Network Reconstruction under Compressive Sensing

Payam Siyari*, Hamid R. Rabiee†, Mostafa Salehi* and Motahareh Eslami Mehdiabadi *
Department of Computer Engineering, Sharif University of Technology
* Email: {siyari, mostafa_salehi, eslami}@ce.sharif.edu
† Email: rabiee@sharif.edu

***Abstract*—Many real-world systems and applications such as World Wide Web, and social interactions can be modeled as networks of interacting dynamical nodes. However, in many cases, one encounters the situation where the pattern of the node-to-node interactions (i.e., edges) or the structure of a network is unknown. We address this issue by studying the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? We propose a novel framework called CS-NetRec based on a newly emerged paradigm in sparse signal recovery called Compressive Sensing (CS). The general idea of using CS is that if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements. In particular, we utilize the observed data of information cascades in the context of CS for network reconstruction. Our comprehensive empirical analysis over both synthetic and real datasets demonstrates that the proposed framework leads to an efficient and effective reconstruction. More specifically, the results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). Furthermore, our framework is capable of near-perfect reconstruction of the desired network in presence of 95% sparsity. In addition, we compared the performance of our framework with NetInf; one of the state-of-the-art methods in inferring the networks of diffusion. The results suggest that the proposed method outperforms NetInf by an average of 10% improvement based on the F-measure.**

I. Introduction

In many scientific and engineering applications, the systems under study can be modeled as a set of networked elements, called nodes. Usually, depending on the domain, the interactions between these elements are shown as the edges between the nodes. In large scale networks, the node-to-node interactions or the network structure is not usually known. In such situations, it is important to propose an efficient method to reconstruct the network structure based on partial observations. This issue is known as the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements?

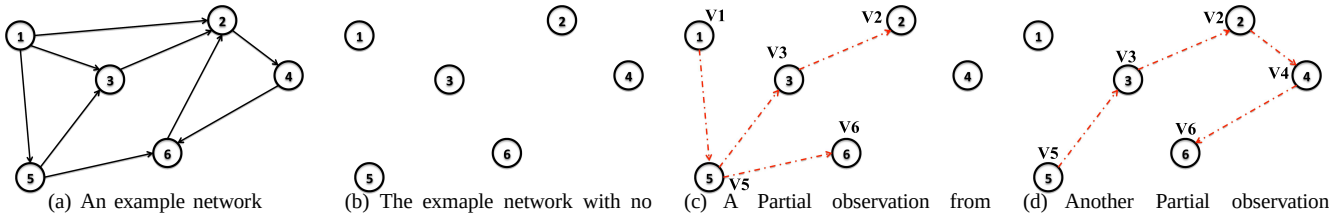
Network reconstruction problem is encountered in many real-world situations. However, it is still a challenging issue to be addressed by introduction of new frameworks. For example, in biological systems, although there has been a great effort in improving the technologies to uncover the Protein interaction data, there are several reports of their inaccuracies [2]. In the analysis of social networks, particularly online social networks, the existence of missing data is almost inevitable due to several reasons, e.g. security, user privacy, data aggregation overhead, etc. Analysis of such data can lead to misleading estimation of network properties [3], [4].

An illustrative example of network reconstruction problem is provided in Figure I. An example network is shown in Figure 1(a), and the network reconstruction problem where only the network's nodes are available is shown in Figure 1(b). Here, the goal is to reconstruct the network in Figure I based on the partial observations in Figures 1(c) and 1(d). In the context of information diffusion, we assume that the observations in Figures 1(c) and 1(d) are the outputs of some processes which are run over the edges of the network. Each process measures a value for each node and its output is a function of these values. Based on the node values and process outputs (and not the edges that process has been run over), we aim to propose a reconstruction scheme for the network. Depending on the domain of study, the observations can be more implicit or complex. In this paper, for the partial observations over the network, we consider information cascades (e.g. virus propagation) as the process, and node hit times (e.g. node infection times) as the node values. We will cover the concepts related to the information networks and information diffusion in Section III.

Although the problem of network reconstruction has been analyzed in various contexts in different approaches [6]–[9], [13], [14], in this paper for the first time, we introduce a general framework called "CS-NetRec"(**C**ompressive **S**ensing for **N**etwork **R**econstruction), based on the concept of Compressive Sensing (CS); a recently emerged paradigm for efficient sparse-signal recovery. Our motivation for using CS is that it can provide a concrete mathematical framework for the problem of network reconstruction.

The basic idea in CS [10]–[12] is that in an appropriate lower dimensional representation (e.g. sparse vector, low-rank matrix, etc.), the under-sampled data of a signal have all the information needed about that signal. In other words, if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements.This means that a signal can be reconstructed from a small set of sampled data. In many real-world situations, the existing sparsity in the network structure, helps to make this technique applicable.

In this paper, we utilize the cascade probability data from



(a) An example network

(b) The exmaple network with no information about the edges

(c) A Partial observation from the network structure - the process output is denoted with $f(v_1, v_2, v_3, v_5, v_6)$.

(d) Another Partial observation from the network structure - the process output is denoted with $f(v_2, v_3, v_4, v_5, v_6)$.

Fig. 1. An example of the network reconstruction problem. (a) An example network. (b) The example network with known nodes and no information about the edges. This is the only information we have about the topology of the network in network reconstruction problem. (c,d) Example partial observations (i.e. node values and process outputs) over the graph used to reconstruct the network of interest.

the diffusion of an arbitrary type of information throughout the desired networked data. We consider each information cascade as the single measurements on the network structure. Then we estimate the probability that a cascade can diffuse over the network by considering the probability of the most likely tree related to a cascade. Finally, by formation of a linear system from the diffusion process, we utilize the theory of CS in order to reconstruct the network of interest.

We evaluated the proposed framework over both synthetic and real datasets with various configurations in terms of the dependency of its accuracy to the number of observations from the network. The results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network) with the F-measure above 0.5 where the number of possible edges to predict is 5 to 10 times more than the number of running cascades. As another part of our experimental results, we demonstrated the effect of different cascade parameters to the accuracy of the reconstruction. In addition, we evaluated the effect of the sparsity of the network structure on the performance of our framework. In the random graph model, we observed that when about 5% of the edges are available, meaning in the presence of 95% sparsity in the graph structure, we have a near perfect recovery with the F-measure above 0.9. Furthermore, we compared our framework with NetInf; one of the state-of-the-art methods for inferring the networks of diffusion. The comparison showed that for different cascade lengths, our method outperforms NetInf by an average of 9-10% improvement in terms of F-measure.

In summary, the main contributions of this paper can be stated as follows:

- Proposing a novel and general framework based on the rich mathematical framework of CS.
- Ability to reconstruct the underlying network without any knowledge about the topological features of the underlying network.

The rest of the paper is organized as follows. In Section II, we provide some of the related works and similar problems to network reconstruction problem and a short survey on the network analysis methods that utilize CS as a part of their algorithms. In Section III, we will state the problem details and the related concepts to the information networks. In Section IV, we introduce our framework in details. The experimental evaluation of the proposed framework is presented in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

A. Network Reconstruction

The problem of reconstructing the network structure from incomplete or missing data have been studied in many different contexts and different techniques. In [13], the main problem being studied is the network completion problem where a network with missing nodes and edges is given and the goal is to estimate the unobserved part of the network. However, in this work it is assumed that the underlying network follows the Kronecker graphs model. A similar problem is also studied as the inference of missing links in the context of survey sampling in [14], and in social and biological networks in [6]–[9].

A related problem to network reconstruction is the matrix completion problem [15] where a data matrix with missing entries is given, and the goal is to predict its missing elements. However, there still exist many challenges to utilize the theory of matrix completion in the reconstruction of the networks. As an example, several properties of real networks (e.g. heavy-tailed degree distribution, binary valued matrix entries) are not considered in the formulation of the matrix completion algorithms.

Also, the problem of link prediction relates to our work in which the aim is to predict the future edges of a network. Link prediction has been studied in social [16], and biological networks [17].

The most related problem to our work is where the goal is to infer hidden underlying network on which some type of information diffuses (e.g. virus, rumor, news, etc.) [5], [18]. Although this problem is a special case of the network reconstruction problem, we develop our general framework based on such configuration for better formulation and evaluation. In this work, we consider the diffusion of virus, rumor or any similar propagating information throughout the network. Some of the works in information networks try to find propagation links by using the structure and topological features of underlying network [19], [20]. However, we do not consider any topological assumptions about the underlying

network, and propose a framework which is based on the rich mathematical framework of CS.

B. Compressive sensing

The developments in compressive sensing began with the seminal works in [21], [22]. The authors showed that combining the l_1-minimization and random matrices can lead to efficient recovery of sparse vectors. Moreover, the authors showed that such concepts have strong potential to be used in many applications.

Consider the linear system:

$$y_{m\times 1} = A_{m\times n}\ x_{n\times 1} \tag{1}$$

Where $m << n$, and we are interested in finding a feasible value for x. Typically, the solution of a linear system can be obtained by the least squares minimization:

$$x^* = \underset{x}{arg\ min}\ \|Ax - y\|_2^2 \tag{2}$$

But when matrix A is rectangular, there exist infinite number of solutions. Thus, we have to add a constraint to the system so that we can limit the solution space. Here, we add the sparsity of x as a constraint to obtain a solution. Specifically, we assume x is k-sparse, meaning there is at most k nonzero elements in the vector x, and $k << n$. In CS theory, it is stated that the sparsest solution can be obtained by:

$$\underset{x}{min}\ \|x\|_0 \quad s.t. \quad Ax = y \tag{3}$$

Since solving Eq. (3) is NP-Hard, one can use l_1-minimization instead [21], [22]:

$$\underset{x}{min}\ \|x\|_1 \quad s.t. \quad Ax = y \tag{4}$$

Combining (2) and (4), we obtain:

$$\underset{x}{min}\ \|x\|_1 + \|Ax - y\|_2^2. \tag{5}$$

This change in the objective function (also known as LASSO [40], [41]) makes it possible to solve the linear system, even in presence of noise or truncated values in the matrix A, and vector y.

The l_1-minimization problem of (4) can be converted to a linear programming problem [10]. This leads to a set of algorithms in CS which are referred to as "Basis Pursuit"[22]. There also exist other sets of algorithms that use a greedy iterative approach, known as "Matching Pursuit"[24]. Moreover, there are several strong guarantees for the reconstruction through l_1-minimization [23].

As we make use of the CS technique in our approach, we would like to have a quick survey of some of the works that utilized this technique in the context of network analysis. CS has been mainly studied in the context of signal and image processing [25]–[27] and its use in the area of network analysis is still in its first stages of development.

The most use of CS in the network analysis has been in the field of wireless sensor networks [28]–[30]. The main idea concerns the recovery of information (e.g. the temperature of an area) based on the samples aggregated from the network sensors.

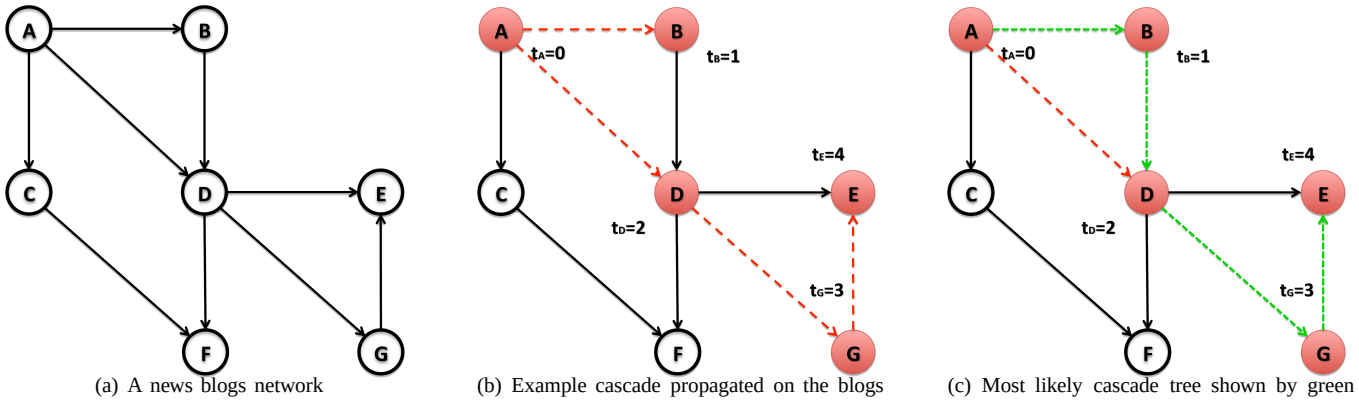
In [31], the question of whether it is possible to quickly infer and monitor the network link characteristics from indirect end-to-end (aggregate) measurements is analyzed. This question lies in the area of network tomography, and in this work different aspects of it are analyzed by using the CS theory. Also, in network traffic monitoring, we can mention the works in [33], [34]. In [33], CS theory is used in order to reduce the memory cost in routers and switches. In [34], CS is exploited in order to recover the missing values of a network traffic matrix. In [32], CS is used in the context of P2P networks. By exploiting CS theory, the authors devise an approach based on random walks to spread CS random combinations to participants in a random peer-to-peer (P2P) overlay network.

The other two works which are more related to ours are [35] and [36]. In both of these works, the authors aim to use CS as a tool to predict the topology of the network, although their settings and assumption are completely different from each other. In [35], the authors introduce a new approach based on the penalized linear regression to estimate sparse partial correlation between different regions of interests of the brain network, and by employing the CS theory they are able to have a successful recovery of the brain network. However, the proposed approach in that work can not be generalized in other areas and can only be used in the context of brain networks. In particular, we can not always define a partial correlation between different regions of the network. For many types of networks, such as online social networks, we don't have access to several sample subjects of the whole network. Hence, we can not define any correlation measure and we are unable to use their approach in networks other than networks with similar propertyes as brain networks. In [36], the authors articulate a general method of addressing the problem of how to uncover the network topology using evolutionary-game data based on compressive sensing. The key to solving the network reconstruction using evolutionary-game data problem lies in the relationship between the agents payoffs and strategies. The interactions among agents in the network can be characterized by an $N \times N$ adjacency matrix and the sparsity of a single node's neighborhood (adjacent nodes) makes the compressive sensing framework applicable.

III. Problem Formulation

A. Problem Statement

Consider the static directed network $G(V, E)$ with $|V| = n$ nodes and set of the edges E. We assume that we are totally aware of which nodes exist in the network and have no information about the edges. To reconstruct the network, we must predict the value of the elements of the adjacency matrix. Hence, we are looking for G^* where $\|G^* - G\| \leq \epsilon$. Ideally, we would like to have $\epsilon = 0$, but we examine a more general setting where it is possible to have noisy or truncated data. Obviously, this amount of information about the network is not sufficient for a tractable reconstruction. Thus, we need some external data about the network for efficient reconstruction.



(a) A news blogs network

(b) Example cascade propagated on the blogs network

(c) Most likely cascade tree shown by green dashed links.

Fig. 2. An example of information diffusion on a news blogs network. (a) A news blogs underlying network. (b) Example of a hidden real cascade tree (i.e. news coverage sequence) propagated on the blogs network. Each node hit by the cascade is shown in a differnet color and is labeld by a time stamp which shows the time of its news coverage. The cascade caused by the propagation of a particular news can be represented as: $News\ Cascade =< (\phi, A, t_A), (A, B, t_B), (A, D, t_D), (D, G, t_G), (G, E, t_E) >$. (c) The most likely cascade tree (shown by green dashed links) as an approximation for the hidden real cascade tree. It can be seen that the most likely cascade tree is not necessarily identical to the real cascade tree in (b).

We assume that there is an external process which has the following features:

1) It can be run several times on the network.
2) Its output can be considered as the linear combination of the edges of the network and a particular measure defined on each edge.

Here, we consider the information diffusion as the external process in which any type of information or disease (e.g. news headlines, virus, rumor, etc.) diffuses over an underlying network. As an example, Figure 2(a) shows a news blogs network, where the nodes represent the blogs. There can be many interpretations for the edges of such network. In this example, we can simply consider for any directed edge (u, v) that node u is one of the sources of the news for node v. It is usually assumed that the diffusion as an external process occurs on a network, meaning the information spreads over the edges of an underlying network. A simple example for the diffusion process is shown in Figure 2(b).The diffusion process on such network can be considered as the coverage of a particular news in the different blogs with different time stamps. However, the network over which propagations take place is usually unknown and unobserved [5]. The goal now is to reconstruct the unknown network over which news originally propagated so that we know each node's information sources.

Consider the network of news blogs in Figure 2(b). Obviously, news propagation acts as a diffusion process and happens frequently in such network, which satisfies the first property that we would like the external process to have. To show that the diffusion process satisfies the second property of an external process, we mention some preliminaries related to the information networks and then in the section IV, we show the second property for this process.

B. Information Networks and the cascading behaviour

Each diffusion of the information throughout the network, creates an information cascade. In other words, an information cascade can be considered as a sequence of nodes, that got hit by the diffusion process.

Definition 1. [5] A cascade is a set of triples $(u, v, t_v)_c$, which means that cascade c reached node v at time t_v by spreading from node u (by propagating over the edge (u, v)). We denote the fact that the cascade initially starts from some active node v at time t_v as $(\phi, v, t_v)_c$. In Figure 2(b), the cascade caused by the propagation of a particular news can be represented as:

$$News\ Cascade =< (\phi, A, t_A), (A, B, t_B), (A, D, t_D), (D, G, t_G), (G, E, t_E) >.$$

We assume the only data we can get from the cascades is the time that the cascade has reached a node. In the news blogs example in Figure 2(b), the cascade data can be considered as the coverage of a particular news in the different blogs ordered by the time stamp of the coverage. In information networks like our news blog example, we are usually not aware of the nodes sources of information. Although there are latent sources for any node, we ignore them and assume each node's information source is its neighbors in the underlying network. This fact happens commonly in blogs networks as we do not know how a blog got information about a particular news. Thus, we only get to observe pairs $(v, t_v)_c$.

Definition 2. Hit time can be described as the time t_v when node v got included by the cascade c. In news blogs example, the cascade hit times of each blog by the news cascade can be shown as:

$$Cascade\ Hit\ Times =< (A, t_A), (B, t_B), (D, t_D), (F, t_F), (G, t_G), (E, t_E) >$$

Now, given such data about node hit times for a number of different cascades, we aim to recover the unobserved directed

$$\begin{bmatrix} LP(c_1|G) \\ LP(c_2|G) \\ LP(c_3|G) \\ . \\ . \\ . \\ LP(c_m|G) \end{bmatrix} = \begin{bmatrix} LP_{c_1}(v_1,v_2) & ... & LP_{c_1}(v_i,v_j) & ... & LP_{c_1}(v_n,v_{n-1}) \\ LP_{c_2}(v_1,v_2) & ... & LP_{c_2}(v_i,v_j) & ... & LP_{c_2}(v_n,v_{n-1}) \\ LP_{c_3}(v_1,v_2) & ... & LP_{c_3}(v_i,v_j) & ... & LP_{c_3}(v_n,v_{n-1}) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ LP_{c_m}(v_1,v_2) & ... & LP_{c_m}(v_i,v_j) & ... & LP_{c_m}(v_n,v_{n-1}) \end{bmatrix} \begin{bmatrix} v_{1,2} \\ . \\ . \\ v_{i,j} \\ . \\ . \\ v_{n,n-1} \end{bmatrix}$$

Fig. 3. The linear system mapped from the diffusion of m cascades on the network where $LP(c_k|G) = \log P(c_k|G)$ and $LP_{c_k}(i,j) = logP_{c_k}(i,j)$. The vectorized network adjacency is the unknown vector which we know that is a sparse one. Note that the ordering of $LP_{c_k}(i,j)$s and $v_{i,j}$s must be the same.

network G, the network over which the cascades originally spread.

We consider the independent cascade model [37] which states that a node spreads the information to each of its neighbors independently with some chosen probability. This model implicitly assumes that every node v in cascade c is hit by at most one node u. That is, it only matters when the first neighbor of v spreads the cascade. In other words, only one neighbor of v actually activates v. Thus, the structure of a cascade c created by the diffusion process is fully described by a directed tree T, contained in the directed graph G.

C. Maximum probability cascade tree

Now, we aim to give an approximation for the directed tree T related to a cascade c. We use the probabilistic model of how cascades spread over the edges of the network.

Definition 3. [5] Define the probability $P_c(u,v)$ as the conditional probability of observing cascade c spreading from u to v. Since the cascade can only propagate forward in time, if node u got reached after node v ($t_u > t_v$), then $P_c(u,v) = 0$. As an intuition, the probability of propagation $P_c(u,v)$ between a pair of nodes u and v in the cascade c decreases with the more difference of their hit times. In news blogs networks, an old news most probably does not provide any information for any blog. For simplicity, we consider $P_c(u,v)$ to follow the well-known exponential cascade transmission distribution [38]:

$$P_c(u,v) = P_c(\Delta_{u,v}) = e^{-\frac{\Delta_{u,v}}{\alpha}} \tag{6}$$

Where $\Delta_{u,v} = t_v - t_u$ and α is the adjustment parameter.

A cascade stops with the probability $(1-\beta)$ and continues over an edge with the probability of β. The likelihood of a cascade spreading in a given tree pattern T is calculated as [5]:

$$P(c|T) = \prod_{u,v \in E_T} \beta P_c(u,v) \prod_{u \in V_T, (u,x) \in E \setminus E_T} (1-\beta) \tag{7}$$

Where $T = (V_T, E_T)$ is a tree.

Also the probability that a cascade c can occur in the graph G is defined as [5]:

$$P(c|G) = \sum_{T \in \tau_c(G)} P(c|T)\ P(T|G) \tag{8}$$

Where $\tau_c(G)$ is the set of all the directed connected spanning trees on a subgraph of G induced by the nodes that got hit by cascade c. In case T is inconsistent with the observed data, then $P(c|T) = 0$.

As there exist exponential number of possible trees, calculating the probability that a particular cascade can happen on an underlying network seems intractable. However, we use an approximation for the likelihood of a single cascade by considering only the most likely tree instead of all possible propagation trees.

$$P(c|G) = \max_{T \in \tau_c(G)} P(c|T)\ P(T|G) \tag{9}$$

We consider all possible trees to be equiprobable to occur. In other words $P(T|G) = \frac{1}{|\tau_c(G)|}$. The maximum probability tree can be found by choosing, for each node v, an incoming edge (u,v) with maximum probability [39]. Figure 2(c) shows the most probable cascade tree for the news cascade in Figure 2(b). As it can be seen, there is no necessity that the most likely cascade tree be the true cascade tree. However, intuitionally it can be considered as a reasonable approximation. We will show that this approximation will be suitable to our needs.

IV. Proposed Framework: CS-NetRec

Based on the assumptions in the previous section, we can run several cascades on the desired network. Hence, we run several cascades and use the cascades' probability data to reconstruct the network of interest. In particular, we form a linear system with the edges as the unknown vector, and try to find a solution for this system (i.e. a reconstruction for the network). More details can be seen in Algorithm IV.1.

Assuming T_c^* is the most likely tree corresponding to the cascade c, we hope that all the edges in T_c^* represent the real edges in the underlying network G, although this is not

necessary. Thus, after taking $\log$ from both sides of the Eq. (9), we can approximate it as the inner product of two vectors:

$$LP(c|G) = vLP(c|T_c^*)^T \ . \ vec(Adj(G)) \tag{10}$$

Where $LP(c|G) = \log P(c|G)$ and $vec(Adj(G))$ is the vectorized binary adjacency matrix of network G in some particular order, and $vLP(c|T_c^*)$ is the vector of individual edge transmission log probabilities with nonzero elements corresponding to the edges in T_c^*. In particular, the k-th element of $vLP(c|T_c^*)$ can be shown as:

$$vLP(c|T_c^*)_k = LP_c(i,j) = logP_c(i,j) \tag{11}$$

It is obvious that for the consistency of the formulation, the element order of both vectors (which correspond to all possible edges in the network G) must be the same. By this approximation, we are now able to show a single run of the diffusion process (i.e. a cascade) as the linear combination of the edges of the network, and the probability measure defined on each edge. Thus, the diffusion process satisfies the second property of an external process that was mentioned in the previous section.

Algorithm IV.1 CS-NetRec(C, V)

for each $c \in C$ **do**
  $T_c^* \leftarrow$ The Most Probable Tree for c. ▷ (Refer to Section III.C)
  for each $(i,j) \in V \times V$ **do**
    $LP_c(i,j) \leftarrow \begin{cases} \log P_c(i,j) & (i,j) \in T_c^* \\ 0 & o.w. \end{cases}$ ▷ (Eq. (6))
  end for
  Calculate $LP(c|G) = \log P(c|G)$. ▷ (Eq. (9))
  Form the vector $vLP(c|T_c^*)$ from $LP_c(i,j)$s. ▷ (Eq.(11))
  Add $LP(c|G)$ as a row to y.
  Add $vLP(c|T_c^*)$ as a row to A.
end for

$x^* \leftarrow \underset{x}{\textit{arg min}} \ \|x\|_1 + \|Ax - y\|_2^2$ ▷ (Refer to Section II.B)

return x^*

Now that the diffusion process satisfies both properties that an external process should have, we use this process and the CS framework in order to reconstruct the network of interest.

We can model m runs of the diffusion process as the linear system: $y = Ax$, where each equation in this linear system is equal to (10) and the unknowns are the edges of the desired network. More details about the formation of this linear system is shown in Figure 3.

Our algorithm consists of two main steps. First, the formation of the linear system and second, finding the solution for it.

For each cascade c (Which forms each equation in the linear system):

1) We find the most probable tree and set it to T_c^*.
2) For each possible edge in G, we calculate the edge transmission log probabilities. In particular, for each edge in T_c^* we use logarithm of Eq. (6) and for the rest of the edges in G we consider zero probability.
3) We calculate $LP(c|G) = \log P(c|G)$ from Eq. (9) and add it as a row to the vector y.
4) We form the vector $vLP(c|T_c^*)$ from previous step and add it as a row to the matrix A.

After the formation of the above linear system, Eq. (5) is used to find a sparse solution for the system.

V. Experimental Evaluation

In this section, we evaluate the performance of the proposed approach for network reconstruction. First we introduce the synthetic and real datasets we use for the evaluation. Next, we introduce the evaluation metrics, and finally we provide an analysis on the results.

A. Datasets

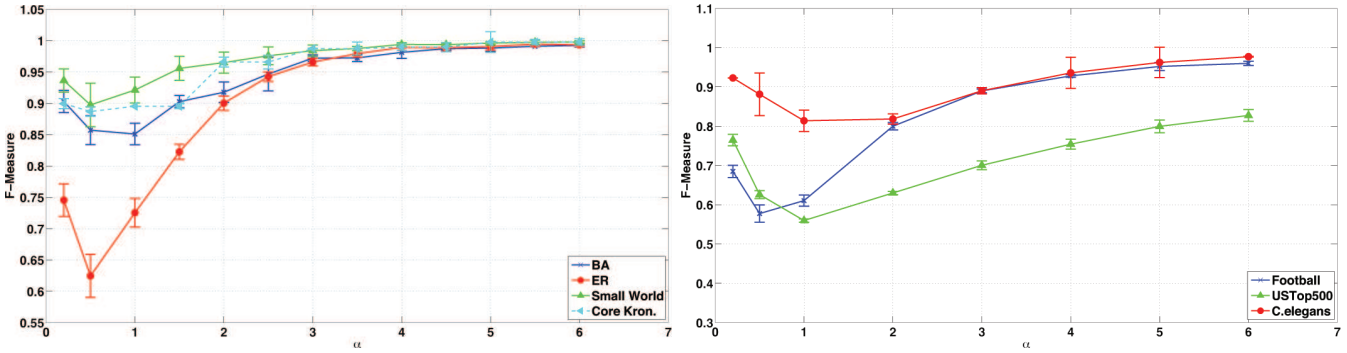
We consider both synthetic and real networks.

1) *Synthetic Networks:*
   We use three well-known classic models for generating directed networks, namely, the Erdos-Renyi model [43] with 500 edges, the Small world graph model [44] with each node being connected to 4 nearest neighbors in ring topology with the rewiring probability of 0.4, and the Barabasi-Albert model [42] with each new node getting connected to 5 existing nodes. Each of these networks have 100 nodes. Also, we use a variant of Kronecker graph model [45], namely, Core-Periphery Kronecker [46] with 256 nodes which results in about 600 edges in several generations.
2) *Real Networks:* We consider three directed real-world networks. First, we consider the network of American football games between Division IA colleges during regular season for Fall 2000 [47] which includes 115 nodes and 615 edges. Second, we consider the neural network of the Caenorhabditis elegans worm (C.elegans) [44] with 306 nodes and 2345 edges, and third, we use the network of 500 busiest commercial airports in the United States [48] with 500 nodes and 2980 edges.

B. Settings

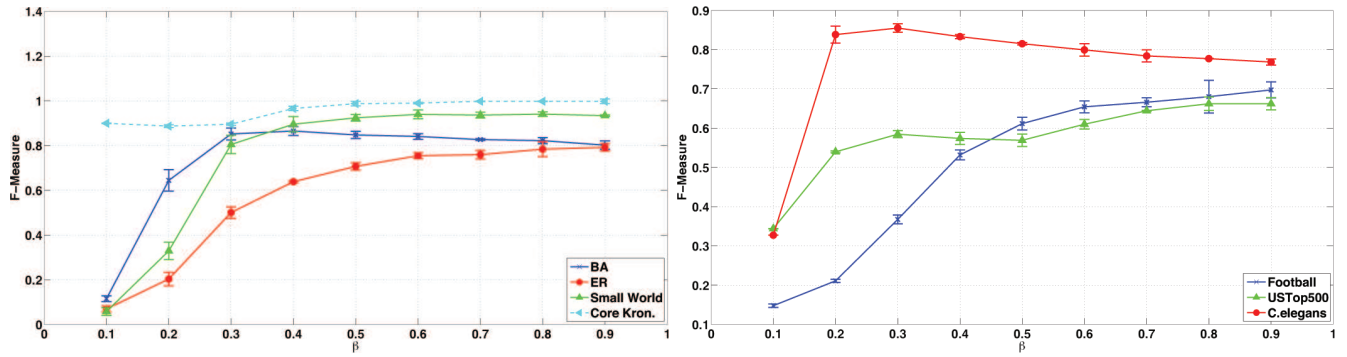
In each of the test cases for the synthetic networks, we generated 100 networks with corresponding set of cascades. For the real-world datasets, we only generated 100 sets of cascades.

To evaluate the accuracy of the proposed framework, we can measure the precision and recall of our method. Precision refers to the number of correctly inferred diffusion links divided by the total number of inferred diffusion links, and recall refers to the number of correctly diffusion links divided by the total number of links in the network. To consider the effects of both precision and recall simultaneously, we used



(a) Synthetic networks (b) Real networks

Fig. 4. α parameter analysis using CS-NetRec - It can be seen that the value $\alpha = 3$ is where the F-measures become cosistent.



(a) Synthetic networks (b) Real networks

Fig. 5. β parameter analysis using CS-NetRec - The value of $\beta = 0.5$ is where the F-measures become cosistent.

the F-measure metric. This metric presents the harmonic mean of both precision and recall:

$$F = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

We use Basis Pursuit to solve our optimization problem. After applying CS, we set a threshold of 0.5 for the predicted values of the vector x. We tested other values of threshold and observed no significant effect in the accuracy of our approach. Thus, we consider all elements above 0.5 as a predicted edge.

C. Parameter Analysis

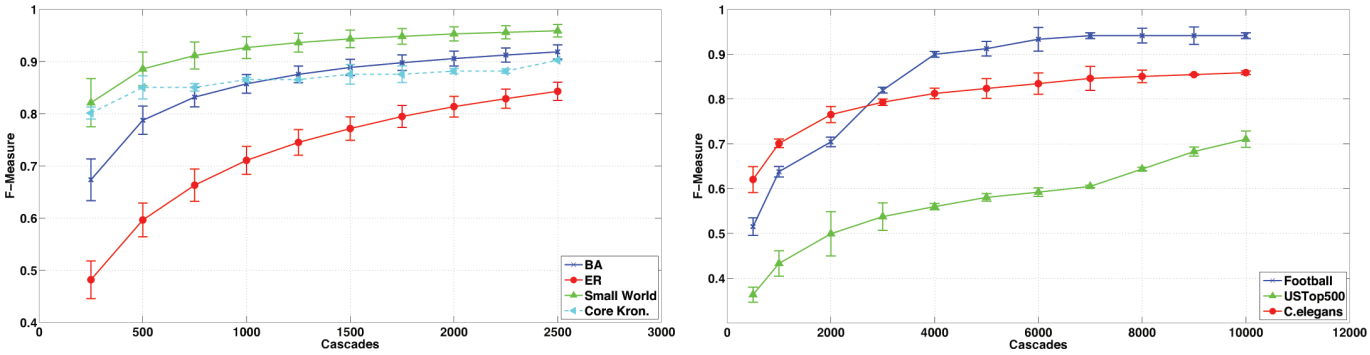
We now want to analyze the sensitivity of our approach to the parameters of the cascades and if possible, fix them so that we get the rest of the results based on the fixed values of the parameters. The information cascades have two parameters to diffuse throughout the network.

First, we show the sensitivity of the results to α which represents the cascades' speed. To this end, we consider the cascades with a constant β equal to 0.5. As shown in the Figure 4, it seems for all networks, the F-measures reach a consistency when the value of α is around 3. Hence we fix this value for α. For the small values of α, i.e. near zero values, the probability of each edge transmission (Eq. (6)) will get closer to zero. This causes more truncated values in the matrix A and also in calculation of $LP(c|G)$. Thus, we expect a lower accuracy for small values of α as shown in the Figure 4. For very large values of α, the probability of each edge transmission approaches to 1, and thus its log becomes equals to zero which results in trivial equations in the linear system ($0 = 0$) and we won't be able to obtain any information from the cascade data.

The length of the cascades, controlled by the parameter β, and can significantly alter the results of any network reconstruction approach. We consider the cascades with a constant α equal to 1. Obviously, when one runs a number of cascades on a network with a small value for β, the cascades will be shorter. Thus, a smaller number of edges will be observed in such situations, and as a consequence, the resulted recall will be very low. The same phenomenon can be observed in our framework in Figure 5, although the F-measure in BA and Small world are effected less from the small recall than the others. Increasing the value of β will make the cascades longer. This causes the matrix A to have less zero variables which results in a longer, and in some cases like BA model, less accurate performance. As seen in Figure 5, in most of the networks, all the measures become consistent when reached to the value of 0.5 for β. Thus we fix β at this value.

D. Cascade dependency:

Obviously, it is almost never possible to have equal number of processes to the number of possible edges in a network as its computationally expensive. Also, increasing the number of



(a) Synthetic networks (b) Real networks

Fig. 6. Different networks cascade dependency using CS-NetRec

transmitting cascades over the network leads to more accurate prediction of the network topology. We tested this scenario by running 250 to 2500 cascades on each of the synthetic networks and the Football network, where there are around 10000 possible edges in the graph. On other real datasets, we ran 250 to 10000 cascades. In C.elegans dataset there are about 93000 possible edges and in US Top 500 dataset, this number reaches to around 250000. In all of our measurements we consider the resulting F-measures.

In Figure 6, particularly for Small world and BA datasets which are closer to real-world graphs than ER model, it can be observed that even on low number of cascades (e.g. half of the number of existing edges in the graphs), we have high values F-measure, almost all above 0.5, considering around 10000 possible edges to be predicted. Thus the results demonstrate that our framework can work accurately even on very low information from the network of interest. In the Kronecker dataset and Small world, we have above 0.8 F-measure when the number of cascades are roughly the same as the number of edges in the graph.

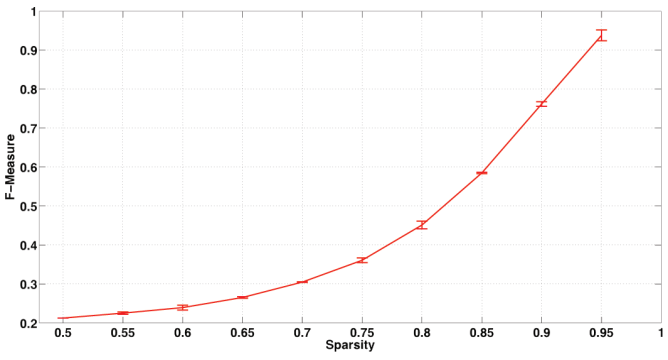


Fig. 7. The effect of the sparsity of the network adjacency in ER model using CS-NetRec. the less sparsity in the unknown vector, the less accuracy in the reconstruction of the network.

In real datasets, we have the same performance in low number of cascades as in the synthetic datasets. In C.elegans and US Top 500, with 2000 cascades we get around 0.75 and 0.5 F-measure while there are around 93000 and 250000 possible edges to predict in each of these datasets respectively. Further, we observe that the F-measure in Football dataset, due to the lower number of vertices, reaches the 0.9 F-measure by 4000 cascades. Thus, we observe that in spite of the same scale of this dataset to the synthetic ones, we need more number of measurements to reach a high F-measure value.

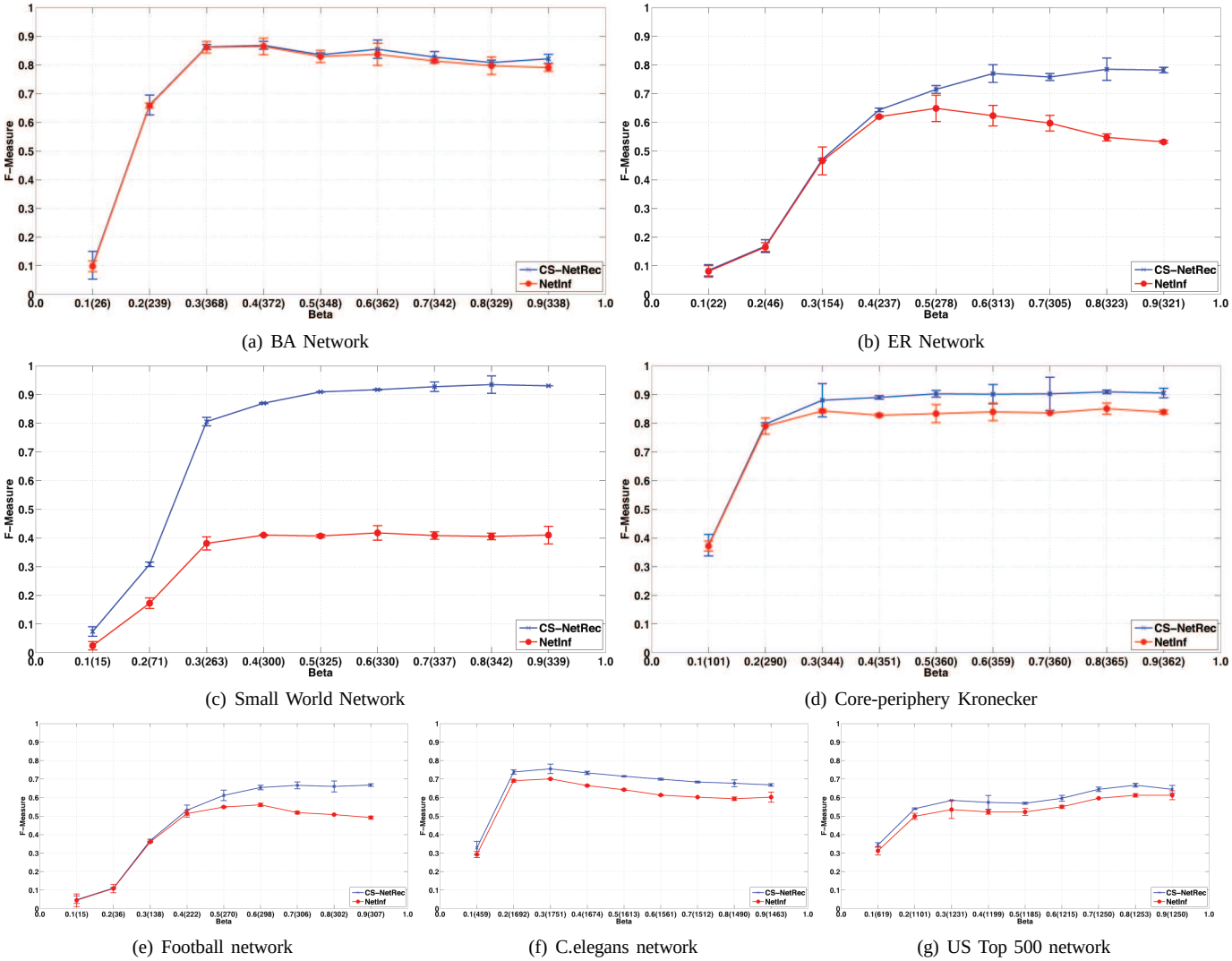
The reason behind better results for F-measure in higher number of cascades that can be presented in several aspects. First, with more cascades we traverse more edges of the graph. Meanwhile, the transmission probability over an edge gets repeated in more equations of the linear system. Thus, the effect of that edge on the increase of cascade probabilities can be seen more and the predicted value for the corresponding element in the vectorized adjacency matrix will get a higher value (i.e. passes the considered prediction threshold). Furthermore, in CS theory, higher number of measurements (i.e. rows of the matrix A) most probably results in better recovery which can be observed in this result.

E. The effect of the sparsity of the network adjacency:

According to the CS theory, the less sparsity in the unknown vector, the less accuracy in its recovery. To show this phenomenon in our framework, we consider several ER networks with 100 vertices and number of edges ranging from 500 to 5000. In other words, the sparsity ranges from 0.5 to 0.95. Again we consider all the F-measures to evaluate the results.

As expected, in Figure 7, we observe that the more sparsity in the same scale (i.e. number of vertices) yields to the better performance of our approach. In 0.95 sparsity, we have a near perfect reconstruction. However, decreasing the sparsity to 0.9, diminishes the F-measure to below 0.8. Thus, in the same dataset, the network reconstruction with more sparsity can result in higher accuracy.

The sparsity can not be compared in different datasets. To see this, we can consider the results for C.elegans and US Top 500 in Figure 6(a) again. Both datasets have above 0.97 sparsity in their vectorized adjacency matrix. But due to the larger scale of US Top 500, and the more cascades needed for the better reconstruction, the F-measure related to C.elegans is relatively higher than US Top 500 with the same number of cascades.



(a) BA Network

(b) ER Network

(c) Small World Network

(d) Core-periphery Kronecker

(e) Football network

(f) C.elegans network

(g) US Top 500 network

Fig. 8. Performance comparison of CS-NetRec with NetInf, considering different values of β parameter (i.e. cascade lengths) and $\alpha = 1$. (a,b,c,d) Sythetic networks (e,f,g) Real networks. We considered 2500 cascades for BA, ER, SmallWorld and Football datasets, 8000 cascades for Kronecker graph and 10000 cascades for C.elegans and US Top 500. The numbers in the parantheses show the number of edges recovered.

F. Performance comparison with NetInf:

As we present our proposed framework in the context of information networks, we compared the performance of our method with NetInf; one of the state-of-the-art methods for inferring networks of diffusion [5].

We consider different values for the parameter β as we would like to evaluate the performance of methods in presence of more local information (i.e. shorter cascades) and more global information (i.e. longer cascades). We also set α to 1 in our tests. Given a fixed number of cascades, we ran NetInf with the number of iterations same as the number of edges that our method can reconstruct. The results can be seen in Figure 8. It can be observed that in all test cases, the resulted F-measure in our method is higher than or equal to NetInf and as we increase the value of β, the difference in F-measure becomes more. In Small world, ER, and Football datasets, the difference is more clear. In $\beta = 0.9$ the improvement by our method in F-measure is about 0.5, 0.25 and 0.18 for Small world, ER and Football datasets respectively. Overall, for different cascade lengths, our method outperforms NetInf by an average 9-10% improvement in terms of F-measure.

VI. Conclusion

In this paper, we studied the problem of network reconstruction and introduced a general framework based on which any type of the observations on an external process on the network can be utilized to reconstruct the network of interest. As a special case, we considered the diffusion of the information cascades as an external process on the underlying network. By utilizing the probabilistic measures of information cascades, we formulated the problem of network reconstruction as a linear system in the context of compressive sensing. Since most of the time, this linear system is under-determined, we used the theory of compressive sensing as a tool to reconstruct the network of interest. By numerical experiments,

we demonstrated that this framework will converge to accurate solutions. In addition, the results suggest that in the context of information networks, our method can perform even better than the state-of-the-art method, NetInf.

Several directions can be pursued for the future work. Here we used the adjacency matrix as the unknown vector and thus it is interesting to look for ways to reduce the dimensionality of the linear system. Also it would be interesting to use other external processes and utilize other features for the network nodes to reconstruct the network of interest using the proposed framework.

REFERENCES

[1] C. von Mering, R. Krause, B. Snel, M. Cornell, SG. Oliver, S. Fields and P.Bork, *Comparative assessment of large-scale data sets of protein-protein interactions*, Nature 417:399403, 2002.

[2] JL. Schafer and JW. Graham, *Missing data: Our view of the state-of-the-art*, Psychol Methods 7:147177, 2002.

[3] CT. Butts, *Network inference, error, and informant (in)accuracy: A Bayesian approach*, Soc Networks 25:103140, 2003.

[4] G. Kossinets, *Effects of missing data in social networks*, Soc Networks 28:247268, 2006.

[5] M. Gomez-Rodriguez, J. Leskovec and A. Krause, *Inferring networks of diffusion and influence*, In proc. of KDD 10, pages 1019-1028, 2010.

[6] R. Guimera and M. Sales-Pardo, *Missing and spurious interactions and the reconstruction of complex networks*, PNAS, 106(52), 2009.

[7] A. Clauset, C. Moore, and M. E. J. Newman, *Hierarchical structure and the prediction of missing links in networks*, Nature, 453(7191):98101, May 2008.

[8] K. Bleakley, G. Biau, and J. Vert, *Supervised reconstruction of biological networks with local models*, Bioinformatics, 23(13):i57i65, 2007.

[9] Y. Yamanishi, J. Vert, and M. Kanehisa. *Protein network inference from multiple genomic data: a supervised approach*, Bioinformatics, 20:i363i370, 2004.

[10] E. J. Candes and T. Tao, *Decoding by linear programming*, IEEE Trans. on Information Theory, 51(12), pp. 4203 - 4215, December 2005.

[11] E. J. Candes, *Compressive sampling*, Proc. International Congress of Mathematics, 3, pp. 1433-1452, Madrid, Spain, 2006.

[12] D. Donoho, *High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension*, Discrete and Computational Geometry , 102(27), pp. 617-652, 2006.

[13] M. Kim and J. Leskovec, *Network Completion Problem: Inferring Missing Nodes and Edges in Networks*, SIAM International Conference on Data Mining (SDM), 2011.

[14] S. Hanneke and E. P. Xing, *Network Completion and Survey Sampling*, Journal of Machine Learning Research - Proceedings Track 5: 209-215, 2009.

[15] E. J. Candes and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math., 9(6):717772, 2009.

[16] D. Liben-Nowell and J. Kleinberg, *The link prediction problem for social networks*, In CIKM 03, pages 556 559, 2003.

[17] D. Goldberg and F. Roth, *Assessing experimentally derived interactions in a small world*, Proceedings of the National Academy of Sciences, 100(8):4372, 2003.

[18] M. Eslami, H. R. Rabiee and M. Salehi, *DNE: A Method for Extracting Cascaded Diffusion Networks from Social Networks*, IEEE Social Computing Proceedings, 2011.

[19] D. Gruhl, R. Guha, D. Liben-Nowell and A. Tomkins, *Information diffusion through blogspace*, In proc. of of the 13th international conference on World Wide Web, pages 491501, 2004.

[20] G. Kossinets, J. M. Kleinberg and D.J. Watts, *The structure of information pathways in a social communication network*, KDD 08, pages 435-443. 2008.

[21] E. J. Candes, J. Romberg, and T. Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Comm. Pure Appl. Math., 59(8):12071223, 2006.

[22] D. L. Donoho. Compressed sensing. IEEE Trans. Inform. Theory, 52(4):12891306, 2006.

[23] E. J. Candes, *The restricted isometry property and its implications for compressed sensing*, Compte Rendus de lAcademie des Sciences, 346:589592, 2008.

[24] J. Tropp and A. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, IEEE Trans. on Information Theory, 53(12):46554666, December 2007.

[25] A. C. Sankaranarayanan, P. K. Turaga, R. G. Baraniuk and R. Chellappa, *Compressive Acquisition of Dynamic Scenes*, European Conference on Computer Vision, Crete, Greece, September 2010.

[26] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly and R. Baraniuk, *Single-pixel imaging via compressive sampling*, IEEE Signal Processing Magazine, 25(2), pp. 83 - 91, March 2008.

[27] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly and R. Baraniuk, *An architecture for compressive imaging*, Int. Conf. on Image Processing (ICIP), Atlanta, Georgia, October 2006.

[28] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, *Compressive wireless sensing* in IPSN, April 2006.

[29] J. Meng, H. Li, and Z. Han, *Sparse event detection in wireless sensor networks using compressive sensing*, in 43rd Annual Conference on Information Sciences and Systems (CISS), pp. 182185, 2009.

[30] J. Haupt, W. Bajwa, M. Rabbat, and R. Nowak, *Compressed sensing for networked data*, IEEE Signal Processing Magazine, vol. 25, no. 2, pp. 92101, March 2008.

[31] W. Xu, E. Mallada and A. K. Tang, *Compressive Sensing over Graphs*, accepted to IEEE InternationalConference on Computer Communications (INFOCOM), 2010.

[32] R. Gaeta, M. Grangetto, M. Sereno, *Local Access to Sparse and Large Global Information in P2P Networks: A Case for Compressive Sensing*, 2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P), vol., no., pp.1-10, 25-27, Aug. 2010.

[33] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar and A. Kabbani, *Counter braids: a novel counter architecture for per-flow measurement*, in ACM SIGMETRICS 2008, Jun. 2008.

[34] Y. Zhang, M. Roughan, W. Willinger and L. Qiu, *Spatio-temporal compressive sensing and internet traffic matrices*, in ACM SIGCOMM 2009, Aug. 2009.

[35] H. Lee, D. S. Lee, H. Kang, B.-N. Kim and M. K. Chung, *Sparse Brain Network Recovery Under Compressed Sensing*, IEEE Transactions on Medical Imaging 30(5): 1154-1165, 2011.

[36] W.-X. Wang, Y.-C. Lai, C. Grebogi, and J.-P. Ye, Network Reconstruction Based on Evolutionary-Game Data, Physical Review X 1, 021021, 1-7, 2011.

[37] D. Kempe, J. Kleinberg and E. Tardos, *Maximizing the spread of influence through a social network*, KDD 03: Proc. of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, pages 137-146, 2003.

[38] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance and M. Hurst, *Patterns of Cascading Behavior in Large Blog Graphs*, In proc. of SDM07, 2007.

[39] J. Edmonds, *Optimum branchings*, J. Res. Nat Bureau Stand. 71B, 233240, 1967.

[40] R. Tibshirani, *Regression shrinkage and selection via the LASSO*, Journal of the Royal Statistical Society B, vol. 58, pp. 267288, 1996.

[41] E. J. Candes, M. Rudelson, T. Tao, and R. Vershynin, *Error correction via linear programming*, pp. 295308, 2005.

[42] A.L. Barabasi and R. Albert, *Emregence of scaling in random networks*, Science, 1999.

[43] P. Erdos and A. Renyi, *On the evolution of random graphs*, Publication of the Mathematical Institute of the Hungarian Acadamy of Science 5, 1761, 1960.

[44] D. J. Watts and S. H. Strogatz, *Collective dynamics of small-world networks*, Nature 393, 440442, 1998.

[45] J. Leskovec and C. Faloutsos, *Scalable modeling of real graphs using Kronecker multiplication*, ICML,ACM International Conference Proceeding Series, 227: pages 497-504, 2007.

[46] J. Leskovec, K.J. Lang, A. Dasgupta and M.W. Mahoney, *Statistical properties of community structure in large social and information networks*, WWW, pages 695-704, 2008.

[47] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, Proc. Natl. Acad. Sci. USA 99, pages 7821-7826, 2002.

[48] V. Colizza , R. Pastor-Satorras and A. Vespignani, *Reaction-diffusion processes and metapopulation models in heterogeneous networks*, Nature Physics 3, 276-282, 2007.