i}, G_{T_k}. In the above, $Corr_{Cluster}(T_i, T_{i+1})$ is evaluated over successive time slots and has $r = 1$.

Similar, to the correlated evolution of individual links we determine the correlated evolution of the same cluster over pairs of G_{T_i}s and the average correlated evolution of clusters over all pairs of G_{T_i}s (let it be, $\zeta_{cluster}$).

Likewise, we can determine $\zeta_{cluster}^{recursive_r}$, where r is the temporal interval between pairs of temporal graphs G_{T_i}, G_{T_k}.

In a similar manner, we can compute $\zeta_{cluster}^{recursive_r} = \frac{\sum_{T_i=T_1}^{T_k} Corr_{Cluster}(T_i, T_i+\Delta T^r)}{T_k-T_1}$, where r is the number of time interval between pairs of time-varying graphs G_{T_i} *and* G_{T_k}.

With the correlation parameters in hand, we now compute the probabilities for the selection of the start and end nodes, and the parameters in evaluating the interactions and probability of interactions. In the above generalized model, we have discussed recursive version of parameters that helps us analyze the scalability of the model and also relate with computational cost. Due to brevity, we refrain from the discussion of scalability and computational cost, rather we consider $r = 1$ for our experimental analysis, excluding the recursive analysis.

Before we discuss the computation of interaction probabilities, we give a schematic presentation (Figure 4.3) of the steps involved in the prediction process and their direct (the solid arrows) and indirect (the dotted arrows) inter-relationship.

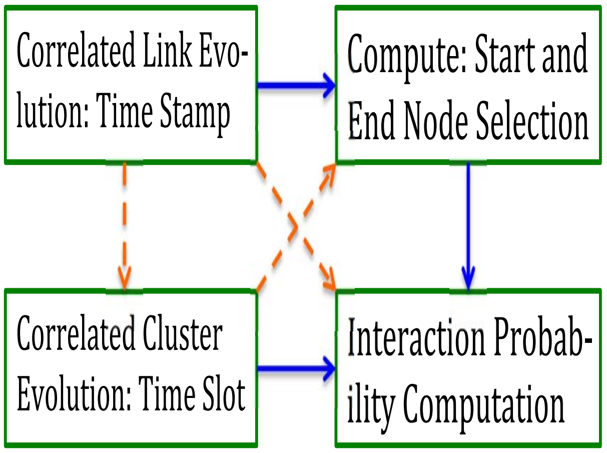

Correlated Link Evolution: Time Stamp
Compute: Start and End Node Selection
Correlated Cluster Evolution: Time Slot
Interaction Probability Computation

Figure 4.3. Schematic Presentation: Inter-relationship Among Prediction Steps

With correlation parameters in hand, we now compute the probabilities for start and end node selection, and the interaction probability, using the parameters evaluated in the previous Subsections.

4.2.4. Markov Model: Computation of Interactions.

Here, we describe the steps involved in computing interactions, i.e. procedure for selecting start and end nodes, and computing the interaction probability using the parameters evaluated in the previous Subsection.

4.2.4.1. Selection of start and end nodes.

We first present the procedure for selection of start and nodes.

Selection of Start Node (Initiator):

The probability that node v becomes initiator at time $t_{i'+1}$ be represented as $q^{v}_{start}(t_{i'+1})$. For any node $v \in C^{j}_{i}$, v to be an initiator at time $t_{i'+1}$ is dependent on its influence and fraction of its neighbors that are initiators at time $t_{i'}$, the correlated evolution of its outgoing links, and average rate of pairwise link creation.

If $\rho_v(t_{i'})$ is the fraction of start nodes (initiators) incident on node v at time $t_{i'}$. Then $\rho_v(t_{i'}) = \frac{d_{in}(v)}{\sum_{v_i} d_{out}(v_i)}$, evaluated w.r.t. t'_i, v_i is neighbor of v. Further, for any node v, its influence depends on the relative weight of its degree and distribution of its node degree. Relative weight of node v's degree at time $t_{i'}$ is $\frac{d(v)}{\forall_{v_i \in C} MAX d(v_i)}$ and its degree distribution is $p(d(v))$. Thus, start node selection probability is computed as follows:

$$\begin{aligned} q^{v}_{start}(t_{i'+1}) = \rho_v(t_{i'}) \times & \frac{d(v)}{\forall_{v_i \in C} MAX d(v_i)} \\ & \times p(d(v)) \times \zeta_{link} \times d_{out}(v) \\ & \times F^{Avg}_{Corr} \times |v \in l_i \cap l_j|, \end{aligned} \tag{4.12}$$

where $l_i \cap l_j$ finds the common node between the two links at time $t_{i'}$, the correlated evolution of outgoing links, and average rate of pairwise link creation for node v are $\zeta_{link} \times d_{out}(v)$ and $F^{Avg}_{Corr} \times |v \in l_i \cap l_j|$ respectively. When $d_{out}(v)$ at time $t_{i'}$ is 0 or $|v \in l_i \cap l_j| = \phi$ at time $t_{i'}$, the partial multiplicative terms in the above are substituted with $1 - \zeta_{link}$ and $1 - F^{Avg}_{Corr}$ respectively.

Selection of End Node (Receiver):

Probability of node v to be selected as end node, denoted as $q^{v}_{end}(t_{i'+1})$, depends on its affinity to start node, the correlated evolution of its incoming links, and average rate of pairwise link creation.

Let $a_{uv}(t_{i'})$ be the affinity between node u and v at time $t_{i'}$, where affinity is inverse of the number of hops between u and v. Thus, the probability of a node being selected as end node given by:

$$\begin{aligned} q^{v}_{end}(t_{i'}+1) = a_{uv}(t_{i'}) \times \zeta_{link} \times d_{in}(v) \times F^{Avg}_{Corr} \\ \times (F(l_i,(u,v)) + F((u,v),l_k)), \end{aligned} \tag{4.13}$$

where (u,v) denotes the respective link and $(l_i,(u,v))$ represent (u,v) follows link l_i. When $d_{in}(v) = 0$, and $(F(l_i,(u,v)) + F((u,v),l_k)) = 0$ at time $t_{i'}$, the partial multiplicative terms in the above are substituted with $1 - \zeta_{link}$ and $1 - F^{Avg}_{Corr}$ respectively. Number of hops is sufficient to limit to five but due to small world concept, during experimental evaluation we find a limit of three hops works fine.

4.2.4.2. Effect of node age and node degree. The interaction of a node for next time interval (say, $t_{i'+1}$) is dependent not only on its duration of presence in the network, but also on its influence/degree in current time. We evaluate a node's duration of presence in the network as $t_p = (t_{i'+1}) - t_a$, where t_a is the time node v start to be part of the network. As $t_{1'}$ is the start of the observation time, the observation duration of the node is $t_o = (t_{i'+1}) - t_{1'}$; thus, the weighted age of the node become $t^{weighted}(v) = \frac{t_p}{t_o}$.

Now, at time $t_{i'}$, the weighted node degree is, $d^{weighted}(v) = \frac{d(v)}{2E_{t_{i'}}}$, where $d(v)$ is node degree and $E_{t_{i'}}$ is total number of edges in the network. To incorporate their mutual effect ψ^{joint}, we consider their joint weighted parameter to predict the occurrence of interaction for node v, thus:

$$\psi^{joint}(v) = t^{weighted}(v) \times d^{weighted}(v). \tag{4.14}$$

The weighted degree of a node is computed relative to the whole network, instead of its respective cluster.

4.2.4.3. Global effect in link prediction. In computing global effect, we consider time slot (i.e. T_is) as the concerned time interval of our time-varying graph, instead of time stamps. First, we consider global structural parameter like modularity index in link formation. At any time T_i, let the number of edges in a cluster C_i^j be E_i^j, the total number of edges in the network be E_{T_i}, and the number of edges that a cluster C_i^j have with other clusters (C_i^k) is $\sum_{C_i^k} E_i^{jk}, \forall k \neq j$. We compute modularity index [98] (Q_i^j) as, a relative measure of the number of edges within C_i^j and across C_i^j. Thus, $Q_i^j = \frac{E_i^j}{E_{T_i}} - \left(\frac{E_i^{jk}}{E_{T_i}}\right)^2$. We formulate the probability of an edge over nodes $u, v \in C_i^j$ (i.e. $\phi^{C_i^j C_i^j}(uv)$) to be dependent on both modularity index, correlated evolution over clusters and influence of the respective nodes. Thus we have:

$$\phi^{C_i^j C_i^j}(uv) = Q_i^j \times \zeta_{cluster} \times \frac{d(u)}{2E^j} \times \frac{d(v)}{2E^j}. \tag{4.15}$$

Further, for a pair of nodes across distinct clusters, we evaluate the probability of a linkage as:

$$\begin{aligned} \phi^{C_i^j C_i^k}(uv) = (1 - \zeta_{cluster}) \times \frac{E_i^{jk}}{\sum_{\substack{l \neq i \text{ and } m \neq j \\ l \neq j \text{ and } m \neq i}} E_i^{lm}} \\ \times \frac{E_i^{jk}(u,v)}{\sum_{\forall u' \neq u \text{ and } v' \neq v} E_i^{jk}(u',v')}, \end{aligned} \tag{4.16}$$

here on the right hand side, the second term shows the connection across these two clusters relative to other cluster pairs, and the last term represents the number of links between (u, v) relative to other links across the cluster pair.

In the above discussions we have omitted $\zeta_{link}^{recursive_s}, \zeta_{cluster}^{recursive_s}$ and their successive complements in the discussion of procedures due to brevity and clarity.

We analytically explain the underlying evolution or the cascaded interaction, i.e. interactions effecting succeeding interactions. Let $\rho_{d_i}^{C_t^j}(0)$ define the initial fraction of degree-d_i nodes in community C_t^j that are having interactions at initial time stamp, i.e at time t_1. Let $\Gamma_{d_i}^{C_t^j}$ represents the number of nodes of degree d_i in community C_t^j with out-degree ≥ 1 at time t, acting as initiators. Thus, $\rho_{d_i}^{C_t^j}(0) = \Gamma_{d_i}^{C_t^j} / \sum_{d_i} \Gamma_{d_i}^{C_t^j}$. Let us define the probability that a degree-d_i node in community C_t^k is about to form an interaction at time step t as $q_{d_i}^{C_t^j}(t)$, where at least one of its neighbors is not forming any interaction.

Probability that the neighbor of a non-initiator node of degree d_i in community C_t^j is going to form a new interaction in time instance $(t+1)$ is given by:

$$\tilde{q}_{d_i}^{C_t^j}(t+1) = \frac{\sum_{C_t^k} \sum_{d_j} p_{d_i d_j}^{C_t^j C_t^k}(t) q_{d_j}^{C_t^k}(t)}{\sum_{C_t^k} \sum_{d_j} p_{d_i d_j}^{C_t^j C_t^k}(t)} \tag{4.17}$$

The probability that a degree-d_i node in community C_t^j will initiate an interaction at time interval t is given by Equation (4.18).

$$\begin{aligned}\rho_{d_i}^{C_t^j}(t) = \rho_{d_i}^{C_t^j}(0) + (1 - \rho_{d_i}^{C_t^j}(0)) \times \sum_{j=0}^{k} \binom{k}{j} \\ \times (\tilde{q}_{d_i}^{C_t^j}(t-1))^j (1 - \tilde{q}_{d_i}^{C_t^j}(t-1))^{k-j}\end{aligned} \tag{4.18}$$

4.2.4.4. Temporal dependency on link . We have temporal graphs over each time stamp, corresponding to each time slot in $[0,T]$. For each link $l \in t_i$, let Z be the random variable reflecting its time dependent occurrence or the interval between its occurrence. Let us assume that $\forall t_i \in T_j$, we gather the occurrence interval for all links over G_{t_i}. Let us represent it by $\{t'_0, t'_1, \ldots, t'_{k-1}\}$. Let I be the maximum interval, and number of times a link occurs is μ, then the i^{th} time interval is given by $t'_i \in [0, \lfloor \frac{I}{\mu} \rfloor], \forall i \in \{0, k-1\}$. From the observed data values for intervals, let r_j represent the number of times a specific interval appears in I, $0 \leq j \leq \lfloor \frac{I}{\mu} \rfloor$. Thus, the probability that the interval for the link occurrence will be j, is given by $\frac{r_j}{k}$.

Concisely, our approach in Section 4.2 performs the following steps to compute prediction probability and select suitable links for subsequent state. It initializes the Markovian transition matrix using the approach in Subsubsection 4.2.2.2, at the start of the observation ($t_{1'}$), over $G_{t'_1} \in G_{T_1}$. For predicting links in the subsequent interval, it selects the start node using the procedure in Subsection 4.2.4.1 and Subsection 4.2.4.2; being two independent parameters, multiplying both gives their joint effect in selecting start node. Once we have the start nodes, the possible destination (end) node selection probabilities are calculated as in Subsection 4.2.4.1. As we have the source and destination node pairs, their interaction probabilities are calculated considering whether they are in the same or different cluster as in Subsection 4.2.4.3. The elements of the Markovian matrix in Subsection 4.2.2.2 are then updated with this probability. The interaction takes place across the selected start and destination nodes with interaction probability above certain threshold limit (say, top 30%). The above procedure is iterated for each successive interval till the end of the training period, evaluated links over each interval being considered for training data for subsequent intervals and then for predicting the link occurrence over time slots $[T, T+1]$ of the test set, and also over time stamps in time slot $T + 1$. We use top 5%, 10%, 20% of the predicted link probabilities and compared it with actual occurrence of links to evaluate our prediction model.

Every created edge contributes to the degree of respective start and end nodes. Thus, the degree distribution of the application network depends on these created edges. The degree distribution of the real world large scale networks are known to exhibit power law. So, we need to investigate whether the correctly predicted edges by our prediction model, results in power law for degree distribution over the respective application network.

4.3. POWER LAW DEGREE DISTRIBUTION OF PREDICTED EDGES

Here, we show that our proposed approach (model) for node and edge selection, results in power law distribution of the resulting node degree over the network, as is observed in real world networks. In Subsection 4.2, we integrated temporal dimension in to our parameters.This helps in synchronization of the prediction model with the underlying network evolution. Moreover, in Subsection 4.2.4.2, we integrate age of node along with its degree, in its selection. Now, let us assume that the time interval between successive edge creation of the node, say v, be $\tau_d(v) = t_{d+1}(v) - t_d(v)$, where $(d + 1)$ and d are its node degree at time $t + 1$ and t. The closest replication model (distribution) of this time interval is given by the power law with exponential cut off [101], $P_l(\tau_d;\ \rho,\sigma) \propto \tau_d^{-\rho} exp(-\sigma\tau_d)$, where d is current degree, ρ,σ are parameters of distribution of τ_d as a function of current node degree d. In Subsection 4.2.4.2, we also integrate node degree that reflects the preferential attachment property exhibited by networks. Thus, the slope ρ of the power law distribution remains constant, whereas parameter σ representing exponential cutoff part becomes faster. Thus, successive edges are added at a faster rate, i.e., $(d+1)^{st}$ edge of a node added faster than d^{th} edge. So, distribution of τ_d becomes $P_l(\tau|d;\ \rho,\sigma) \propto \tau^{-\rho} exp(-\sigma d\tau)$. A degree d node v, samples with intervals τ from $P_l(\tau|d;\rho,\sigma) = \frac{1}{Z}\tau^{-\rho} exp(-\sigma d\tau)$ and remains inactive for a period τ. Additionally, the experimental analysis of nodes life time in Section 4.6 shows that they are exponentially distributed with parameter say, λ.

Theorem 4.3.1 The resulting node degree distribution exhibits power law.

Proof 4.3.2 *We evaluate the normalization constant Z from the distribution for time interval* $P_l(\tau|d;\rho,\sigma) = \frac{1}{Z}\tau^{-\rho} exp(-\sigma d\tau)$*. We have,* $Z = \int_0^\infty \tau^\rho e^{-\sigma d\tau} d\tau = \frac{\Gamma(1-\tau)}{(\sigma d)^{1-\tau}}$*. Let* T *be the life time of a node. Between two successive samples of a node from the distribution* $P_l(.)$*, it (node) remains inactive for* $\tau(k)$ *time period. Each successive sampling increases node degree by* 1*. Let* D *be the final degree a node attained in its life time* T*. In attaining*

D, the node remains inactive in total for:

$$\sum_{d=1}^{D} \tau(k) \leq T. \tag{4.19}$$

On a similar note, the expected time interval $E(\tau|d;\rho,\sigma)$ for a node of degree d is:

$$E(\tau|d;\rho,\sigma) = \frac{\Gamma(2-\rho)}{\Gamma(1-\rho)}(\sigma d)^{-1}. \tag{4.20}$$

From Equations (4.19) and (4.20), we relate lifetime T and expected final degree D of the node to have inequality:

$$\sum_{d=1}^{D} \frac{\Gamma(2-\rho)}{\Gamma(1-\rho)}(\sigma d)^{-1} = \frac{\Gamma(2-\rho)}{\Gamma(1-\rho)}(\sigma)^{-1} \sum_{d=1}^{D} d^{-1} \leq T. \tag{4.21}$$

We know that $\sum_{d=1}^{D} d^{-1} = \Theta(lnD)$. From Equation (4.21), the final degree D of a node with lifetime T is:

$$D \approx exp\left(\frac{\Gamma(1-\rho)}{\Gamma(2-\rho)}\sigma T\right) \tag{4.22}$$

Thus, D is exponential function of the lifetime (T) of a node. We can write it as, $D = exp(\varphi T)$, where $\varphi = \frac{\Gamma(1-\rho)}{\Gamma(2-\rho)}\sigma$.

Further, the node life time is exponentially distributed; thus, the distribution of node degree (D) using λ and φ will be:

$D \sim \frac{\lambda}{\varphi D} e^{\frac{-\lambda}{\varphi} logD} = \frac{\lambda D^{-(1+\frac{\lambda}{\varphi})}}{\varphi}.$

With exponent $(1 + \frac{\lambda}{\varphi})$, the node degree is power law distributed. Thus, predicted edges in our model results in power law degree distribution, as in case of real world networks.

In the following, we consider information theoretic prediction model in further reduction of uncertainty and thus, enhancing the prediction accuracy.

4.4. INFORMATION THEORY IN ELIMINATING PREDICTION UNCERTAINTY

Entropy is a metric for measuring the uncertainty (or, information content) over the probability distribution of a random variable [35]. Rather than the weighted average of the information content over all the outcomes, we are interested in the information content of any specific output called its self information. In order to explore the causality and association relation in the interactions over node pairs, we consider the concept of mutual information. First we formally define self information and mutual information [35], before using them in the context of link prediction.

Definition 4.4.1 *Self and Mutual Information [35]: Let x be the outcome of a random variable X with probability $p(x)$. The self-information or uncertainty ($I(x)$) of the outcome x is given by $I(x) = log\frac{1}{p(x)} = -logp(x)$. That is, the higher the self-information of x, the lower is its prediction outcome.*

Now in case of two random variables X, Y with joint and marginal probability mass functions respectively $p(x, y)$ and $p(x)$, $p(y)$, the mutual information is defined as $I(X; Y)$ $= \sum_{x \in X} \sum_{y \in Y} p(x, y) log\frac{p(x,y)}{p(x)p(y)} = \sum_{x,y} p(x, y) log\frac{p(x,y)}{p(x)p(y)} = \sum_{x,y} p(x, y) log\frac{p(x|y)}{p(x)}$. For outcomes $X = x$ and $Y = y$ we have $I(x; y) = log\frac{p(x,y)}{p(x)} = -logp(x) - (-logp(x|y)) =$ $I(x) - I(x|y)$. Thus, mutual information is the reduction in the uncertainty in one random variable due to another variable and in absence of causal relation mutual information will be 0.

In reference to future interactions, we reduce the uncertainty among pairs of structurally correlated nodes for future interactions. For this we use the triad structure of interaction patterns among nodes.

Definition 4.4.2 *Conditional Self-Information in Prediction Accuracy: Let us consider any node v_i and let $N^t_{v_i}$ be its set of neighbors interacting with it at time t. Let a pair of correlated nodes be (v_i, v_j) and their respective neighbors at time t be $N^t_{v_i}$ and $N^t_{v_j}$, which are represented as N^t_i and N^t_j for simplicity. Let the set of common neighbors be, $CN^t_{i,j}$ =*

$N_i^t \cap N_j^t$. *Now, we predict the possible future interaction between pair of nodes* (v_i, v_j) *as the conditional self-information content* $-I(L_{i,j}^{t+1}|CN_{i,j}^t)$. *From the definition of self-information content, the smaller is its value, the higher is prediction accuracy for the link* $L_{i,j}^{t+1}$ *at time instance* $(t+1)$.

Definition 4.4.3 *Capturing Correlation into Prediction: We know that for any two random variables* X *and* Y*, the mutual information can be represented as the reduction in uncertainty in one variable due to the information of the other variable, i.e.,* $I(X;Y) = I(X) - I(X|Y)$. *In order to incorporate the correlation between future interaction of a pair of nodes and their common neighbors in reducing prediction uncertainty, we use the above relation. Thus, we define* $I(L_{i,j}^{t+1}|CN_{i,j}^t) = I(L_{i,j}^{t+1} - I(L_{i,j}^{t+1}; CN_{i,j}^t))$*, where* $I(L_{i,j}^{t+1})$ *is the self information of the interaction between node pairs* (v_i, v_j) *at time* $(t+1)$. *The self-information is evaluated on the prediction probability of elements of Markovian matrix* (M_{t+1})*, i.e,* $p_{d_i d_j}^{C_i C_j}$*, as defined in Subsection 4.2.2. The mutual information* $I(L_{i,j}^{t+1}; CN_{i,j}^t)$ *effects reduction in uncertainty of the predicted link* $L_{i,j}^{t+1}$ *due to interactions with common neighbors.*

When nodes in $CN_{i,j}^t$ are independent of each other, mutual information is the aggregate over all nodes (ls) in $CN_{i,j}^t$ as given in Equation (4.23).

$$I(L_{i,j}^{t+1}; CN_{i,j}^t) = \sum_{\forall l \in CN_{i,j}^t} I(L_{i,j}^{t+1}; l) \tag{4.23}$$

Now we capture the correlation for all node pairs that have interactions with each node $l \in CN_{i,j}^t$ and their corresponding mutual information as in Equation (4.24), by computing the average mutual information over all such node pairs.

$$I_{avg}(L_{i,j}^{t+1}; l) = \frac{1}{|N_l^t|(|N_l^t| - 1)} \sum_{\substack{\forall i', j' \in N_l^t, \\ i' \neq j'}} I(L_{i',j'}^{t+1}; l) \tag{4.24}$$

From the definitions of mutual information and conditional self-information, and from all pairs of nodes that share a single common neighbor (say l) we will have:

$$\forall i, j \in N_l^t : I_{avg}(L_{i,j}^{t+1}; l) = I_{avg}(L_{i,j}^{t+1}) - I_{avg}(L_{i,j}^{t+1}|l) \tag{4.25}$$

Now, the conditional probability corresponding to the conditional self-information for interactions can be evaluated using the modularity index relative to node l as follows:

$$\forall i, j \in N_l^t : p_{avg}(L_{i,j}^{t+1}|l) = |CNN_l^t| / \left| \binom{N_l^t}{2} \right| \tag{4.26}$$

where $|CNN_l^t|$ is number of neighbor pairs of l with mutual interactions at time t, i.e. connected neighbor pairs of l, and $\left| \binom{N_l^t}{2} \right|$ is all possible pairs of neighbors of node l.

Lemma 4.4.1 *Average Mutual Information and Average Conditional Self-Information: Given the prediction probabilities* $p_{avg}(L_{i,j}^{t+1})$*, the average mutual information* $I_{avg}(L_{i,j}^{t+1}; l) = \frac{1}{|N_l^t|(|N_l^t|-1)} \sum_{\substack{\forall i',j' \in N_l^t \\ i' \neq j'}} \left(-logp(L_{i',j'}^{t+1}) + log \frac{|CNN_l^t|}{\left| \binom{N_l^t}{2} \right|} \right)$.

Proof 4.4.2 $I_{avg}(L_{i,j}^{t+1}; l) = \frac{1}{|N_l^t|(|N_l^t|-1)} \sum_{\substack{\forall i',j' \in N_l^t \\ i' \neq j'}} I(L_{i,j}^{t+1}) - I_{avg}(L_{i,j}^{t+1}|l)$

$= \frac{1}{|N_l^t|(|N_l^t|-1)} \sum_{\substack{\forall i',j' \in N_l^t \\ i' \neq j'}} (-logp(L_{i',j'}^{t+1}) - (-logp(L_{i',j'}^{t+1}|l)))$

$= \frac{1}{|N_l^t|(N_l^t|-1)} \sum_{\substack{\forall i',j' \in N_l^t \\ i' \neq j'}} (-logp(L_{i',j'}^{t+1})) + log|CNN_l^t| / \left| \binom{N_l^t}{2} \right|$.

Thus, we compute the likelihood score of the interaction between node pairs, in terms of their conditional self-information as follows.

$$I_{avg}(L_{i,j}^{t+1}|CN_{i,j}^{t}) = I(L_{i,j}^{t}) - \sum_{l \in CN_{i,j}^{t}} I_{avg}(L_{i,j}^{t+1};l) \tag{4.27}$$

We illustrate the steps discussed above, for uncertainty reduction over the computed prediction probabilities of the transition matrix in Subsection 4.2, through a simple example in Figure 4.4 and show its effectiveness.

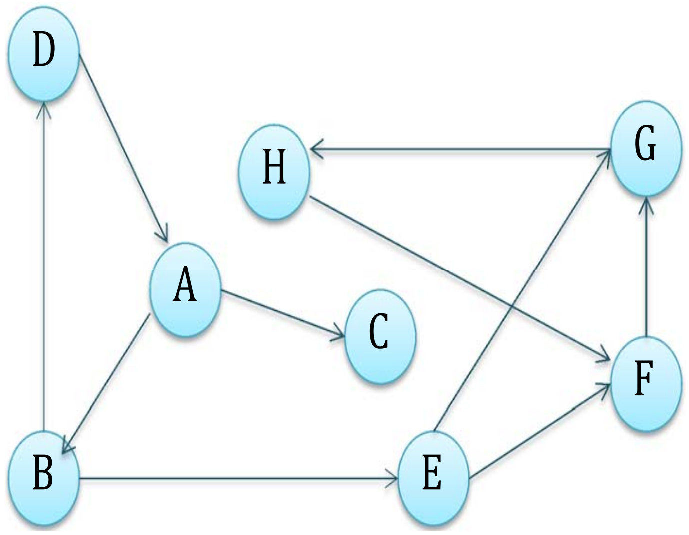


Figure 4.4. Example: Information Theory in Prediction

Example: In our simple example node degrees are $1,2,3$, with frequency $1,2,5$ respectively; thus, the respective degree distributions $(p(d))$'s are $\frac{1}{8}, \frac{2}{8}, \frac{5}{8}$. Let the two clusters be $C_1 = \{A,B,C,D\}, C_2 = \{E,F,G,H\}$. From definition in Subsection 4.2.2.2, we obtain p_{BC}, $p_{BD}, p_{CD}, p_{CE}, p_{CH}, p_{EG}, p_{GH}, p_{EH}$. For simplicity we are omitting the cluster and time instance notations in this illustration. Using the above procedure we find the prediction probability for interaction or link (B,C). We have the set of common neighbors of nodes B and C as, $CN_{BC} = \{A\}, N_A = \{B,C,D\}$, with only connected node pair (B,D) in N_A. Evaluating the conditional self-information content over the node pairs in N_A we get, $I(L_{BD}|A) = -0.477$, similarly $I(L_{BC}|A) = I(L_{CD}|A) = 0.477$. The self-information $I(L_{BD}) = -log20 + log1728$. Similarly, $I(L_{BC}) = -log\frac{4}{576}$. Now, $I_{avg}(L_{BC,BD,CD}; A) = 0.8651$. So, $I_{avg}(L_{BC}|A) = 3.0269$. Similarly, $I_{avg}(L_{CD}|A) = 1.2949$. Thus, interaction or link between node pairs C, D is more likely to occur.

Similarly, over prospective links E, H and B, C, the 1^{st} one is higher probable with $I_{avg}(L_{EH}|CN_{EH}) = 0.113$ and $I_{avg}(L_{BC}|CN_{BC}) = 3.0269$, from the fact that they have two common neighbors in comparison to node pairs B, C with only one common neighbor A.

Further, different nodes vary with the magnitude that they contribute to the mutual information and hence in reducing uncertainty in link prediction. For example, consider between nodes F and A. From the above computation we have $I_{avg}(L_{EG,GH,EH}; F) = 1.106$ and $I_{avg}(L_{BC,BD,CD}; A) = 0.8651$. So, in computing conditional self information that is the likelihood of occurrence of a link, the later one reduces uncertainty by lesser amount in comparison to the fast one. Thus, node F contributes to reducing uncertainty in predicting future interaction to a higher degree than node A. We can explain this from the network structure that the corresponding node is associated with. Though, both the nodes have degree 3, node F has higher number of triads or denser structure associated with, in comparison to node A. Thus, the proposed information theoretic formulation, incorporates the underlying network structure to help reduce prediction uncertainty.

4.5. ORDER OF PREDICTION MODEL

An important question in the accuracy of prediction model is how should we consider the dataset, so that the prediction outcome over the test set is neither under-fit nor over-fit? A simple way used in this is to do k-fold cross validation as we use in Subsection 4.6; that also makes the model more robust and generalized one. A significant underlying aspect addressed here is the amount of data considered as training and test set.

Besides this another important aspect is, how many previous states the Markov model needs to keep as history, i.e. the order of the Markov model. More so, to integrate temporal evolution of local and global structure we consider both fine and coarse-grained time scale. The links at successive stages which we model as states of the evolving network graph, in turn corresponds to the states of the Markov model. So, the question is how many states s, or how many states over corresponding time scales that the model needs to keep as history. Another way to put it is, how many states s will be sufficient to eliminate the uncertainty in the stochastic prediction model. We use the information theoretic and compression based concepts and results from [35] and [140] in giving a optimal bound (upper bound). From our definition of prediction model in Subsection 4.2.2, we can infer the following definition of states.

Definition 4.5.1 *The evolution history or state transition history of the links (and hence the Markov model) is given by state set $s = \{s_1, s_2, \ldots\}$. The associated state transition matrix $M(t)$ and $M(T)$ updates the subsequent state. In fact, these states (s_is') are not necessarily distinct.*

Furthermore, the learning of the stochastic process over the history of states is done offline (though, can also be online too), to get better insight of the possible future state. The essence of the prediction model is to incorporate the correlation information (repetitive nature) from history states. In fact, it is the repetitive nature of identifiable states of history that makes stationarity as an intrinsic property of this stochastic model. Thus, from [35] we can infer the following definition over stochastic process associated with states of the prediction model.

Definition 4.5.2 *The stochastic prediction model for links is a stationary process $\mathscr{P} = \{p_i\}$, where $p_i = s_i \in s$ when the i^{th} transition occurs. The joint distribution of any sub-sequence $p_i s$ is invariant relative to the shifts (say, t) in time axis. i.e., $Pr[p_1 = s_1, p_2 = s_2, \ldots p_z = s_y] = Pr[p_{1+t} = s_1, p_2 = s_{2+t}, \ldots p_{y+t} = s_y]$. The state transition history is a sample trajectory in $\mathscr{P}$.*

In the Markovian prediction model, the question that arises is whether higher order adds to the better prediction of the generalized prediction model that is to investigate whether it make the model richer. The intuition is that there has to be a bound to the richness, as the sought after general model has to be the richest. So, at what order (say, k) should we stop. From the definitions of entropy rate in [35], we have the entropy rate per state and the continual entropy rate as follows:

Definition 4.5.3 *The* entropy rate per state $H(\mathscr{P})$ *for a stochastic process $\mathscr{P} = \{p_i\}$, is defined by $H(\mathscr{P}) = \lim\limits_{y\to\infty} \frac{1}{y} H\{p_1, p_2, \ldots, p_y\}$, when the limit exists. Similarly, the* ***conditional entropy rate*** *for the same process is given by $H'(\mathscr{P}) = \lim\limits_{y\to\infty} H(p_y | p_1, p_2, \ldots, p_{y-1})$, when the limit exists.*

Further, the equality of $H(\mathscr{P})$ and $H'(\mathscr{P})$ also follows from [35] due to the following.

For any set of k discrete random variables with joint probability distribution given by $Pr(p_1, p_2, \ldots, p_k) = Pr[p_1 = s_1, p_2 = s_2, \ldots, p_k = s_k], \forall i, s_i \in s$, the joint entropy is given by $H(p_1, p_2, \ldots, p_k) = \sum\limits_{i=1}^{k} H(p_i | p_1, p_2, \ldots, p_{i-1})$. From the above conditional entropy, it implies that higher the history of states, better is the model or more information rich. In other words, lower order models underestimate the associated uncertainty in the prediction of the underlying model. In fact, from the results in [35] the highest order of the Markov model naturally follows from the following fact. *For a stochastic stationary process $P = \{p_i\}$, the conditional entropy $H(p_y | p_1, p_2, \ldots, p_{y-1})$ is a decreasing function in y and is limited by $H'(\mathscr{P})$.*

It is clear now that the marginal improvement in model richness starts to die out soon. Intuitively, the largest meaningful order has something to do with largest chain of dependency observed in the state history and is evident from the left hand side of the joint entropy of the k-states ($H(p_1, p_2, \ldots, p_k)$). Thus, the per state entropy rate would then represent the running average of conditional entropy rates. Thus, from the result of [35] we have: *For a stationary stochastic process $\mathscr{P}$, both the limits in definition 4.5.3 exist and are equal i.e. $H(\mathscr{P}) = H'(\mathscr{P})$.*

Now, the only thing we need to decide is the appropriate order for the universal/generalized model. This is achieved by a class of compression algorithms proposed by Ziv and Lempel [140]. The larger contexts in the conditional probabilities helps in higher prediction accuracy/richness of the higher order Markov model, as inferred earlier. As, there is a limit to the model richness for stationary processes, the symbol wise model by Ziv and Lempel should eventually converge to universal model. In fact, from the result in [55], the bound is deduced to be $O(log(y))$.

The incremental parsing model (IP) asymptotically outperforms a Markov predictor of any finite order, thus attaining finite state predictability. But the rate at which the predictability is attained is $O(\frac{1}{\sqrt{logy}})$, which is slower than the rate $O(\sqrt{2^k/y})$, for which the number of effective states that IP has is $O(y - logy)$ and thus, its equivalent Markov model order is given by $O(log\frac{y}{logy}) = O(logy - loglogy) \approx O(logy)$, where y is the (total) number of states.

4.6. EXPERIMENTAL RESULTS

We evaluate our approach using real world data sets and comparing the prediction accuracy over the standard receiver operating characteristic (ROC) curve with false positives in X- axis and true positives in Y-axis to represent the outcomes in successive time intervals. The area under the ROC curve (AUC measure) represents the achievable true positive rate (TPR) relative to false positive rate (FPR). The ROC curve helps us evaluate the effectiveness of prediction model from tradeoff between TPR and FPR. The higher the TPR than FPR, the better the model; though, in general TPR > FPR is considered a good model.

We evaluate and compare the accuracy of our Markov model based approach (Das) relative to the state-of-the-art approaches such as: Adamic-Adar score (AA), Katz-measure, Common Neighbors (CN) mentioned in [88] and the recent dynamic approaches such as Sarkar et al. [116] (Sarkar) and Tylenda et al. [130] (Tylenda). The AA, CN, Katz are static approaches based on the aggregated graph over $[0,T]$. TPR and FPR are computed from fraction of correct predictions and fraction of wrong predictions. The measure is equally applicable to evaluate prediction accuracy of selected start nodes. Further, the AUC values resemble ROC curve, in reflecting the performance/accuracy of the approach.

We use Twitter, Facebook, Enron and DBLP dataset. Use of Enron and DBLP data set help effective comparison with the work of Sarkar et al. [116] and Tylenda et al. [130], as they also use the same data sets. Further, Twitter network data set helps us evaluate the effectiveness of all the dynamic approaches. The Facebook dataset used by Viswanath et al. [132] contain wall-to-wall post relationship among $11,470$ users between years 2004-to-2009. The DBLP dataset is from [46], to which the year of collaboration timestamps are attached and 3000 authors with at least 5 papers are considered. With Twitter API, over $88,930$ time stamped interactions from reply, retweet, favorite message during a period of 32 weeks are collected from Twitter network (July, 2015-to-January, 2016). In the Enron data set [49], the email communication in the range of July 2000 to March 2002 is considered for our experiments.

Over Enron, Facebook and twitter, the data is time stamped in hour:min format. A day is taken as the smallest time unit i.e. time stamp over these networks, whereas a month and a week taken as time slots over them, respectively. For DBLP network, we consider a year as the time slot and time stamps are half yearly. The frequency of interaction between node pairs is incorporated over the respective time stamps.

We consider all time stamps in one slot to predict over next time stamp and all time slots to predict over next slot; thus, the number of history states s equals the number of time stamps or total number of time slots in respective time domain. For example, in twitter application network, we consider time stamps over three slots. Thus, $s = 21$ for time stamps, and $s = 31$ for time slots, used for prediction over next time stamp or time slot. Further, over Enron email application network we have $s = 20$ w.r.t. time slots, whereas, $s = 30$ over time stamps.

In Figure 4.5, the age of nodes over three networks follow exponential distribution, as the elongated shape along X and Y-axis for each is observed. This justifies our consideration of exponential parameter λ in Subsection 4.3. Further, the inter-arrival time between interaction is power law distributed (exponent between 2-to-3) with exponential cut off over all the three networks, as is evident from the Figure 4.6. Moreover, the parameters ρ associated with their distribution remains almost constant (Y-value almost same throughout per curve) with degree d, where as the other associated parameter σ grows linearly with time, as is reflected in Figure 4.7. We used these facts in Subsection 4.3. In Figure 4.5 and Figure 4.6, we presented separate graph for each network owing to distinct ranges.

First, we show the accuracy of our start node selection with top 30% of the computed probabilities over DBLP, Twitter and Enron datasets in Figure 4.8. To clarify further, when we use top 30% of the link prediction probabilities, we check the TPR and FPR values within this set of predicted links. Owing to the relatively lesser dynamic (time-varying) nature of the DBLP dataset, it has better accuracy than other two datasets.

Now, we evaluate the prediction accuracy over the next time slot, i.e. $[T, T + 1]$. The ROC curves in Figure 4.9 and the 2nd graph of Figure 4.8 presents the prediction accuracy of our approach (Das) relative to the other approaches over three datasets. The three dynamic approaches (Tylenda, Sarkar and Das) outperform the remaining static approaches, as is evident from the ROC curves (and AUC values Table 4.2). Further more, the probabilistic Markov model in (Das) has higher prediction accuracy than the integration of local network structure and time series based approaches (Tylenda and Sarkar). The better prediction accuracy of our model is attributed to the two way temporal analysis, that is consideration of two time scales and the corresponding local and global structural evolution. Here, we use top 20% of the predicted links in all cases, for evaluating accuracy of the proposed model. Each of the ROC curves shows the variation of rate of correct predictions to rate of wrong prediction.

In the 1st graph of Figure 4.10, we show the link prediction accuracy while selecting top 5%, 10%, 20% of the prediction probabilities for links over Twitter network. In the 2nd graph we show the prediction accuracy (with top 20% and 30% link selection) over time stamps of one slot ($[T, T + 1]$). Here, we have included the effect of cluster evolution and rate of evolution computed over time slots, to predict for the next time stamps (days in Twitter network). Thus, prediction accuracy (with 20%) for time stamps and slots are almost similar over Twitter network.

To avoid possible over fitting in the above predictions, instead of taking only $[0, T]$ as training set and $[T, T + 1]$ as test set, we rather consider k-fold cross validation. To compute the prediction accuracy of model, we iterate it k times, over our observation interval and take average of the prediction outcomes over all test intervals. For example: in Twitter application network, of the 32 week (slots) observation period, we take sets of 31 weeks as training set in each iteration and predict over the remaining slot in that iteration. Over 32 test sets, we take average of the prediction outcome to determine the final prediction accuracy of the model. Table 4.3 shows the resulting prediction accuracy of three dynamic models, over three distinct datasets, with higher accuracy of our proposed model (Das). In Figure 4.11 we show the corresponding ROC curves for Enron and Twitter networks. Here, we show only the dynamic models, as static models have further lower accuracy.

It is interesting to note that the ROC curves for Figure 4.9 and 2nd graph of Figure 4.8, seems to have a lesser TPR around FPR value of 0.1 (we do not give any smoothed version of ROC curves, so at some points we may see this), for all baseline approaches (CN,

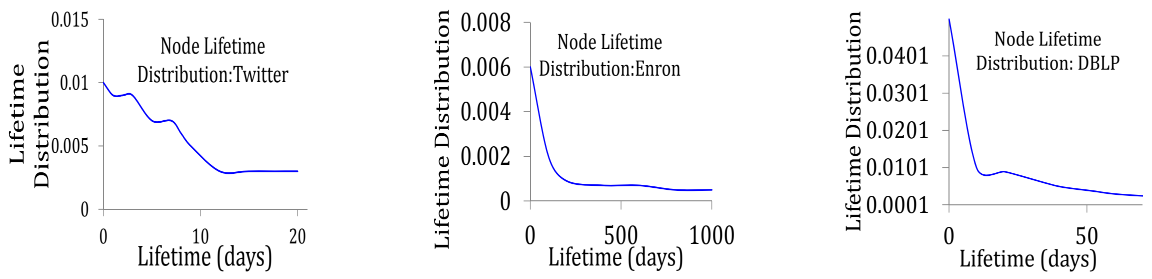


Figure 4.5. Distribution of Node Lifetime

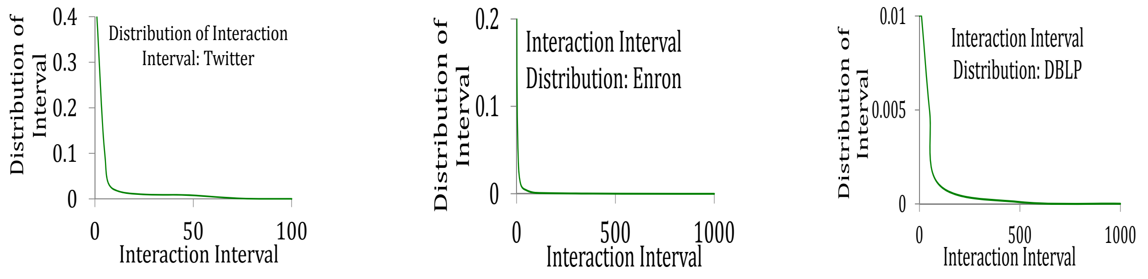


Figure 4.6. Inter-arrival Time Distribution: Interactions

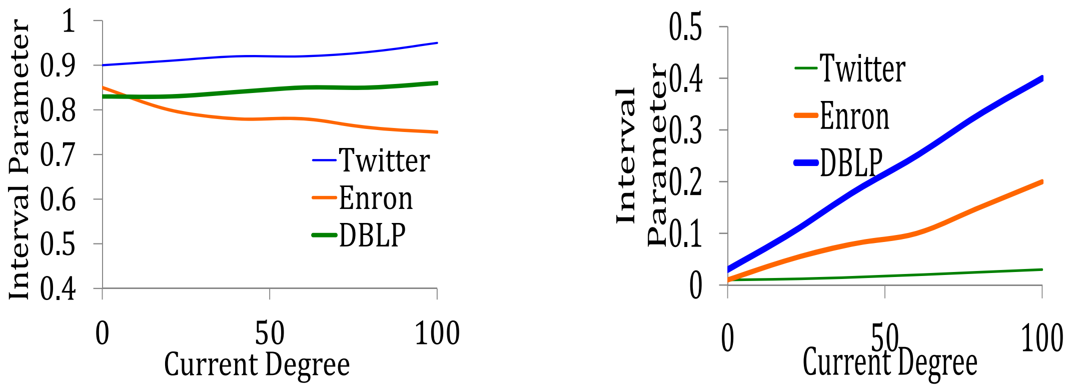
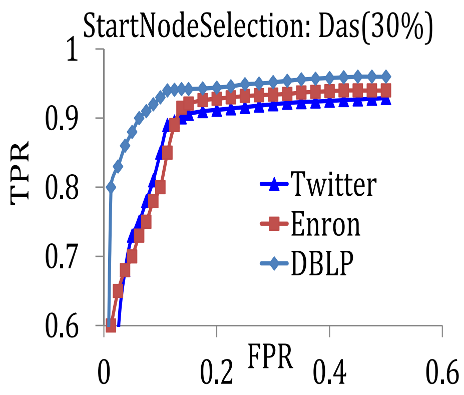


Figure 4.7. Evolution of ρ and σ with Node Degree

AA, and Katz); but, the curves have an overall upward trajectory. Further, the ROC curves in Figure 4.11 have relatively lesser prediction accuracy than the earlier curves, without over fitting being taken care of.

Over the subsequent time slots, we show variation of prediction accuracy in Figure 4.12 over 10 consecutive time slots and time stamps for Markov model, with top 20% and 30% selection of predicted links for Facebook network.



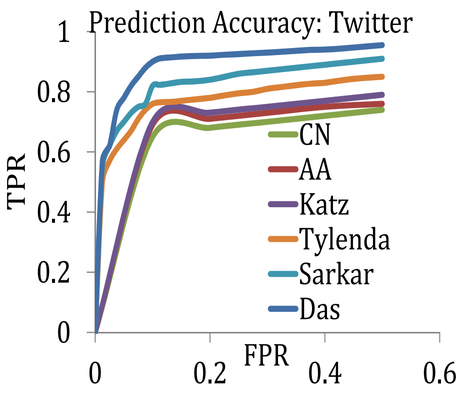
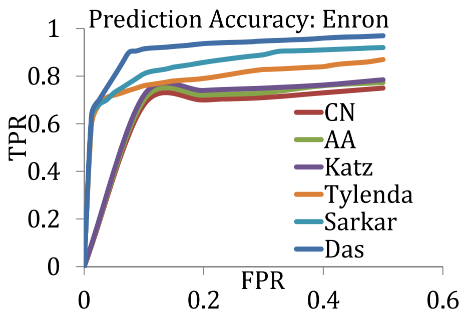


Figure 4.8. Accuracy over Node Selection and Link Prediction



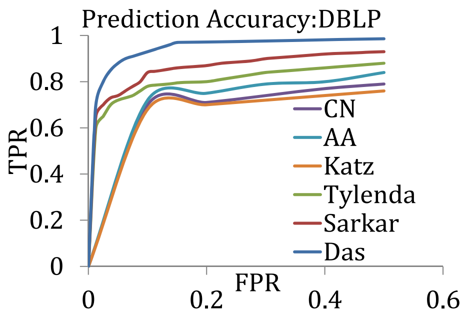
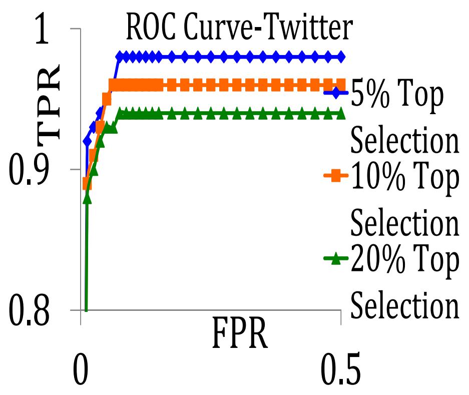


Figure 4.9. Accuracy over Link Prediction

Table 4.2. AUC Value with Our Model

AUC results over Twitter, Enron, DBLP datasets						
Dataset	CN	AA	Katz	Tylenda	Sarkar	Das
Twitter	0.6467	0.6816	0.6968	0.7643	0.8293	0.9183
Enron	0.6615	0.6753	0.7389	0.7835	0.9031	0.9328
DBLP	0.7523	0.7721	0.7338	0.8230	0.9143	0.9613

We now investigate the effect of number of history states. We show the results for one of the heavily dynamic network data set, Facebook for brevity and sufficiency. Considering the division of time stamps and slots as described above, we have 2190 time slots and 52560 time stamps, each as states over respective time domain, for Facebook network data set.



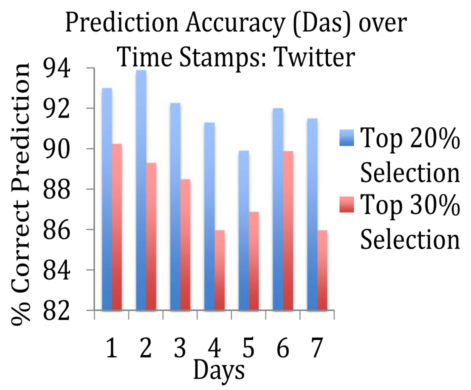
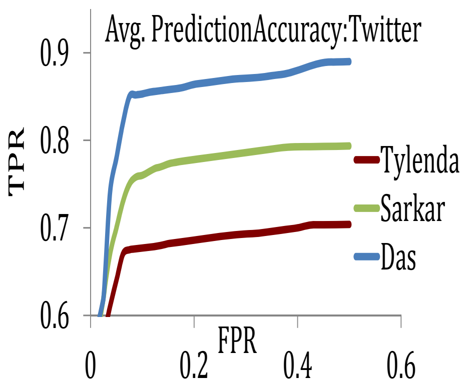


Figure 4.10. Link Prediction Accuracy: Time Slot and Time Stamp

Table 4.3. Average AUC Value with Our Model

AUC results over Twitter, Enron, DBLP datasets			
Dataset	Tylenda	Sarkar	Das
Twitter	0.7039	0.7935	0.8503
Enron	0.7313	0.8189	0.8635
DBLP	0.7953	0.8456	0.9093



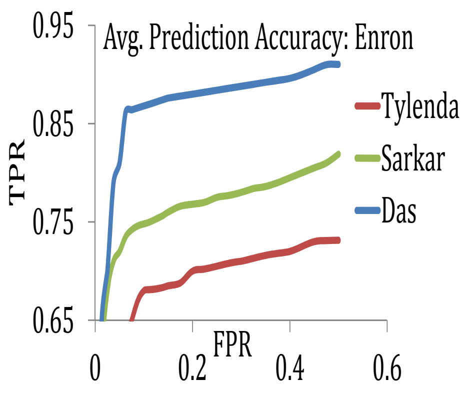
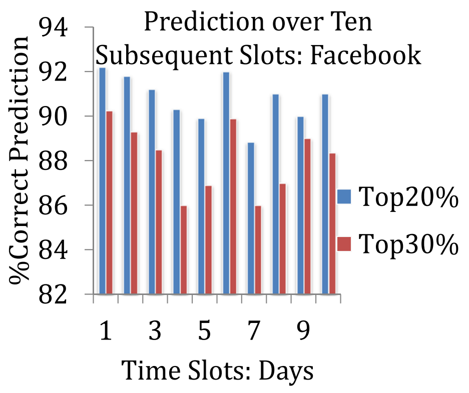


Figure 4.11. Average Link Prediction Accuracy over All Slots

We take s = 50 as history states and show the outcome averaged over all subsequent slots, over both Markov and ITH model. We also show the prediction accuracy for s = 10 and s = 16. We show the prediction accuracy for s = 10in the 1st graph of Figure 4.13.



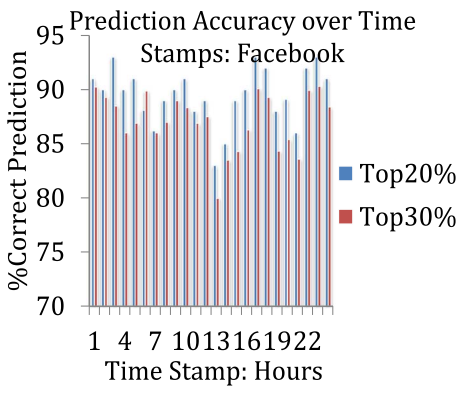
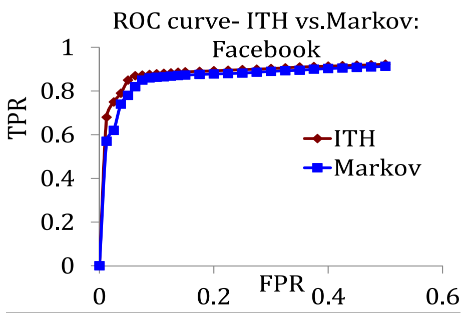


Figure 4.12. Prediction Accuracy: Time Slots and Stamps

Table 4.4. Average AUC Value: Markov and ITH Model

AUC results over Twitter, Facebook, DBLP datasets		
Dataset	Das	ITH
Twitter	0.8503	0.8493
Enron	0.8635	0.870
Facebook	0.8563	0.8589
DBLP	0.9093	0.9188



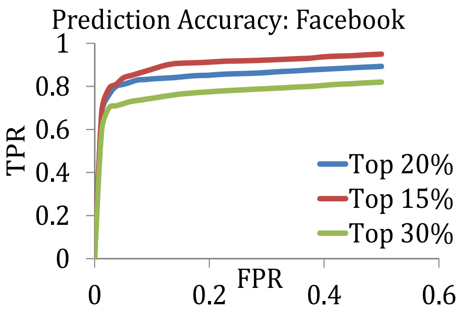


Figure 4.13. Prediction: ITH vs Markov, And Varying Degree of Selection

We show the corresponding AUC value for the comparison of ITH and Markov model in Table 4.4. Here the accuracy of ITH model is slightly better than Markov model. But for $s = 16$, we find the prediction accuracy for Markov model to converge to the accuracy of ITH model, possibly as the number of states are almost $O(logs)$, s is the total number of possible states. We also show the prediction accuracy of Markov model over Facebook network data set with top 15%, 20%, 30% selection of predicted links in the 2nd graph.

4.7. SUMMARY

To sum up, the effectiveness of our Markov model approach lies in the incorporation of the temporal evolution over both fine and coarse grained time domain, along with their correlated structural evolution (i.e. links and clusters respectively), rate of evolution and evolution dynamics in building our prediction model bottom up. This captures the randomness and multiple evolution profiles resulting in higher prediction accuracy. Further, our node selection in Markov model reflects higher prediction accuracy. Further, our information theoretic approach not only uses Markovian transition matrix as its basis, but also integrates network structural properties to further reduce underlying entropy. Both our approaches outperform the state-of-the-art static approaches, as well as the recent dynamic approaches, as is evident from the ROC curves (also AUC value) over three highly dynamic and one relatively static example network. We observe that our Markov model is competent enough with the proposed information theoretic model, as both have almost same prediction accuracy; more so, when we approach the upper bound of $O(logs)$, on total number of history states the prediction accuracy of Markov model converges with the information theoretic model, thus justifying that we do not need to keep track of all the history states for best prediction accuracy, irrespective of how large the network data sets are. More so, we derived the theoretical upper bound on number of desired states of the Markov prediction model for best prediction accuracy. The computation cost of the prediction model is bounded by $O(\Delta t logs)$, where Δt is the computation time per state and $logs$ is number of history states taken into consideration.

5. CONTACT PATTERNS IN FUTURE CONTACT PREDICTION OVER MOBILE NETWORKS

As discussed earlier, with advances in the Internet and mobile technology, and decreasing cost of mobile devices, large scale dynamic pervasive networks are now ubiquitous in solving many earlier service limitations. The challenge here is in its underlying temporal graph. It introduces technical limitations in efficient routing, maximal coverage with minimal latency, data offloading, to effective dissemination over mobile networks or mobility induced dynamic networks. The efficient solution to these inter-related problems lies in the novel prediction strategies for most accurate future contacts; more so, inter-contact period between node pairs, and their future contact time are also of significant use. In contrast to the existing strategies that consider regular pattern and periodic nature of contacts, we propose a novel stochastic Poisson process model that employ multi-recurrent, dependent pattern of contact as the basis of our novel prediction model. We use variants of cascaded non-homogeneous Poisson process model as our proposed model. We predict number of contacts relative to a node and over all nodes in any future interval, future contact time over a user and a pair of users. We discuss the process details in the following sections. Finally, we validate our model with a widely used empirical data set, and compare our model with doubly recurrent and homogeneous Poisson process model to conclude the superiority of our prediction model. We discuss the details in the following Subsections.

The following discussion is organized as follows. In Subsection 5.1, we present preliminary concepts and definitions used in contact prediction strategy, along with the formal problem definition. Subsection 5.2 describes our proposed variant of NHPP model. It includes framework to capture both multi-recurrent contact pattern and also random contact pattern, while Subsection 5.4 reports experimental results and finally a summary of our proposed approach and the results are offered in Subsection 5.5.

5.1. PRELIMINARY DEFINITIONS AND CONCEPTS: CONTACT PREDICTION

As earlier, we represent the underlying temporal network graph over an observation period as $G(V,E,[0,T])$, where $V\ and\ E$ represent the set of vertices and edges respectively, and $[0,T]$ is the observation period. The subgraph at any time period δt is represented as $G_{\delta t}(V_{\delta t}, E_{\delta t})$, where $V_{\delta t}$, $E_{\delta t}$ are the corresponding vertices and edges. Corresponding to real world application, where a contact is established between a pair of mobile devices/nodes within communication range, an edge is established in the underlying graph between a pair of nodes, i.e. $(v_i, v_j) \in E$ at time instance t called future contact time (contact initiation time), where v_i, v_j are the respective mobile nodes in the application network.

As mentioned earlier, we are leveraging the generalized recurrent pattern. It is not limited to only two degrees of periodicity (i.e. short term and long term or, daily and weekly periodicity), rather it takes into account the multi-recurrent contact pattern. Further, both direct and indirect dependent nature of the contact patterns over temporal dimension are considered to predict future contacts. In this context we formally define our problems as follows:

Let the contact remain active in time $[t, t+\delta t)$, then δt is called the contact period/duration for the corresponding node pair, $t \geq 0, \delta t > 0$. The number of contacts that a node v_i has in time interval δt i.e. $\forall v_j : (v_i, v_j) \in E_{\delta t}$ are called its neighbors ($\sum_{v_j}(v_i)$). For any node v_i, we predict its number of contacts during time period $[t, t+\delta t)$ given by $N[t, t+\delta t) = N(t+\delta t) - N(t)$.

We also define inter-contact period between a pair of nodes (v_i, v_j) having contact at time t and the subsequent contact at time $t+t'$, as the time period between its two successive contacts, where $t' > 0$. For any pair of nodes v_i, v_j we evaluate its aggregate inter-contact period as the ratio of the total time units of inter-contact period of the node pair to the total number of their contacts. Let k be the number of times the node pair v_i, v_j are in contact. Let

t'_{ij} be the inter-contact period corresponding to pair of subsequent contacts say, i^{th} *and* j^{th} contact for $i, j \in k$. Thus, the aggregate inter-contact period of the corresponding node pair is given by $\frac{\sum_{\forall i,j \in k: j=i+1} t'_{ij}}{k}$.

We describe our proposed model framework in detail below.

5.2. PROCESS MODEL

Let us define the number of observed contact counts at any time $t \in \{0, T\}$ in a temporal counting process as $N(t)$, that incorporates multi-recurrent contacts and their underlying dependency.

Let $[t_0, t_{n+1})$ denote the observation interval of the experiment and $0 < t_0 < t_1 < \ldots < t_{n+1}$ be the successive contact times of the same pair of nodes, say nodes v_i *and* $v_j, v_i \neq v_j$. Let the set of contact events be represented as $0, 1, \ldots, n$. Over the process model $N(t)$, let the latent variable z_i to indicate the dependence among recurrent contact observations, with z_n over observation interval $[t_n, t_{n+1}]$. Let the set of observed and hidden variables be $\{o_0, o_1, \ldots, o_n, o_{n+1}\}$ and $\{z_1, z_2, \ldots, z_n, z_{n+1}\}$. Further, with hidden state z_n, the n^{th} inter-contact time period is $t_{n+1} - t_n$. We propose to use the intensity function $(\lambda(t))$ underlying a $NHPP$ to predict the occurrence of interactions. As we study the interactions over temporal domain, it is natural to visualize the interaction set as time stamp of their occurrence along the time axis.

The overall graphical process model is shown in Figure 5.1. The counting process is the result of recurrent process $(N_r(t))$, and dependency among underlying recurrent events (shown with process hidden states Z_is').

In the following Subsections (5.2.1, 5.2.3), we consider process models for recurrent, dependent contacts and stochastic event dependent contacts. In the first subsection we consider contacts with multiple recurrent patterns with correlation among recurrent events.

5.2.1. Process Model With Recurrent Pattern. To incorporate multiple recurrent nature of contacts, we consider the intensity function as a sinusoidal function.

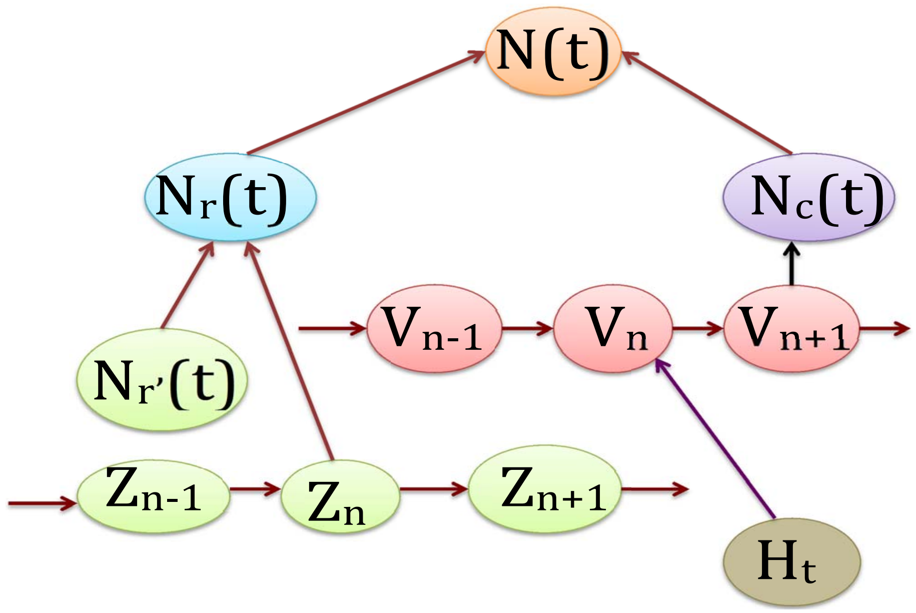


Figure 5.1. Stochastic Process Hierarchy

In contrast to the homogeneous Poisson process (HPP) model that has stationary intensity function independent of time, i.e. $\lambda(t) = \lambda$, we consider the standard non-homogeneous Poisson process model ($NHPP$) $\{N_{r'}(t) : t \geq 0\}$ as our basis with intensity function given by:

$$\lambda_{r'}(t) = \sum_{c=1}^{p} A_c sin(\omega_c t + \varphi_c), \tag{5.1}$$

where the sinusoid function represents recurrent contact pattern, with amplitude A_c, angular frequency ω_c and phase φ_c corresponding to cycle (period) c.

Let us consider that n contacts are observed at time instances $t_1 < t_2 < \ldots < t_n$ in time interval $[0,T)$. Consider the $NHPP$ stochastic process as basis with intensity function as described in Equation (5.1). The corresponding parameter set is given by $[A_1, A_2, \ldots, A_p, \omega_1, \omega_2, \ldots, \omega_p, \varphi_1, \varphi_2, \ldots, \varphi_p, c]$. Let us denote the above parameter set as $\theta_{r'}$. In determining the maximum likelihood estimates of the parameters, we determine the log-likelihood function of the parameter set with $N_{r'}(T) = n$ and $t' = (t_1, t_2, \ldots t_n)$, as log is monotonic. Thus, the log-likelihood of $\theta_{r'}$ is given by Equation (5.2).

$$\begin{aligned} \mathscr{L}_{r'}(\theta_{r'}|n,t) = & \sum_{c=1}^{p}\sum_{j=1}^{n} A_c sin(\omega_c t_j + \varphi_c) - \\ & \int_0^T exp\{\lambda_{r'}(z)\}dz \end{aligned} \tag{5.2}$$

Now the question is, how we determine the initial estimates for parameters of the above intensity function. We observe that if we get initial estimates of frequencies $\{\omega_1, \omega_2, \ldots, \omega_p\}$, then the initial estimates of rest of the parameters can be easily derived. In lieu of prior information of the process, the initial estimate of the frequencies are obtained from sampled data, with Fourier transform. Once we have the estimated values of frequencies, we estimate for amplitudes $\{A_1, \ldots, A_p\}$ and phases $\{\varphi_1, \ldots, \varphi_p\}$ are determined as follows. The log-likelihood function of Equation (5.2) is given as Equation (5.3).

$$\begin{aligned} \mathscr{L}_{r'}(\theta_{r'}|n,t) = & \sum_{c=1}^{p}\sum_{j=1}^{n} A_c sin(\omega_c t_j + \varphi_c) - \\ & \int_0^T \{\Pi_{c=1}^{p} exp\ (A_c sin(\omega_c z + \varphi_c))\}dz \end{aligned} \tag{5.3}$$

We assume $\omega_c = \frac{2\pi}{\mathscr{C}}$, and $\mathscr{C}$ as the length of one cycle. To simplify Equation (5.3), we consider values of ω_c, for which the observation interval constitutes complete cycles. Let k denote the number of cycles of length $\mathscr{C}$ in $[0,T)$, thus $T = \mathscr{C}k$. Now, the integral in the right hand side of Equation (5.3) can be represented as follows:

$$\begin{aligned} & \int_0^T \{\Pi_{c=1}^{p} exp(A_c sin(\omega_c z + \varphi_c))\}dz = \\ & \Pi_{c=1}^{p} \int_0^T exp\{A_c sin(\omega_c z + \varphi_c)\}dz = \\ & \Pi_{c=1}^{p} k \int_0^{\mathscr{C}} exp\{A_c sin(\omega_c z + \varphi_c)\}dz. \end{aligned} \tag{5.4}$$

We have, $sin(\omega_c t_j + \varphi_c) = sin\omega_c t_j \; cos\varphi_c + cos\omega_c t_j \; sin\varphi_c$. Further, let $\zeta = \omega_c z = \frac{2\pi z}{\mathscr{C}}$. Substituting it in the above log-likelihood function of Equation (5.3) and (5.4), we get:

$$\begin{aligned} \mathscr{L}_{r'}(\theta_{r'}|n,t) &= \sum_{c=1}^{p} A_c B(\omega_c) \; cos\varphi_c + \sum_{c=1}^{p} A_c B'(\omega_c) \; sin\varphi_c - \\ &\Pi_{c=1}^{p} k \int_0^{2\pi} \frac{\mathscr{C}}{2\pi} exp\{A_c sin(\zeta + \varphi_c)\} d\zeta \\ &= \sum_{c=1}^{p} A_c B(\omega_c) \; cos\varphi_c + \sum_{c=1}^{p} A_c B'(\omega_c) \; sin\varphi_c - \\ &\Pi_{c=1}^{p} \frac{T}{2\pi} \int_0^{2\pi} exp\{A_c cos\zeta\} d\zeta, \end{aligned} \tag{5.5}$$

where for $c = 1, \ldots, p$, $B(\omega_c) = \sum_{j=1}^{n} sin(\omega_c t_j)$ and $B'(\omega_c) = \sum_{j=1}^{n} cos(\omega_c t_j)$.

Now, from Equation (5.5), $\Pi_{c=1}^{p} \frac{T}{2\pi} \int_0^{2\pi} exp\{A_c cos\zeta\} d\zeta = \Pi_{c=1}^{p} T I_0(A_c)$, where $I_0(A_c)$ is the Bessel function of first order.

In order to estimate the parameters we take partial derivatives of the above equation with respect to φ_c and A_c, and equate to 0. Thus, we have:

$\frac{\partial \mathscr{L}_{r'}(\theta_{r'}|n,t)}{\partial \varphi_c} = A_c \; cos\varphi_c \; B'(\omega_c) - A_c \; sin\varphi_c \; B(\omega_c) = 0.$

$\frac{\partial \mathscr{L}_{r'}(\theta_{r'}|n,t)}{\partial A i_c} = sin\varphi_c \; B'(\omega_c) + cos\varphi_c \; B(\omega_c) - \Pi_{c=1}^{p} T I_0(A_c) = 0.$

From [84] we can deduce the parameters to be:

$\forall c \in \{1, \ldots, p\}$,

$\varphi_c = tan^{-1}[B'(\omega_c)/B(\omega_c)]$ and

$A_c = \frac{\sqrt{(B^2(\omega_c) + B'^2(\omega_c))}}{n_c}$,

where n_c is the number of contacts in the corresponding cycle c over interval $(0, \lfloor \frac{\omega_c T}{2\pi} \rfloor \frac{2\pi}{\omega_c}]$. Further, we see that the unknown continuous parameters corresponding to a cycle depend only on the frequency of that cycle that is already known.

Now that we have the needed parameters for the rate function of the recurrent $NHPP$, we will formulate the total number of contacts relative to a node or node pairs, next contact time, inter contact period and contact duration of node pairs.

Now, the probability density function of the contacts at the i^{th} interval is given by:

$$f(t_i|t_{i-1}) = \lambda_{r'}(t_i).exp(-\int_{t_{i-1}}^{t_i} \lambda_{r'}(z)dz). \tag{5.6}$$

Similarly, the cumulative distribution function for the future contact time say, τ_{i+1} conditioned on the observed value $\tau_i = t_i$ of the current contact time is given by:

$$F_{\tau_{i+1}|\tau_i}(t|t_i) = \begin{cases} 1 - exp[-\int_{t_i}^{t} \lambda_{r'}(z)dz] & t \geq t_i \\ 0 & \text{otherwise} \end{cases}$$

With recurrent process model as our basis, we now integrate with it the relationship between states of unobserved variables (i.e. dependent recurrent events). We present the enhanced model in the following subsection.

5.2.2. Multi-Recurrent Dependent Contact Pattern Model. We thus propose the resulting enhanced process model on our basic recurrent $NHPP$ model to be a cascaded non-homogeneous Poisson process model, $N_r(t)$.

We represent the cascaded non-homogeneous Poisson process model in Figure 5.2. Here, the time axis represents the sequence of observed time intervals. Further, it also has the set of observed variables as $\{o_0, o_1 \ldots, o_n, o_{n+1}\}$ and the corresponding set of hidden variables as $\{z_1, z_2, \ldots, z_n, z_{n+1}\}$, as discussed in Section 5.2.

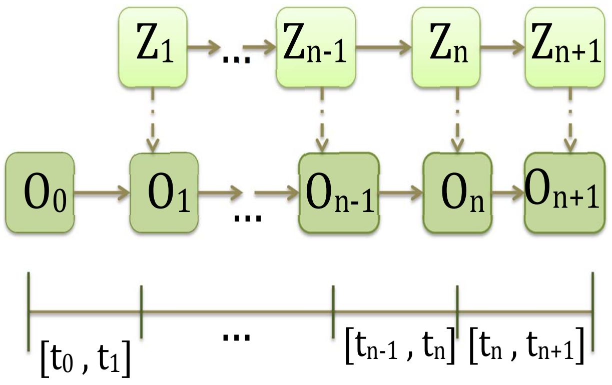


Figure 5.2. Cascaded NHPP

In this model, we have two hidden states reflecting either occurrence of a dependent recurrent contact or absence of it between successive time intervals. We represent them as s_1 or s_0 respectively. We define the corresponding state transition matrix as: $\mathscr{S} = \begin{pmatrix} s_1 & 1-s_0 \\ 1-s_1 & s_0 \end{pmatrix}$, where s_{ij} are transition probability between states i to j. The switch over the hidden state occur only at the time of the contact, when observed new state is guaranteed. Further, within each specific cyclic process say, weekly or semester wise pattern, the corresponding intensity function is uniform, but it is between the cyclic patterns and among distinct cycles that the intensity function varies. The transition function implies whether it is continuation of the earlier state (intensity function) or a different one. In other words, the hidden variable z_i represents whether the i^{th} event is a continuation of the current cyclic pattern ($z_i = 1$) or a new one ($z_i = 0$).

Thus, in the cascaded NHPP model (refined recurrent Poisson process model), $\{N_r(t) : t \geq 0\}$, where for any time $t_i \leq t$, $N_r(t_i)$ represents the number of contacts in time interval $[0, t_i)$, $t_i \leq t$. Here, this can be easily classified to contacts relative to each node or pair of nodes as needed. Similarly, the i^{th} contact time depends on both the state of hidden variable z_i and the previous contact time t_{i-1}. Since, observation at t_i depends on t_{i-1}, Figure 5.2 shows an initial output value O_0 with no corresponding hidden state. Let probability distribution of the first hidden state be $\{p_0, p_1\}$, where it refers to initial distribution corresponding to states s_0, s_1. Now, with hidden state s_i, the observations are created with probability as follows, where λ_r^0 refers to the contact rate at initial state and $1 \leq s_i \leq n$.

$$\eta = P(t_i | t_{i-1}, s_i) = \begin{cases} 1 - exp[-\int_{t_{i-1}}^{t_i} \lambda_r^0(z)dz] & s_i = s_0, \\ 1 - exp[-\lambda_r(t_i - t_{i-1})] & s_i = s_1, \end{cases}$$

We define contact rate at initial state λ_r^0, as the average contact rate, i.e. the ratio of total number of contacts relative to the corresponding time interval. We evaluate the intensity function (λ_r) and estimate the parameters in Subsection 5.3.

Now, we redefine the probability density function of contacts in the i^{th} interval from Equation (5.6) as:

$$f(t_i|t_{i-1}) = \lambda_r(t_i).exp(-\int_{t_{i-1}}^{t_i} \lambda_r(z)dz)). \tag{5.7}$$

Similarly, the cumulative distribution of future contact time (τ_{i+1}) is redefined as:

$$F_{\tau_{i+1}|\tau_i}(t|t_i) = \begin{cases} 1 - exp[-\int_{t_i}^{t} \lambda_r(z)dz] & t \geq t_i \\ 0 & \text{otherwise} \end{cases}$$

Now, the cumulative distribution function of inter-contact period ($\delta\tau_i$) conditional on $t_1, t_2, \ldots, t_i$ is given by:

$$F_{t_i}(\tau) = 1 - exp(-\lambda_r(t_i + \tau) + \lambda_r(t_i)) \tag{5.8}$$

In the following subsection we present the stochastic process model to capture contacts occurring due to sparse, random events and indirect correlation among such events.

5.2.3. Process Model With Random Patterns. We propose a Markov modulated Poisson process model ($MMPP$) in order to capture the random, rare event specific contact patterns and the underlying event specific indirect dependency that cause abrupt spike and/or damp in contacts. In general, $MMPP$ models are suitable when there is a randomly changing component effecting observations, which are well captured by these models. The advantage of using $MMPP$ models to capture the random contact observations here, lies in its ability to provide explicit closed-form prediction formulae for both events and the corresponding observations over any future time horizon that is over short term and long term time frames.

Let us consider the corresponding stochastic process $\{N_c(t) : t \geq 0\}$ with intensity function given by:

$$\lambda_c(t) = \lambda_c(H_t), \tag{5.9}$$

where H_t is the event process with random events that are possibly indirectly correlated resulting in hidden states V_is . The counting process $N_c(t)$ represents the number of contacts in time $(0,t]$, caused due to random events. Let us consider H_t over $\{1,...,n\}$ with discrete values $v \in H_t$. We represent the intensity function corresponding to a value (event) from the set as $\lambda_c(v) = \lambda_{c_v}, v \in \{1,...,n\}$.

Let us assume the underlying indirect correlated event process H_t follows a continuous time homogeneous Markov process with generator $\mathcal{G} = (g_{vu})_{n\times n}$. Let the initial distribution be $\beta_v = P(H_0 = v)$. The transitions in the MMPP model is shown in Figure 5.3. The discrete event states/values are represented as $\{1,2,...,n\}$, the transition probabilities among the states are shown as $g_{vu}, u, v \in \{1,...,n\}$ and the intensity function of contacts associated with respective states/events is given by $\lambda_i, i \in \{1,...,n\}$, as shown in the figure.

Let the vector of possible (initial) intensities over the discrete events/values be represented as $\Lambda_c^{init}(v) = (\lambda_{c_1}, \lambda_{c_2}, ..., \lambda_{c_n})^{init}$. The initial probability distribution over discrete parameter values/events is given by $\beta_v^{init} = \{\beta_{v_1}, \beta_{v_2}, ..., \beta_{v_n}\}^{init}$. Thus, the unknown parameters of the proposed model that needs to be determined are given by Equation (5.10, 5.11, 5.12):

$$\Lambda_c = (\lambda_{c_1}, \lambda_{c_2}, ..., \lambda_{c_n}) \tag{5.10}$$

$$\mathcal{G} = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1k} \\ g_{21} & g_{22} & \cdots & g_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ g_{k1} & \vdots & \cdots & g_{kk} \end{pmatrix} \tag{5.11}$$

$$\beta_v = (\beta_{v_1}, \beta_{v_2}, ..., \beta_{v_n}) \tag{5.12}$$

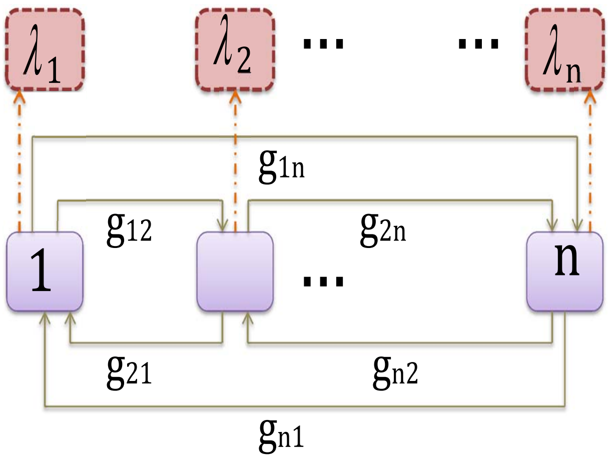


Figure 5.3. MMPP for Event Specific Indirect Correlation

Now, our objective is to determine the maximum likelihood estimates for the unknown parameters in the above equations for $\Lambda_c, \mathscr{G}, \beta_v$.

We consider the stochastic contact process and the underlying indirectly correlated event process being continuously observed over n state transitions, say over fixed interval $[0, t_n)$. Here, t_n is the time for n^{th} transition of event process H_t corresponding to indirectly correlated events that cause the corresponding contacts/observations. During the interval $[0, t_n)$, we represent the indirect variables as $v = \{(v_0, t_0), (v_1, t_1), \dots, (v_{n-1}, t_{n-1}), v_n\}$, where v_n corresponds to the n^{th} state of the indirect correlated process effecting the contact outcome. Let t_{s_i} is the time that the process remains in a state say, v_i, so $t_n = t_{s_0} + t_{s_1} + \dots + t_{s_{n-1}}$. Let the time instances for the observed contacts be denoted as $t' = (t'_1, t'_2, \dots, t'_k)$, where $0 \leq t'_1 \leq \dots \leq t'_k \leq t_n$ in time interval $[0, t_n)$.

The likelihood function can now be written as in the following Equation (5.13).

$$\mathscr{L}_c(\Lambda_c, \mathscr{G}, \beta_v | t', v) = \Pi_{j=1}^{k} \lambda_{c_{e(t'_j)}} . exp\left(- \sum_{i=0}^{n-1} t_i \lambda_{c_{v_i}} \right) \\ . \beta_{v_0} \Pi_{v \neq u} g_{vu}^{n_{vu}} \Pi_{e \in v} e^{g_{ee} \tau_e}, \tag{5.13}$$

where $e(t'_j) \in v$ is the observed state of the underlying event, n_{vu} is the transition frequency in the sample $v_0, v_1, \ldots, v_n$ and τ_e is the total observed time during which the indirect correlated event process occupied state $e(t'_j)$ during interval $[0, t_n)$.

The log-likelihood function corresponding to Equation (5.13) is presented in Equation (5.14).

$$\begin{aligned} log\mathscr{L}_c(\Lambda_c, \mathscr{G}, \beta_c | t', v) = & \sum_{j=1}^{k} \lambda_{c_{e(t'_j)}} - \sum_{i=0}^{n-1} t_i \lambda_{c_{v_i}} \\ & + log\beta_{v_0} + \sum_{v \neq u} g_{vu}^{n_{vu}} + \sum_{e \in v} g_{ee}\tau_e, \end{aligned} \tag{5.14}$$

Since log function is monotonic, the maxima for Equation (5.14) remains unaffected for Equation (5.13), and the earlier one is much easier to work with. We thus determine the maximum likelihood estimates for the parameters in $\Lambda_c, \mathscr{G}, \beta_v$ from Equation (5.14).

With univariate optimization technique the maximum likelihood estimates of parameters is as follows:

$$\begin{aligned} \lambda_{c_v} &= \frac{N_v(t')}{\tau_v}, \quad \forall v \in H_t \\ g_{vu} &= \frac{n_{vu}}{\tau_v} \quad \forall v \neq u \in H_t \\ \beta_v &= 0 \quad \forall v \in v \setminus \{v_0\}, \end{aligned} \tag{5.15}$$

where $N_v(t') = |\{k : e(t'_j) = v\}|$ is the number of times $e(t'_j) = v$.

Once we have the maximum likelihood estimates from Equation (5.15), we proceed to evaluate the prediction of the expected frequency of contacts over any future time interval $t > 0$. We observe the stochastic contact process $(N_c(t))$ and the event process (H_t) guiding indirect event correlation over time interval $[0, T)$.

Now, given the number of contacts and the states of the hidden event correlation process till time t, we evaluate the expected number of contacts that will take place over the Δt future time interval as follows:

$$O_t(\Delta t) = E(N_{t+\Delta t} - N_t | N_{t'}, X_{t'}; 0 \leq t' \leq t). \tag{5.16}$$

Above contact prediction can be computed explicitly using the following equation:

$$O_t(\Delta t) = \sum_{v_i \in v} \lambda_c(v) \int_0^{\Delta t} p_{v_i v_j}(t')dt' \tag{5.17}$$

where $p_{v_i v_j}(t') = P(X_{t'} = v_j | X_0 = v_i), v_i, v_j \in H_t$ are evaluated using maximum likelihood estimates of $\mathscr{G}, \theta_c$.

Below we discuss parameter estimation over our application data set.

Here, with respect to our application domain (MIT data set) we are not given specific events. So, we consider month wise contact as events.

5.3. PARAMETER ESTIMATION AND ESTIMATION FROM DATA

We consider the reality mining data set [108] from the MIT media laboratory for our experiments. It is a data set collected over 94 subjects, provided with Nokia 6600 mobile phones, over the time period from September 2004 to June 2005. Since, bluetooth devices respond to other bluetooth devices within 5-10 meters, it resembles the contact establishment process. The scans performed over 5 minutes intervals creates logs with starting time of contact, duration of contact, the two phone numbers involved, i.e. the source and destination of contact.

To estimate parameters, we divide the data into training and test data set. We estimate parameters over the training data set. In Subsection 5.4 we show the prediction outcomes over test data set. To be precise, we use k-fold cross validation [78], where data is divided into k -equal portions. Over each iteration we learn over $k-1$ data portions and validate our prediction over the remaining data portion. Thus, we repeat the process k-times and take the average to have our parameter estimates and prediction outcomes.

From Subsection 5.2.1, to evaluate the parameters in θ, we first need to obtain initial estimates of the frequencies (ω_c) from standard spectral analysis with FFT (using *fftwtools* package in *R*). With known frequency initial estimates for φ_c, A_c can also be computed.

Let $\phi_h, \phi_d, \phi_w, \phi_m, \phi_s$ correspond to the relative change for hour, day, week, month and semester respectively or the interval of the day (with half hourly intervals, there is total of 48 intervals, whereas with hourly intervals, there is 24 intervals in total), the day or the week or the month or the semester in which time t falls. On the circumference of circle, we take into account 24 hours, 7 days, 4 weeks, 12 months, 3 semesters (Fall, Spring, Winter break) respectively. Let n_h, n_d, n_w, n_m, n_s be the number of contacts that occur during hour h, day d, week w, month m, semester s respectively. We now define $n_h^{total} = \sum_{h=0}^{23} n_h$, $n_d^{total} = \sum_{d=0}^{6} n_d$, $n_w^{total} = \sum_{w=0}^{3} n_w$, $n_m^{total} = \sum_{m=0}^{11} n_m$, $n_s^{total} = \sum_{s=0}^{2} n_s$. Now we evaluate $\phi_h, \phi_d, \phi_w, \phi_m, \phi_s$ as follows:

$$\begin{aligned} \phi_h &= \frac{24 n_h}{n_h^{total}} \quad \phi_d = \frac{7 n_d}{n_d^{total}} \quad \phi_w = \frac{4 n_w}{n_w^{total}} \\ \phi_m &= \frac{12 n_m}{n_m^{total}} \quad \phi_s = \frac{3 n_s}{n_s^{total}} \end{aligned} \tag{5.18}$$

Now, we evaluate $\lambda^0 = \frac{n_s^{total}}{t_{n+1} - t_0}$. Further, λ is defined relative to specific set of cyclic pattern, and within specific cycle we have the needed parameters evaluated earlier.

We use the values corresponding to $\phi_h, \phi_d, \phi_w, \phi_m, \phi_s$ as the magnitude of ω_c corresponding to the respective cycle. Once we have ω_c, we can evaluate φ_c, A_c. With all the needed parameters we can evaluate λ_r and rest of the dependent components/variables. Here we do not use any specific Estimation approach as from the data set we can classify contacts as per requirement.

The multiplicative decomposition $\lambda = \lambda^0 \phi_h \phi_d \phi_w \phi_m \phi_s$ indicates the occurrence of the event on specific hour of the day, day of the week, week of the month, month of the semester and the semester itself. In fact, the contact rate over the transitions from one cyclic pattern to another can be stated as follows:

$\lambda^{transition} = \lambda^0 \phi_h \phi_d$ or,

$\lambda^{transition} = \lambda^0 \phi_d \phi_w$ or,

$\lambda^{transition} = \lambda^0 \phi_w \phi_m$ or,

$\lambda^{transition} = \lambda^0 \phi_m \phi_s$ respectively, based on the current and subsequent cyclic pattern (whether it is hourly and daily or daily and weekly or so on).

In evaluating the effect of direct recurrent pattern of contacts, though we have the relative periodic changes and the corresponding evaluation methodology for ω_c, φ_c and A_c, what we lack is the association of events (observed time instances) to respective periods/cycles. So, we use the existing Expectation Maximization (EM) algorithm [115] (using the EM package in *R*) to iteratively maximize the assignment of event instances to cycles based on the optimum probability distribution and also infer the model parameters. The EM algorithm iteratively updates the probability distribution over the cycle assignments and maximum likelihood of parameter estimates by going through E-step and M-step. The two step process proceeds until it converges to a local optimum of the incomplete data likelihood. In the E-step, a probability distribution over cycle assignments, conditioned on the current parameter estimates is computed. In the M-step, the parameter estimates are updated to their maximum likelihood values, conditioned on the current distribution of cycle assignments. After several repetitions of the E-step and M-step the algorithm converges. We have considered 30 iterations per training data set.

For brevity, the resulting uniform and transition intensity function is shown in Figure 5.4 (1st two plots) for monthly cycles only. The probability density function for the transition probability s_{ij} is in the 3rd plot.

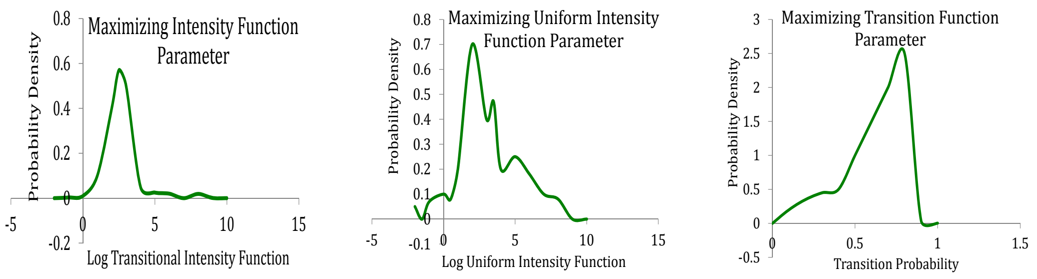


Figure 5.4. Intensity and Transition Function

In Subsection 5.4, we evaluate the prediction accuracy of future contacts over our application data set.

5.4. EXPERIMENTAL RESULTS

To predict number of contacts, future contact time we analyze the statistical features over our experimental data set. From the given data distribution, we see there is a sudden drop of contacts over the month of September and February. For brevity, only weekly and monthly overall contact distributions are shown in Figure 5.5.

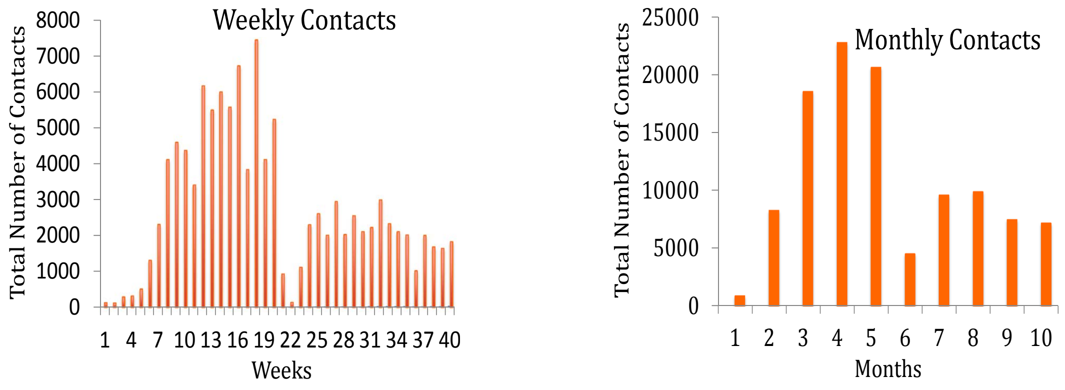


Figure 5.5. Weekly and Monthly Contacts

We organize the data over all users as a group, per user, and pair of users relative to hour, day, week and semester. Once we have the estimated parameters over the respective organized training data set, we validate the model of its prediction accuracy over the test data set. We use the formulations from Subsection (5.2.2), to predict the total number of contacts, future contact time over respective time frames. Using k- fold cross validation with 9 months, we consider 8 months as training data set whereas, data over 1 month is test set. We iterate 9 times, to have each month acting once as our test set and the remaining 8 months as training set for the respective iteration. We take average of the Parameter and prediction outcome over the 9 iterations as the as the prediction outcome. We use the F-score to validate the test (prediction) accuracy.

Figure 5.6, shows the accuracy of prediction for total number of contacts for all participants and relative to a single participant (say participant id 82), over hourly, daily, weekly, monthly, semesterly periods. The accuracy over single participant is higher owing to less variability or unpredictability associated with multiple participants/users. Similarly, the accuracy decreases with higher prediction prediction interval. The accuracy of 0.82 to 0.9 (per user case) reflects the the data to be of periodic nature (owing to the periodicity of human mobility pattern), though there is also random component that is more prominent over all user.

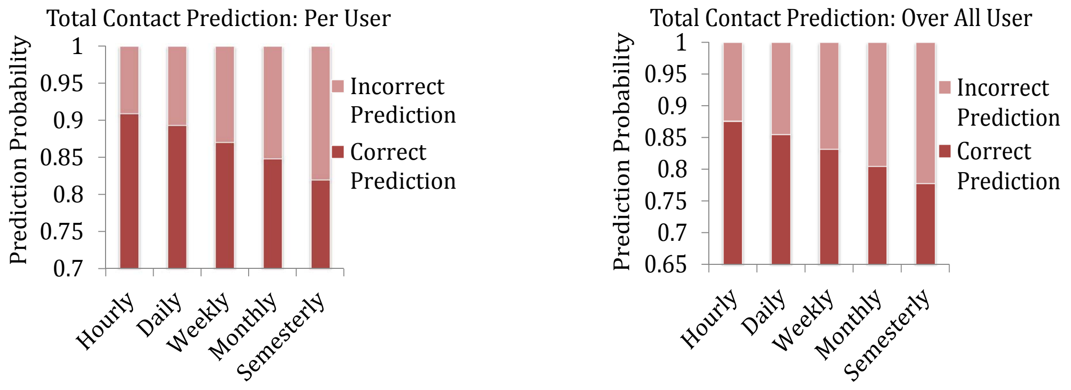
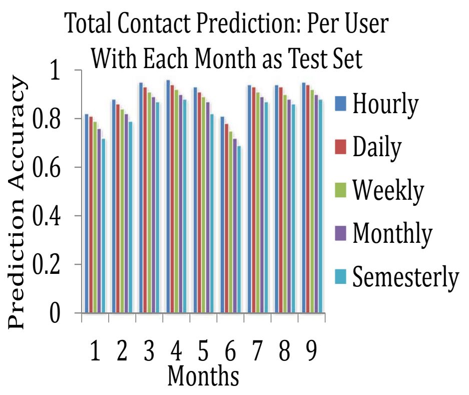


Figure 5.6. Total Contact Prediction Averaged over Test Set

We examine the prediction performance over each test set (i.e. over each month that acts as the test set over respective iteration), Figure 5.7. The prediction accuracy over the months of September and February are lowest, this might be owing to the sudden drop in contacts in these two months, possibly as a result of random events.

The average future contact time over test set relative to both individual user (say, 96) and user pair (say, 96 and 29) is in Figure 5.8, with the later having more affirmative prediction accuracy. This is possibly due to the definite identity of the two contacting nodes, where as individual user has more variability (all other users it is in contact).

Finally, we compare the effectiveness of our prediction model ($N(t)$) relative to the doubly recurrent model and HPP model, our model outperforms the other two. For brevity, we predict only the total contacts per user and over all users, Figure 5.9 (contact time



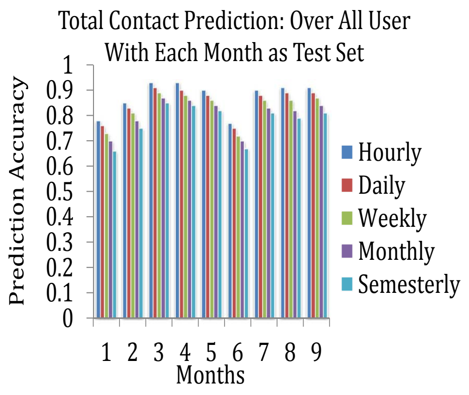
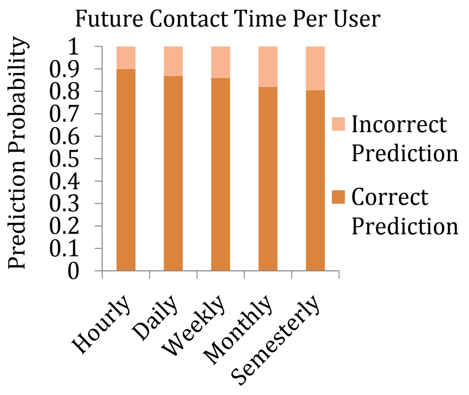


Figure 5.7. Monthly Contact Prediction Accuracy



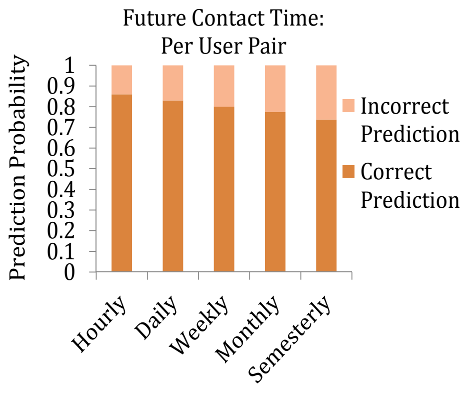
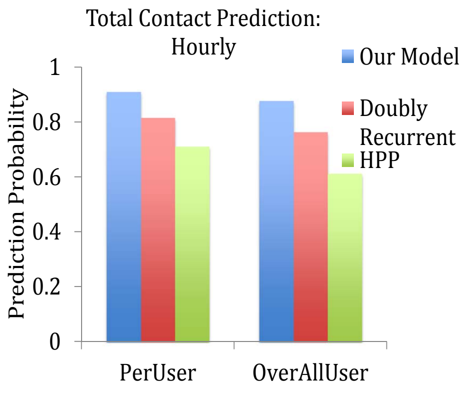


Figure 5.8. Future Contact Time Prediction Accuracy

comparison is obvious and follows similarly). In all three models the prediction accuracy is better in case of per user evaluation, than over all users; owing to the less variability in case of single user. Further, we also show that this holds over distinct time intervals (here, showing outcome for hourly and monthly intervals only).



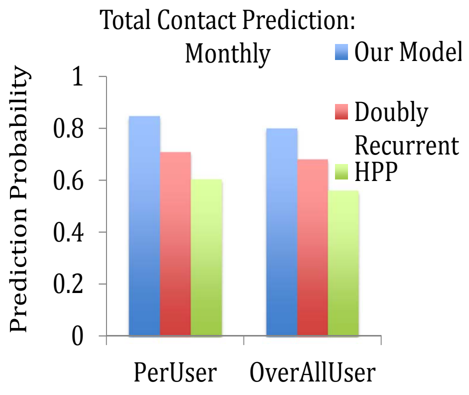


Figure 5.9. Model Effectiveness Evaluation: Total Contact

5.5. SUMMARY

In this part of the dissertation, we propose a novel stochastic contact prediction model that takes into account both multiple recurrent nature of contact patterns, along with direct dependency on recurrent events. The higher prediction accuracy of our model relative to HPP and doubly periodic models, lies in isolating/assigning events relative to specific recurrent pattern (i.e hourly, daily, weekly, monthly and semester wise patterns) and capturing their dependency. We validate our prediction model with F-score that shows to have value above 0.82 (highest 0.909) and 0.778 (highest 0.876) for total number of contacts in case of single and all participants respectively. We have also discussed the prediction accuracy of future contact time of a user and a user pair.

The monthly prediction accuracy Figure (5.7) has lowest value for September and February (0.66 to 0.72). This implies extending the model to capture random events and their inter-dependency. Thus, it is natural to extend the present model to an additive Poisson process model, to reflect not only sudden changes in contact pattern, but also also investigate any possible random contacts and their inter-dependency. This is how our proposed process model $N(t)$ comes up. We also describe the framework to capture the random contact patterns using an $MMPP$ model.

6. CONCLUSIONS

The main challenge in successfully and efficiently implementing the applications and services over large scale networks is not only the size of the network but also the network structure and its underlying intrinsic dynamics that gives rise to core conceptual problems over them, such as evaluating *centrality* metric (betweenness centrality), *predicting future links and contacts*. In this dissertation, we address these three core conceptual challenges from the perspective of underlying network structure and the intrinsic network dynamics (temporal characteristics), that help overcome these critical bottlenecks in successful implementation of application services.

Issues like intrinsic community formation, influential nodes (edges), future links and contacts are addressed over dynamic networks with and without node mobility. These aspects of dynamic networks are the basis behind successful applications over real world networks. Though, we have considered online social networks, the Internet, and mobile network data sets to experimentally validate our approach, our approach is equally applicable to other real world networks, that are known to be intrinsically dynamic.

We present three of our works in Section Three, Section Four and Section Five, respectively. Here, in Subsection 6.1, we present the summary of the contributions in this dissertations and in Subsection 6.2 we present our future research work.

6.1. SUMMARY OF CONTRIBUTIONS

We propose an efficient betweenness centrality computation approach in our third section. With large scale networks reflecting intrinsic community structure, we aim to leverage the divide and conquer algorithmic strategy in proposing the betweenness centrality algorithm. In the process we first propose a community formation/detection algorithm. It leverages the power law based degree distribution along with incremental accumulation and creation of virtual nodes, and semi-local optimal node selection strategy in detecting communities in a computationally efficient manner. Over sparse graphs with $|E| \sim |V|$, our algorithm costs $O(|V| - m|V|)$, $m < 1$, $k \sim 1$ computation time, whereas over dense graphs with $|E| \simeq |V|^2$, the proposed algorithm incurs a cost of the order of $O(|V|^2 - mk^2|V|)$ and is computationally efficient than existing algorithms. Further, over the community structured network we propose our betweenness centrality algorithm. It takes advantage of the dense intra-community edges and sparse inter-community edges over the intrinsically clustered network structure. The computation of betweenness centrality incurs cost of $O(|V|^2 + \frac{1}{2}|V|^{\frac{3}{2}} log|V|)$, with computational cost better than existing betweenness centrality approaches.

In the fourth section, we propose a novel link prediction strategy over intrinsically dynamic networks. The problem is interesting not only because of its use in several applications, but also because of the computational challenges of prediction accuracy over large scale, sparse networks. Our approach not only takes into consideration the occurrence of new links between node pairs, but also takes into account the occurrence of recurring links between them. Our probabilistic approach is inspired by the network evolution process, where interactions or occurrence of links between node pairs is a cascading process. In fact, the occurrence of links over a time frame gives rise to the possible creation of communities. The links, themselves are occurrences over smaller time-frame. Thus, we visualize the links and communities as microscopic and macroscopic structural elements over two distinct time frames. We take into account their evolution and effect in formation of future links. We also investigate the improvements in accuracy with information theoretic analysis, where entropy and prediction accuracy are interrelated. We validate the approaches over online social networks like, Twitter, Facebook, Enron email network, and DBLP co-authorship network. More so, we evaluate the performance (prediction accuracy) of the approaches relative to existing benchmarked stationary and temporal approaches. The quantitative results show our Markov model outperforms the two recent dynamic approach by 16.42% to 19.81% and 7.5% to 15.88% respectively, and obviously outperforms the state-of-the-art static approaches Further, the information theoretic approach has resulted negligible performance improvement over our approach. We also obtain an upper bound on number of history states to be taken into consideration for optimal prediction accuracy and derive that it is in *logarithmic* order of the number of states.

We present the novel contact prediction approach over mobile networks in the fifth section. The intermittent connectivity associated with mobility of the network elements is a challenge for successful prediction. The interesting point here is the fact that human mobility patterns are non-homogeneous and periodic in nature. More so, his movement dependent activities are multi-periodic in nature. More so, there is also possible random contacts without any periodic occurrence. Here, we propose a cascaded non-homogeneous Poisson process model to capture the periodic contacts and the underlying contact dependency, with a sinusoidal function as the intensity function to take care of the multi-periodic nature. We also propose the Markov modulated Poisson process model to capture the random contacts. The integration of both the model counts for prediction accuracy of number of contacts and contact time over the application domain. We give a detailed experimental evaluation of our approach and compare its accuracy with existing homogeneous Poisson process models and doubly periodic models. Further, our ongoing work reflects that among the two class of prediction models, i.e. the network evolution based probabilistic temporal models and the non-homogeneous Poisson process based models neither is suitable for the other class of problems.

6.2. FUTURE WORK

We propose to improve on our community formation algorithm over aggregated dynamic network and its incremental version over temporal network to develop a dynamic community formation algorithm. Further, we also propose to use the incremental and dynamic version of the community formation algorithm, to formulate a computationally efficient temporal betweenness centrality algorithm. We also aim to parallelize the algorithms to obtain higher performance.

Further, considering the fact that the success of many applications over temporal (and dynamic) networks is dependent on link (contact) prediction accuracy. We would like to analyze the effect and hence propose effective algorithms and mechanisms (frameworks) for information diffusion, routing and smart environment (smart city) management applications. Here, we also plan to investigate the energy efficient approaches/ algorithms.

In the link prediction problem over dynamic networks we would like to justify the varying consideration of specific time stamps and slots over different application domains, say over Twitter and Enron e-mail network.

We also propose a hybrid $SARIMA$ and RNN model based on our current work, to predict future contacts over mobile networks.

More so, our underlying network graphs are binary in nature, in essence the edges (and nodes) either exist or do not exist. So, the inherent question is how about random graphs? Well, the solution strategy for link prediction problem intrinsically considers this aspect, by proposing a probabilistic model where the link occurrence is the computed prediction probability and we consider links above certain threshold as discussed in detail there. Considering this link (occurrence) probability in fact gives us a visualization of random graphs. A more relevant and detailed question we would like to address is how the considered problems vary with random graphs. Realistically enough, the real world networks in fact resemble random graphs, where the physical reality of link (link occurrence) happens when it exceeds certain threshold probability. More so, over a temporal domain it also integrates this time varying occurrence probabilities, while physical existence of links happens. In fact, some of the basic network models [52] studies the possibility of random graphs as the basis for generating (replicating or reflecting) the real world networks and investigated how closely the proposed random graph models resemble the real networks.

They also considered problems like centrality, network evolution, disease propagation etc., though these random graphs later reflected their inability in replicating real-world networks and their dynamics.

REFERENCES

[1] S. F. Adafre and M. de Rijke. Discovering missing links in wikipedia. In *Proceedings of the 3rd ACM International Workshop on Link Discovery*, pages 90–97, 2005.

[2] C. Aggarwal and K. Subbian. Evolutionary network analysis: a survey. *ACM Computing Surveys (CSUR)*, 47(1):1–10, 2014.

[3] C. C. Aggarwal, Y. Xie, and S. Y. Philip. On dynamic link inference in heterogeneous networks. In *SDM*, pages 415–426. SIAM, 2012.

[4] C. C. Aggarwal, Y. Xie, and P. S. Yu. A framework for dynamic link prediction in heterogeneous networks. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 7(1):14–33, 2014.

[5] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social Network Data Analytics*, pages 243–275. Springer, 2011.

[6] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of Network and Computer Applications*, 37:380–392, 2014.

[7] R. Albert and A.-L. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–97, 2002.

[8] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 635–644, 2011.

[9] D. A. Bader, S. Kintali, K. madder, and M. Mihail. Approximating betweenness centrality. In *WAW*, 2007.

[10] M. Baglioni, F. Geraci, M. Pellegrini, and E. Lastres. Fast exact computation of betweenness centrality in social networks. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 450–456, 2012.

[11] A.-L. Barabási. Linked: The new science of networks, 2003.

[12] A. L. Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435:207–211, 2005.

[13] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[14] A. Barrat, M. Barthelemy, and A. Vespignani. *Dynamical processes on complex networks*. Cambridge university press, 2008.

[15] V. Batagelj. Semirings for social networks analysis. *Journal of Mathematical Sociology*, 19(1):53–68, 1994.

[16] V. Batagelj and A. Mrvar. Pajek – analysis and visualization of large networks. In *GRAPH DRAWING SOFTWARE*, pages 77–103. Springer, 2003.

[17] A. Bavelas. A mathematical model for group structures. In *Human Organization*, volume 7, pages 16–30, 1948.

[18] A. Beach, M. Gartrell, and H. Richard. Solutions to security and privacy issues in mobile social networking. In *Proc. of IEEE International Conference on Computational Science and Engineering, 2009.*, volume 4, pages 1036–1042. IEEE, 2009.

[19] M. A. Beauchamp. An improved index of centrality. *Behavioral Science*, 1:161–163, 1965.

[20] C. A. Bliss, M. R. Frank, C. M. Danforth, and P. S. Dodds. An evolutionary algorithm approach to link prediction in dynamic social networks. *Journal of Computational Science*, 5(5):750–764, 2014.

[21] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.

[22] U. Brandes. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos, to appear in special issue on Complex Networks' Structure and Dynamics*, 2008.

[23] B. Bringmann, M. Berlingerio, F. Bonchi, and A. Gionis. Learning and predicting the evolution of social networks. *IEEE Intelligent Systems*, 25(4):26–35, 2010.

[24] M. Brown. Statistical analysis of non-homogeneous poisson processes. *Stochastic point processes*, pages 67–89, 1972.

[25] Y. Cao, X. Chen, T. Jiang, and J. Zhang. Socast: Social ties based cooperative video multicast. In *Proceedings of The IEEE INFOCOM*, pages 415–423, 2014.

[26] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.

[27] I. Chlamtac, M. Conti, and J. J.-N. Liu. Mobile ad hoc networking: imperatives and challenges. *Ad hoc networks*, 1(1):13–64, 2003.

[28] R. I. Ciobanu and C. Dobre. Predicting encounters in opportunistic networks. In *Proceedings of the 1st ACM workshop on High performance mobile opportunistic systems*, pages 9–14, 2012.

[29] A. Clauset and N. Eagle. Persistence and periodicity in dynamic proximity networks. In *In Proc. DIMACS Wrkshop on Computational Methods for Dynamic Interaction Network*, 2007.

[30] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, pages 1– 6, 2004.

[31] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.

[32] V. Conan, J. Leguay, and T. Friedman. Characterizing pairwise inter-contact patterns in delay tolerant networks. In *Proceedings of the 1st international conference on Autonomic computing and communication systems*, page 19. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[33] M. Conti and M. Kumar. Opportunities in opportunistic computing. *Computer*, 43(1):42–50, 2010.

[34] T. H. Coremen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001.

[35] T. M. Cover and J. A. Thomas. *Elements of information theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[36] E. M. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '07, pages 32–40, New York, NY, USA, 2007. ACM.

[37] S. Das and S. K. Das. Leveraging network structure in centrality evaluation of large scale networks. In *40th IEEE Conference on Local Computer Networks (LCN)*, pages 579–586, 2015.

[38] S. Das and S. K. Das. Multi-periodic contact patterns in predicting future contacts over mobile networks. In *18th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–10. IEEE, 2017.

[39] S. Das and S. K. Das. A probabilistic link prediction model in time-varying social networks. In *IEEE International Conference on Communications (ICC)*, pages 1–6, 2017.

[40] S. Das, S. K. Das, and S. Ghosh. Inferring future links in large scale networks. In *41st IEEE Conference on Local Computer Networks (LCN)*, pages 244–252, 2016.

[41] S. Das, S. K. Das, and S. Ghosh. Leveraging contact pattern to predict future contact pattern in mobile networks. In *Proceedings of the 8th ACM International Workshop on Hot Topics in Planet-scale mObile computing and online Social neTworking*, pages 13–18. ACM, 2016.

[42] S. Das, J. Leopold, S. Ghosh, and S. K. Das. Graph partitioning in parallelization of large scale networks. In *41st IEEE Conference on Local Computer Networks (LCN)*, pages 176–179. IEEE, 2016.

[43] S. Das, J. Leopold, S. Ghosh, and S. K. Das. Graph compaction in analyzing large scale online social networks. In *IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[44] E. Della Torre and C. Longo. Poisson's equation in inhomogeneous media. *Proceedings of the IEEE*, pages 1141–1142, 1965.

[45] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal Statistical Society. Series B (methodological)*, pages 1–38, 1977.

[46] H. Deng, J. Han, B. Zhao, Y. Yu, and C. X. Lin. Probabilistic topic models with biased propagation on heterogeneous information networks. In *Proceedings of the 17th ACM SIGKDD*, pages 1271–1279, 2011.

[47] H. T. Dinh, C. Lee, D. Niyato, and P. Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, 2013.

[48] Y. Dong, J. Tang, S. Wu, J. Tian, N. Chawla, J. Rao, and H. Cao. Link prediction and recommendation across heterogeneous social networks. In *12th IEEE International Conference on Data Mining (ICDM)*, pages 181–190, 2012.

[49] E. Email. Enron e-mail data set, 2015.

[50] D. Eppstein, M. T. Goodrich, M. Löffler, D. Strash, and L. Trott. Category-based routing in social networks: Membership dimension and the small-world phenomenon. *Theoretical Computer Science*, 514:96 – 104, 2013.

[51] D. Erdős, V. Ishakin, A. bestavros, and E. Terzi. A divide and conquer algorithm for betweenness centrality, 2015.

[52] P. Erdos and A. Rényi. On the evolution of random graphs. *Bull. Inst. Internat. Statist*, 38(4):343–347, 1961.

[53] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[54] F. D. V. Fallani, V. Latora, L. Astolfi, F. Cincotti, D. Mattia, M. G. Marciani, S. Salinari, A. Colosimo, and F. Babiloni. Persistent patterns of interconnection in time-varying cortical networks estimated from high-resolution eeg recordings in humans during a simple motor act. *Journal of Physics A: Mathematical and Theoretical*, 41(22):224014, 2008.

[55] M. Feder, N. Merhav, and M. Gutman. Universal prediction of individual sequences. *IEEE Transactions on Information Theory*, 38(4):1258–1270, 1992.

[56] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 1977.

[57] L. C. Freeman, S. P. Borgatti, and D. R. White. Centrality in valued graphs: a measure of betweenness based on network flow. *Social Networks*, 13:141–154, 1991.

[58] M. P. Freeman, N. W. Watkins, E. Yoneki, and J. Crowcroft. Rhythm and randomness in human contact. In *Proc. IEEE International Conference onAdvances in Social Networks Analysis and Mining (ASONAM)*, pages 184–191, 2010.

[59] M. R. Garey and D. S. Johnson. *Computers and intractability; A guide to the theory of NP-completeness*. W. H. Freeman & Co., 1990.

[60] R. Geisberger, P. Sanders, and D. Schultes. A faster algorithm for betweenness centrality. *SIAM*, 2001.

[61] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure. In *ICML*, volume 1, pages 170–177, 2001.

[62] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[63] A. L. Goel and K. Okumoto. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE transactions on Reliability*, 3:206–211, 1979.

[64] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, 2008.

[65] R. Groenevelt, P. Nain, and G. Koole. The message delay in mobile ad hoc networks. *Performance Evaluation*, 62(1):210–228, 2005.

[66] R. Gross and A. A. Information revealation and privacy in online social networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, WPES '05', pages 71–80. ACM, 2005.

[67] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *IEEE Transactions on Mobile Computing*, 11(5):821–834, 2012.

[68] S. Hanneke, W. Fu, E. P. Xing, et al. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

[69] O. Hassanzadeh, A. Kementsietsidis, L. Lim, R. J. Miller, and M. Wang. Semantic link discovery over relational data. In *Semantic Search over the Web*, pages 193–223. Springer, 2012.

[70] P. Holme and J. Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[71] P. Holme and J. SaramÃd'ki. Temporal networks as a modeling framework. In *Temporal Networks*, Understanding complex systems, pages 1–14. Springer Berlin Heidelberg, 2013.

[72] A. Howard, M. J. Matarić, and G. S. Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *Distributed Autonomous Robotic Systems 5*, pages 299–308. Springer, 2002.

[73] Z. Huang and D. K. Lin. The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing*, 21(2):286–303, 2009.

[74] A. M. Kaplan and M. Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.

[75] D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 504–513, 2000.

[76] V. Kerbs. Inflow 3.1- social network mapping software, 2005.

[77] R. Kitchin. The real-time city? big data and smart urbanism. *GeoJournal*, 79(1):1–14, 2014.

[78] R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, volume 14, pages 1137–1145, 1995.

[79] M. Kolar, L. Song, A. Ahmed, and E. P. Xing. Estimating time-varying networks. *The Annals of Applied Statistics*, pages 94–123, 2010.

[80] G. Kossinets, J. Kleinberg, and D. Watts. The structure of information pathways in a social communication network. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 435–443, 2008.

[81] V. Kostakos. Temporal graphs. *Physica A: Statistical Mechanics and its Applications*, 388(6):1007–1023, 2009.

[82] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications*, pages 337–357. Springer, 2010.

[83] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong. Slaw: A new mobility model for human walks. In *Proc. IEEE INFOCOM*, pages 855–863, 2009.

[84] S. Lee, J. R. Wilson, and M. M. Crawford. Modeling and simulation of a nonhomogeneous poisson process having cyclic behavior. *Communications in Statistics-Simulation and Computation*, 20(2-3):777–809, 1991.

[85] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[86] P. Lewis and G. Shedler. Simulation of nonhomogeneous poisson processes with log linear rate function. *Biometrika*, 63(3):501–505, 1976.

[87] P. A. Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes by thinning. Technical report, DTIC Document, 1978.

[88] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[89] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences of the United States of America*, 102(33):11623–11628, 2005.

[90] R. N. Lichtenwalter and N. V. Chawla. Vertex collocation profiles: subgraph counting for link analysis and prediction. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1019–1028. ACM, 2012.

[91] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 243–252, 2010.

[92] Y. Lu and J. Garrido. Doubly periodic non-homogeneous poisson models for hurricane data. *Statistical Methodology*, 2(1):17–35, 2005.

[93] R. D. Malmgren, J. M. Hofman, L. A. Amaral, and D. J. Watts. Characterizing individual communication patterns. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 607–616, 2009.

[94] M. McNett and G. M. Voelker. Access and mobility of wireless pda users. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(2):40–55, 2005.

[95] R. Michalski, S. Palus, P. BrÃşdka, P. Kazienko, and K. Juszczyszyn. Modelling social network evolution. In *Social Informatics*, volume 6984 of *Lecture Notes in Computer Science*, pages 283–286. Springer Berlin Heidelberg, 2011.

[96] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[97] G. Miritello, E. Moro, and R. Lara. Dynamical strength of social ties in information spreading. *Phys. Rev. E*, 83:045102, Apr 2011.

[98] M. E. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126, 2003.

[99] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45:167–256, 2003.

[100] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69, Jun 2004.

[101] M. E. J. Newman. *Networks: An introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.

[102] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2), Feb. 2004.

[103] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora. Graph metrics for temporal networks. In *Temporal Networks*, pages 15–40. Springer, 2013.

[104] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora. Graph metrics for temporal networks. *CoRR*, 2013.

[105] P. Olsson and C. Folke. Adaptive comanagement for building resilience in social-ecological systems. *Environmental Management*, 34(1):75–90, July 2004.

[106] H. P. Network reachability of real world contact sequences. *Phys. Rev. E*, 71, 2012.

[107] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *Communications Magazine, IEEE*, 44(11):134–141, 2006.

[108] A. Pentland, N. Eagle, and D. Lazer. Inferring social network structure using mobile phone data. *Proceedings of the National Academy of Sciences (PNAS)*, 106(36):15274–15278, 2009.

[109] A.-K. Pietilänen and C. Diot. Dissemination in opportunistic social networks: the role of temporal communities. In *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*, pages 165–174, 2012.

[110] G. P. R. On the geographical interpretation of eigenvalues. *Transactions of the Institute of British Geographers*, (42):53–86, Dec. 1967.

[111] E. Richard, S. Gaïffas, and N. Vayatis. Link prediction in graphs with autoregressive features. *The Journal of Machine Learning Research*, 15(1):565–593, 2014.

[112] W. Richards. Social network analysis software links, 2005.

[113] M. Riondato and E. M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 413–422. ACM, 2014.

[114] S. M. Ross. *Introduction to probability models*. Academic press, 2014.

[115] T. Rydén. An em algorithm for estimation in markov-modulated poisson processes. *Computational Statistics & Data Analysis*, 21(4):431–447, 1996.

[116] P. Sarkar, D. Chakrabarti, and M. Jordan. Nonparametric link prediction in dynamic networks. *arXiv preprint arXiv:1206.6394*, 2012.

[117] R. R. Sarukkai. Link prediction and path analysis using markov chains. *Computer Networks*, 33(1):377–386, 2000.

[118] S. Scellato, M. Musolesi, C. Mascolo, and V. Latora. On nonstationarity of human contact networks. In *Proc. 30th IEEE International Conference on Distributed Computing Systems Workshops*, pages 105–111, 2010.

[119] S. Scellato, A. Noulas, and C. Mascolo. Exploiting place features in link prediction on location-based social networks. In *Proceedings of the 17th ACM SIGKDD*, pages 1046–1054, 2011.

[120] A. Shimbel. Structural parameters of communication networks. *The Bulletin of Mathematical Biophysics*, 15(4):501–507, 1953.

[121] A. Solanas, C. Patsakis, M. Conti, I. Vlachos, V. Ramos, F. Falcone, O. Postolache, P. Perez-martinez, R. Pietro, D. Perrea, et al. Smart health: a context-aware health paradigm within smart cities. *IEEE Communications Magazine*, 52(8):74–81, 2014.

[122] M. Spiliopoulou. Evolution in social networks: A survey. In *Social Network Data Analytics*, pages 149–175. Springer, 2011.

[123] C. Staudt, A. Schumm, H. Meyerhenke, R. Gorke, and D. Wagner. Static and dynamic aspects of scientific collaboration networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 522–526, Aug 2012.

[124] J. Stehlé, A. Barrat, and G. Bianconi. Dynamical and bursty interactions in social networks. *Phys. Rev. E*, 81:035101, 2010.

[125] S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[126] J.-Z. Sun. Mobile ad hoc networking: an essential technology for pervasive computing. In *Proceedings of IEEE International Conferences on Info-tech and Info-net ICII*, volume 3, pages 316–321, 2001.

[127] J. Tang, M. Musolesi, C. Mascolo, and V. Latora. Temporal distance metrics for social network analysis. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 31–36. ACM, 2009.

[128] J. Tang, S. Scellato, M. Musolesi, C. Mascolo, and V. Latora. Small-world behavior in time-varying graphs. *Physical Review E*, 81(5), May 2010.

[129] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Advances in Neural Information Processing Systems*, 2003.

[130] T. Tylenda, R. Angelova, and S. Bedathur. Towards time-aware link prediction in evolving social networks. In *Proceedings of the 3rd Workshop on Social Network Mining and Analysis*, page 9. ACM, 2009.

[131] M. Valencia, J. Martinerie, S. Dupont, and M. Chavez. Dynamic small-world behavior in functional brain networks unveiled by an event-related networks approach. *Phys. Rev. E*, 77:050905, May 2008.

[132] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2Nd ACM Workshop on Online Social Networks*, WOSN '09, pages 37–42, New York, NY, USA, 2009.

[133] B. Viswanath, A. Post, and K. P. Gummadi. An analysis of social network-based sybil defenses. In *Proc. of the ACM SIGCOMM*, volume 40, pages 363–374, 2010.

[134] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440, 1998.

[135] S. White and P. Smyth. A spectral clustering approach to finding communities in graphs. In *In SIAM International Conference on Data Mining*, 2005.

[136] G. Wiki. Dataset for our experiment, 2014.

[137] H. Wu, J. Cheng, S. Huang, Y. Ke, Y. Lu, and Y. Xu. Path problems in temporal graphs. *Proceedings of the VLDB Endowment*, 7(9):721–732, 2014.

[138] L. Yang, H. Jiang, S. Wang, L. Wang, and Y. Fang. Characterizing pairwise contact patterns in human contact networks. *Ad Hoc Networks*, 10(3):524–535, 2012.

[139] J. Zhao and G. Cao. Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 57(3):1910–1922, 2008.

[140] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.

VITA

The author received her doctorate in Computer Science from the University of Missouri Science and Technology at Rolla in Decemebr 2017. Prior to that she was a research associate in the Computer Science and Engineering Department, Indian Institute of Technology, Kharagpur, and worked with Professor Sudeb Kumar Prasant Pal. Before that she received her Masters from the Jadavpur University and did her MTech research under the guidance of Professor Susmita Ghosh. She received UGC/AICTE Fellowship and Institute Fellowship, and was selected for Infosys Fellowship.