Network Diagnosis from Information Spread

Yuli Baryshnikov  Peter Winkler*

June 16, 2004

Abstract

We describe a scheme for finding clusters in a network by observing via a small set of entry nodes how messages, and versions of messages, spread. The technique involves linear algebra and relies on a novel model of the effect of information propagation.

1 Introduction

Suppose that inferences are desired regarding the organization of some communications network, but that the network topology is unknown and information is obtainable only by observing which of some fixed set of accessible nodes receives a message.

Over the course of observing many messages, possibly instigated by the information-seeker, one might expect to find that some of the nodes behave as if they were closely connected in the network. Identifying clusters of nodes may enable the seeker to infer connections, to predict the behavior of future messages, and/or to disseminate future information efficiently.

As our prime example we take the network to be the Internet (or some part of it), with communication by electronic mail. Here timing information (as considered, for example, in [7, 15]) is of little use because the time between reception and passing on of a message is highly variable, depending as it does on user habits and time of day. When a news item, rumor or joke spreads in the Internet, however, there is much potential information to be gleaned by observing which of some set of accessible nodes have been reached.

In the Internet and many other communications networks, the seeker has (through his accessible nodes, or “agents”) the opportunity to subtly alter messages, forwarding a different version, for example, to each of an agent’s contactees. In email, such alterations can take the form of minor changes in text or heading. When messages arrivals are later detected by other agents, their versions evidently provide additional information to the seeker; hence whatever means she uses to surveil the network should be able to take advantage of such information. As we shall see, the keeping track of versions can measurably improve our diagnostics.

There have been numerous studies of actual communications networks based on traffic analysis of all messages passed in a given time period, e.g. numbers of messages passed from i to j for every

*Mathematical Sciences Research Center, Bell Labs, Lucent Technologies. Research supported by NMIMT/ICASA grant MDA904-01-C-0327.
adjacent pair \((i, j)\) of nodes in a network; see e.g. \([1, 2, 14]\). Usually either the topology is known or it is ignored, treating the network as complete; in many cases much is known about the users, for example their level in a company or role in a high school. Thus there may be natural clusters whose sociological significance can be tested.

Here we do not assume access to all nodes or all messages, nor to any \textit{a priori} knowledge about individual nodes; our goals are correspondingly more modest. We simply wish to make some reasonable statistical inferences about the behavior of the nodes we can access.

Moreover, we anticipate that information will be gathered and analyzed in real time, while patterns are (slowly) changing. Therefore it is of importance to us that useful information be retained in some efficient data structure, enabling continual updates.

We will argue below that it is unreasonable to expect to reconstruct the topology of a network when we only have data about which messages are received by which agent. However, it may be possible to deduce the existence of “clusters,” that is, groups of agents who are in close contact with one another.

Our ultimate goal is an automated system which maintains a data structure, updating it as information about the spread of particular messages is received. The system should be able at any time both to suggest and test hypotheses for clusters among agents.

We do not, of course, construct such a system in detail in this paper, nor do we test a prototype on real data. We do, however, suggest a mathematical model for the network and several approaches for extracting the parameters of this model, one based on very recent research involving semi-definite programming. The approaches are described and tested on toy constructions.

\section{Agents and Instances}

We will assume throughout that there is a fixed set \(U\) of \(n\) \textit{agents} which participate as nodes in some communications network; it is possible that \(U\) consists of all nodes in the network, but in general we imagine that they are a relatively small subset.

When message \(m\) is introduced into the network, either by an agent or by some other node, we will have the opportunity to observe which agents receive the message. The result is a random vector \(X = (X_1, \ldots, X_n) \in \{-1, 1\}^n\) where \(X_j = 1\) if agent \(j\) got (or originally sent) \(m\), and -1 otherwise. (Later, it will sometimes be convenient to use \(\{0, 1\}\)-valued vectors instead, where “1” means that a message, or particular version of a message, has been received by a particular agent.)

The vector \(X\) constitutes one \textit{instance}, which we associate with a time \(t\). For most of what follows we will assume that \(m\) instances \(X(1), \ldots, X(m)\) have been observed, resulting in a binary \textit{incidence matrix} \(A = \{a_{ij}\}\) with \(a_{ij} = X_j(i)\). It is from this matrix that we wish to infer network structure.

\section{De Finetti’s Theorem}

In our modeling, we rely on the following corollary (or reformulation) of de Finetti’s Theorem \([9]\) on infinite sequences of exchangeable random variables. A sequence of random variables is said to be \textit{exchangeable} if its joint distribution is invariant under any finite permutation of its indices.

Assume that a finite set \(g\), with \(|g| = t\) is fixed.

\begin{itemize}
\item \(X_1, \ldots, X_t\) are exchangeable.
\item \(X_1, \ldots, X_t\) are independent.
\item \(X_1, \ldots, X_t\) are identically distributed.
\end{itemize}
Theorem 3.1. Let \((X_1^1, X_2^1, \ldots), (X_1^2, X_2^2, \ldots), \ldots, (X_1^k, X_2^k, \ldots)\) be sequences of exchangeable random variables with values in \(V = \mathbb{R}^d\). Then there exists a random variable \(\mu\) on the (Polish) space \(\mathcal{M}\) of probability measures on \(V^\mathbb{N}\) such that conditioned on \(\mu\), \(X^\tau = (X_1^\tau, X_2^\tau, \ldots)\) are i.i.d. random variables drawn from a distribution \(F^{\tau}_{\mu}\), and the \(X^\tau\)'s, for \(\tau \in \mathcal{G}\), are independent.

Remark

One can extend this theorem to finite but large populations (see, e.g., [3, 10]), insofar one is concerned with correlation functions of small order. Specifically, the marginal distributions of outcomes for any tuple of users converge, as the population sizes go to infinity, to a mixture of Bernoulli samples.

In our situation, the set \(\mathcal{U}\) of agents is split into groups \(T_1, \ldots, T_T\), and within each group, the agents are indistinguishable for an outside observer (trying to infer the network structure). Hence, it is natural to represent each user in group \(T_\tau\) by a random variable (no matter which, by definition) from the exchangeable sequence \((X_1^\tau, X_2^\tau, \ldots)\). We will denote the group to which the user \(j\) belongs by \(T_{\tau(j)}\).

Examples

- **Billboards:** Here, the groups of users are identified by the information the users have access to; given this access, whether or not a user gets the message is independent of the rest. This is exactly a mixture of Bernoulli distributions.

- **Communities:** Here, the ease of communication between users depends only on the community to which the user belongs. This can be modeled as bond percolation (see, e.g., [13]). In other words, one assumes that there is some specific, though unknown, topology and each edge will operate with some independent probability to add one of its ends to the set of nodes which have “gotten the word.”

More generally, the bond percolation model can be defined using the probability matrix

\[
P_{\tau_1, \tau_2, \tau_1, \tau_2} \in \mathcal{G}:
\]

a user in \(\tau_1\) sends a news to a user in \(\tau_2\) with this probability.

4 The Percolation Model

It is evident that whether agent \(j\) gets the \(i\)-th message depends on many factors, including the intrinsic “power” of the message to induce forwarding, the degree to which \(j\) is central in the network, and the degree to which the message penetrated the cluster or clusters to which \(j\) belongs; plus, of course, sheer random luck. In particular, \(X_j^1\) and \(X_k^1\) may be highly (positively) correlated for some pairs \([j, k]\) of agents, and some mechanism for producing this correlation must be included in our model.

One difficulty with the model is that it is highly unrealistic to assume that all messages have the same likelihood of being propagated; surely, some messages will have greater interest than others.

A more serious difficulty is that even without varying message strengths, there is simply not enough information available from the matrix \(A\) to recover network topology.
Consider, for example, the two networks pictured in Figure 1, each involving 3 agents. In the “Δ” network, each edge transmits independently with probability \( p \); in the “Y” network, with probability \( q \).

Suppose we take \( p \) to be the root of the polynomial \( p^3 - 3p + 1 \) between 0 and 1 — about 0.34729636 — and \( q = 1 - p \). Begin messages at (say) agent 1 of each network. In the Δ network, the probability that agent 2 receives the message is

\[
p + (1-p)p^2 = p + p^2 - p^3 = p + p^2 - (3p - 1) = (1 - p)^2 = q^2
\]

which is precisely the probability that agent 2 gets the message in the Y network. The probability that both agents 2 and 3 get the message in the Δ network is

\[
p^3 + 3p^2(1-p) = -p^3 + 3p^2 - p^3 = -p^3 + 3p^2 - (3p - 1) = (1 - p)^3 = q^3
\]

which, again, matches the result in the Y network. In view of these equalities and their symmetric analogues, it is clear that the two topologies cannot be distinguished by observing message probabilities.

In general, the number of degrees of freedom for a topology with edge-capacities is \( \binom{N}{2} \), where \( N \) is the total number of nodes, while a probability for each possible value of \( X \) gives us \( 2^N \) degrees of freedom; so even absurdly accurate measurements of message probabilities, requiring exponentially many observations, cannot hope to recover the network topology when \( n < N - \frac{1}{2} \log_2 N \). In other words, there is no hope of recovering the parameters of the percolation model unless nearly all of the nodes of the network are agents, and probably not even then.

5 The “Types” Model

We have seen that it is unreasonable to expect to be able to reconstruct a network from an incidence matrix of messages. However, the percolation model does suggest some behavior which is confirmed by experience and can perhaps be duplicated in a simpler model. Everyone has observed a tendency within a web community for messages either to reach only a very limited audience, or to spread like a wildfire to almost every member of the community. This can be attributed to a manifestation of phase transition in the bond percolation model, where there is a rather sharp probability threshold separating the two behaviors.

A message which spreads rapidly in a given community (in particular, among a given set \( \mathbb{U} \) of agents) will be said to “percolate.” We will represent this phenomenon by a random variable \( Z \)
associated with message strength. The idea is that the reception of a message by an agent will depend on $Z$ and certain characteristics of the agent; those agents who react in a particular way to $Z$ will constitute a type $\tau$ and will exhibit a positive correlation when compared with one another.

If we can identify these correlations, we can in theory assign the agents to a small number of “types” $1, \ldots, t$. The set of agents of type $\tau$ will be denoted $T_\tau$, so that $T_1, \ldots, T_t$ constitute a partition of $U$. We will infer that types whose members exhibit highest positive correlation are tightly connected in the network; other types may constitute looser clusters or simply all agents not in a cluster. We designate by $\tau(j)$ the type of agent $j$, so that by definition $j \in T_{\tau(j)}$. The number $t$ of types is always assumed to be small.

Let $Z(i)$ be the instance of $Z$ associated with message $i$. We assume that given the value of $Z(i)$, every agent behaves independently with regard to whether or not it receives message $i$; thus, all correlation is a result of not fixing the value of $Z(i)$.

We define a profile $\pi$ to be a function from $Z$ to the unit interval $[0,1]$; the idea is that each type $T_\tau$ of agent has an associated profile $\pi_\tau$, and $\pi_{\tau(i)}$ is the probability that a given agent in $T_\tau$ receives message $i$, given that $Z(i) = z$.

The importance of distinguishing two types rests on the observation that types can be subdivided recursively if the quantity of message data warrants. Thus our experiments, as well as some of our theory, will be centered on the two-type case.

## 6 First Order Correlations: The Frequency Vector

The first idea in approaching the problem of separating the groups of agents is to do it based on the frequencies with which the users receive the information. These frequencies, by virtue of Law of Large Numbers, should converge, as the number of trials increases, to some limits, and the deviations of the experimental values from these limits can be estimated using the Central Limit Theorem. Hence, one can distinguish the groups of agents by observing to which of the different limits the frequencies of the observations converge for a particular agent.

This approach has an obvious drawback: if the limits of the frequencies are close, there is no way to distinguish between them. This drawback is characteristic of any method relying on the first order characteristics not involving the mutual interdependence of observations by different users. To overcome this, it is necessary to rely on higher order correlation functions. The simplest such mechanism turns out to be quite efficient, and is given by spectral methods, which we describe below.

## 7 The Correlation Matrix

Our attempt at recovering the types of agents will rest on the pairwise correlations of the agents (and/or the messages). Let us assume the conditions of Section 3, in particular the heuristic assumption that the 2-correlations between different users behave like those of Bernoulli mixtures. Specifically, we can postulate that for for $X_1, X_2$, the inner product $X_1 \cdot X_2$ represents the degree of
proximity between these two observations\textsuperscript{1}. Now, given a series of “measurements”
\[ X(1), X(2), \ldots, X(m), \]
where \( X = (X^\tau)_{\tau \in B} \), one can form the correlation matrix
\[ C = (C_{ij}), \quad C_{ij} = \frac{1}{m} \sum_t X_i(t) \cdot X_j(t). \]

Also, define
\[ E = (E_{ij}), \quad E_{ij} = \mathbb{E}C_{ij}. \]

Define
\[ e^\tau(\mu) = \mathbb{E}(X_i^\tau|\mu) \]
to be the expected value of \( X^\tau \) conditioned on \( \mu \).

Then, one has
\[ E_{jk} = e(\tau(j), \tau(k)) := \mathbb{E}_\mu e^{\tau(j)}(\mu) \cdot e^{\tau(k)}(\mu) \]
if \( j \neq k \) and
\[ E_{jj} = e(\tau(j)) := \mathbb{E}_\mu \mathbb{E}(|X_i|^2|\mu). \]

**Example**

Assume that \( X = 1 \) or 0, depending on whether or not a message is seen. In this case, \( e^{\tau}(\mu) = P(X = 1|\mu) \), and \( |X|^2 = X \).

For two distinct agents \( j \) and \( k \), we have
\[ E_{jk} = E_{X_j}X_k = \int e^{\tau(j)}(\mu)e^{\tau(k)}(\mu)P(d\mu), \quad (1) \]
and
\[ E_{jj} = \int e^{\tau(j)}(\mu)P(d\mu). \quad (2) \]

A simple but important example is a network in which the agents have just two types, 1 and 2, and \( \mu \) takes just two values, say \( \mu_1 \) and \( \mu_2 \). We might suppose, for instance, that the agents of type 1 constitute a cluster in which the message is likely either to percolate throughout or avoid almost entirely. Thus, we might have:
\[ e^1(\mu_1) = .9; e^1(\mu_2) = .1 \]
while
\[ e^1(\mu_1) = .6; e^1(\mu_2) = .4. \]

In this case, if \( \text{Pr}(Z = \mu_1) = 0.5 \), we would have \( \mathbb{E}X_j = 0.5 \) for all agents \( j \), but the matrix \( E \) would have nontrivial structure: the diagonal of the matrix will be identically
\[ E_{jj} = .5, \]

\textsuperscript{1}if not, one can always try to change the representation accordingly; the issue of possibility to represent “similarity” with inner products (Euclidean distances) deserves attention and has been discussed in the literature, but won’t be here.
but otherwise

\[
E_{jk} = \begin{cases} 
.41 & \text{if } \tau(j) = \tau(k) = 1 \text{ and } j \neq k, \\
.29 & \text{if } \tau(j) \neq \tau(k), \\
.26 & \text{if } \tau(j) = \tau(k) = 2 \text{ and } j \neq k.
\end{cases}
\]

We remark that the matrix \( E \) is a multiple of a diagonal matrix and a matrix of rank 2. More generally, if all \( e^{\tau} \)'s are identically equal to \( e \), the expected matrix of correlations is sum of \( e \) times the unity matrix and a matrix of rank equal to the number of types of agents. Thus, we might reasonably expect analysis of eigenvectors of \( A^\top A \) to help us identify the clusters \( T_1, \ldots, T_t \).

8 The Spectral Approach

The problem of recovering connections from pairwise correlations arises in many settings. For example, in [16], the objective is to determine which SNPs (Single Nucleotide Polymorphisms) in the human genome are associated with a given disease. The SNPs constitute the columns of a binary matrix, like our message incidence matrix, but the rows are already partitioned according to whether the subject exhibits the disease. Our problem would be akin to finding associations among SNPs in an unknown population.

Heuristic spectral methods are often employed in such statistical problems and in this section, we suggest such an approach for our problem of finding a good partition of agents. Since, as previously mentioned, we can partition recursively, we concentrate below on the \( t = 2 \) case.

Let us concentrate on the spectral decomposition of the sample correlation matrix \( C \). First of all, we notice that for large \( m \), the coefficients of this random matrix are (in \( L^2 \) norm, at least) close to their expected values \( E_{jk} \), that is, \( \mathbb{E}R_{jk}^2 = O(1/m) \), where \( R_{jk} = C_{jk} - E_{jk} \).

We will not go into the details of the convergence of spectral decomposition of the perturbed matrix \( C = E + R \) to that of the unperturbed matrix \( E \), concentrating here only on the recovery of the clustering of agents from the spectral decomposition of \( E \).

This can be done as follows. Recall that the users are split into groups, and, according to (1) and (2), all diagonal elements are identical for indices varying within a group, and the same is valid for the entries without the diagonal. Hence, it is immediate to check the following.

**Proposition 8.1.** For any group \( \tau \), let the vector

\[
v_\tau = \sum_{j: \tau(j) = \tau} v_j,
\]

be the sum of all vectors in \( V \) with indices in \( \tau \). Then

1. The subspace \( V_\tau \) of \( V \) generated by \( v_\tau \)'s is an invariant subspace for \( E \), and the restriction \( E_\tau \) of the quadratic form defined by \( E \) to this subspace in the basis \( (v_\tau) \) is given by the matrix

\[
\bar{E} = (E_\tau)_{\tau\tau'} = \begin{cases} 
   n_\tau e_\tau + n_\tau(n_\tau - 1)e(\tau, \tau) & \text{if } \tau = \tau', \\
   n_\tau n_{\tau'} e(\tau, \tau') & \text{if } \tau \neq \tau',
\end{cases}
\]

where \( n_\tau = |v_\tau|^2 \) is the size of the cluster \( \tau \).
2. The complementary subspace is generated by vectors $e_i - e_{i+1}$, $\tau(j) = \tau(j+1)$. Moreover, the spectral radius of $E$ restricted to $V_{\tau'}^c$ is bounded, as the $n_{\tau}$ increase to infinity.

These results imply that if all cluster sizes $n_{\tau}$ grow in concert, say as $n_{\tau} \approx f_{\tau} \times n$, where $f_{\tau} > 0$, $\sum_{\tau} f_{\tau} = 1$, then the spectrum of the matrix $E$ will have $t$ (the number of clusters) eigenvalues scaling as $n_{\tau} \lambda_2^2$, and the rest of the eigenvalues bounded. Here $\lambda_2^2$ are the eigenvalues of matrix given in the basis $(v_{\tau})_{\tau}$ by

$$F = (f_{\tau} f'_{\tau} e(\tau, \tau')).$$

In particular, provided that the eigenvalues $\lambda_2^2$ are positive, the highest $t$ eigenvalues of $E$ will correspond to vectors in $V_{\tau'}$, that is, vectors with components constant within a group. Moreover, we can expect, for $m, n$ large, that the components of the eigenvectors of $C$ are also approximately constant within a group. This forms the basis of our spectral recovery procedures.

Consider the essential (for us) case of 2 groups. The matrix $F$ is a $2 \times 2$ symmetric matrix with nonnegative elements. The eigenvector corresponding to its highest eigenvalue has positive coordinates, and, by orthogonality, the second largest eigenvalue for $F$ has components of different signs. Thus, just by clustering the agents according to the sign of the component of the eigenvector of $C$ corresponding to the second largest eigenvalues, we can hope to recover the split of the set of agents.

9 Experimental Results with the Spectral Approach

Here is an example for the Billboard scenario, that is, for the case where, conditioned on $\mu$, the agents values are independent. There are two clusters of agents, of size 100 each, and 2 types of events, A and B. In the event A, an agent in Class 1 sees the message with probability 20%, and an agent in Class 2 with zero probability. For events of type B, the probabilities reverse. We run an experiment with 20 events of types A and B, each.

One would expect, as discussed above, the eigenvectors of the sample correlation matrix to be close to those of their expectation. This is indeed the case: we show below the plots of 200 coordinates of eigenvectors corresponding to the second-largest eigenvalues of matrices C and E, respectively. The agents are grouped according to their clusters (the first 100 coordinates corresponding to the first cluster, the remaining 100 to the second), and one can see clearly how well the eigenvectors separate groups.

We note here that since the incidence matrix has non-negative entries, so does the correlation matrix $C$, whence the largest eigenvalue would necessarily have all positive components (by the Perron-Frobenius Theorem), and the corresponding eigenvector would be a poor candidate to separate agents just by the signs of the components. The second eigenvector, on the contrary, would be forced (by orthogonality to the first one) to have both positive and negative components and would be a natural candidate as a separating vector.

Using this as a guideline, we performed a wide range of experiments within the Billboard scenario. Here are some of the plots (see Figures (??, ??)).

One can see that there is a strong correlation between the quality of prediction given by the second eigenvector and the second eigenvalue: this is understandable, as the coordinates of the second eigenvalue of the sampled matrix $C$ are just noisy perturbations of the second eigenvector of expected value of $C$, $E$ (which has constant coordinates across the groups of agents of the same
type). The amplitude of this perturbation decreases as $1/\lambda_2(E)$, which, again, is close to $1/\lambda_2(C)$. Most important for us is the fact that the spectrum of $C$ can be determined explicitly, and therefore can serve as a proxy for the quality of detection.

Below we show several more pictures (coming from the same sequence of experiments), supporting this observation.

Another way to see the effect is to consider the probability for the Billboard scenario (with identical probabilities of seeing a message for either group) as a function of $\pi_1(A), \pi_2(A)$:

**{0, 1}-Valued Matrices versus {±1}**

The analysis and experiments above were done with $\{0, 1\}$ matrices, yielding non-negative incidence and correlation matrices. If instead we employ $\{±1\}$-valued matrices, both the largest and the second-largest eigenvectors of $\tilde{E}$ can have sign alternations. In this case, it seems that we would be forced to choose between one of the eigenvectors, which in the presence of noise, might be a nontrivial task (perhaps relying on human supervision).

Figure 3: There are 2 groups of agents, of sizes $g_1 = 100$ and $g_2 = 100$. The *a priori* probability of seeing a message is 0.4 for either group, with probabilities $\pi_1(A)$ for a first group agent to see the message given event $A$, respectively 0.1, 0.2 and 0.3 for three graphs above. The probability $\pi_2(A)$ for a second group agent to see the message, given $A$, is the horizontal coordinate. (Given *a priori* probability $p_1 = \pi_1(A)P(A) + \pi_2(B)P(B) = .4$, the probabilities $\pi(B)$ are determined unambiguously). Plotted are the percentage of correctly recognized agents (in red or dark grey) and the second-largest eigenvalue, scaled by 100 (in green or light grey).
Figure 4: Setup is as above, with $g_1 = 100$, $g_2 = 100$, total probability to see a message is 0.4 for either group; $\pi_1(\mathcal{A}) = 0.3, 0.4$ and 0.6 for the a first group agent given $\mathcal{A}$; horizontal axis is again $\pi_2(\mathcal{A})$.

Figure 5: As above, with $g_1 = 100$, $g_2 = 100$, total probability to see a message is 0.8 for either group; $\pi_1(\mathcal{A}) = 0.7, 0.8$ and 0.9 for the a first group agent given $\mathcal{A}$; horizontal axis is again $\pi_2(\mathcal{A})$.

Figure 6: The valley in the plot corresponds to vanishing second eigenvalue in the reduced matrix $\tilde{E}$. The noise overcomes the separating power of the second eigenvector and the quality of the detection deteriorates.

A (partial) remedy is to consider the coordinate-wise products of signs for both first and second eigenvectors. This is illustrated below:
The Semidefinite Programming Approach

The arguments for using the spectral approach are heuristic; in contrast, the following newer approach, advocated in [8], can be used to obtain provably good approximations. However, we must caution the reader that said approximations are (1) based on ideal conditions, e.g. a perfect model; (2) dependent upon expectation over certain perfectly random choices; and (3) not very impressive in the worst case. Thus, we are a long way from being able to prove or cite a theorem guaranteeing the effectiveness of the method below.

Nonetheless, approximation algorithms devised to provide weak guarantees in ideal worst cases have turned out, in many cases, to be quite good in practice. The method suggested in [8] seems well suited to our problem, is mathematically elegant and reasonably efficient. The general idea is to replace a problem which seeks to optimize the values \( \mathbf{n} \) real variables, by one in which the variables have been replaced by unit vectors in \( \mathbf{n} \)-space. The result is a semidefinite programming problem which can be solved in time polynomial in \( \mathbf{n} \); the vectors are then transformed to scalars by projection onto a random normal, perhaps followed by further rounding.

The use of semidefinite programming in this way was pioneered in the theory of computing community by Goemans and Williamson [12], who used it to obtain approximation algorithms with better guaranteed performance than had hitherto been achieved. Very recently, Alon and Naor [5] added to this approach a tool from analysis, Grothendieck’s inequality, and then applied in [8] to “correlation clustering”—which sounds like, and indeed is, very close to the problem at hand. However, we will make no apologies for changing the approach to suit, since, as observed above, we cannot really transfer the guarantees obtained in [8].

The type of quadratic program considered in [8] is the maximization of the quantity

\[
\sum_{j=1}^{\mathbf{n}} \sum_{k=1}^{\mathbf{n}} c_{jk} y_j y_k
\]

subject to \( y_j \in \{-1, 1\} \). Observe that the diagonal elements \( c_{jj} \) play no role here, adding only a constant shift. We can put our 2-type clustering problem precisely in this form, interpreting \( \{j : y_j = 1\} \) as \( T_1 \) and \( \{j : y_j = -1\} \) as \( T_2 \); for the \( c_{jk} \)'s we take the entries of \( A^\top A \).

If we took instead \( c_{jk} = C_{jk} = \mathbf{E} \mathbf{x}_j \mathbf{x}_k \), which takes (off the diagonal) only four values according to the types of \( j \) and \( k \), it is immediate that the optimum \( y = (y_1, \ldots, y_n) \) takes the same value in
coordinates representing the same type. If the cross-terms (that is, $C_{jk}$ for $j$ and $k$ of distinct types) are negative, we will have $y_j = y_k$ just when $\tau(j) = \tau(k)$, as desired.

However, owing to messages of different strengths, we can expect that often even the cross-terms to be positive. In that case, we pursue the heuristic of finding a good cut $c$ and lowering all the entries of the matrix $C$ by that amount.

Unfortunately, optimizing $\vec{y} = (y_1, \ldots, y_n)$ is not something we can expect to be able to do efficiently, unless $P=NP$; in fact it is NP-hard even to obtain a solution which approximates the optimum to better than a factor of $15/17$, in polynomial time [8]. Instead, we replace each $y_j$ by a unit vector $\vec{v}_j$ in $\mathbb{R}^n$, so that we are now maximizing

$$\sum_{j=1}^{n} \sum_{k=1}^{n} C_{jk} \vec{v}_j \vec{v}_k$$

subject to $\vec{v}_j \in \mathbb{R}^n$, $\vec{v}_j \cdot \vec{v}_j = 1$. This is now a semidefinite programming problem, which can be exactly solved (see, e.g., [4]).

To get back to the $y_j$’s, which will give us our cut, we need to convert the vectors $\vec{v}_j$ to -1’s and +1’s. This, in effect, means that our initial problem of cutting the agents in two, which can be thought of as partitioning the column vectors $X(i) = A_i, \in \{-1, 1\}^m$ into two groups, has been replaced by a problem of partitioning real unit vectors.

In the theory of computing literature, the latter job is accomplished by projection onto a random vector (and truncation). That this works rests in part, in this case, on an analytic result confirming Grothendieck’s Conjecture.

Since our objective is not to prove something assuming worst-case data but to look for the most natural solution, we replace the random vector by one which is designed to cut the $\vec{v}_j$’s as cleanly as possible. If the vectors tended to lie near $\vec{u}$ and $-\vec{u}$ for some $\vec{u}$, it would of course be obvious to choose $\vec{u}$ on which to project; we attempt to find this vector by maximizing

$$\sum_{j=1}^{n} (\vec{v}_j \cdot \vec{u})^2$$

which is easily done. We then take $y_j = \text{sign}(\vec{v}_j \cdot \vec{u})$ as our cut.

To summarize the semidefinite programming method, as adapted for our problem:

1. Compute $A^T A$ and zero the diagonal;
2. Shift the entries, if necessary, in as natural a way as possible to make the lower entries negative;
3. Solve the vector optimization problem (5) using semidefinite programming;
4. Find the vector $\vec{u}$ maximizing (6);
5. Take $T_1 = \{ j : \vec{v}_j \cdot \vec{u} > 0 \}$, $T_2 = U \setminus T_1$. 

12
11 Experimental Results with the Semidefinite Programming Approach

We tested the methods described above against the more straightforward spectral methods. The results are mixed. The quality of detection in groups seems to improve in some situations where we would like it to improve (where spectral method performs dismally). However, we were not able to predict when this happens, and given the syncretic nature of our algorithms, detailed analysis may require extensive study.

Figure 8: Comparison of quality of separation between semidefinite programming and standard spectral methods. The input data are taken from the Billboard model, with *a priori* probability 0.7. We consider three cases, \( \pi_1(\Lambda) = 0.5 - 0.9 \), letting \( \pi_2(\Lambda) \) vary. The corresponding plots are shown for the second-largest eigenvector in the standard approach, and for both eigenvectors in SDP.
12 Messages with Versions

Consider the following simplified percolation model. The agents are split into 2 groups, and information is passed from agent to agent. Once a message reaches an agent, it is forwarded to some (random) number of other agents, both inside and outside her group. Of course, the intensity of these communications should differ if there is any difference between the groups. We assume a random graph model, where the average number of links from an agent to other agents in the same group is $c_i$, while the average number of links going outside the group is $c_o < c_i$.

![Figure 9: Two examples of information propagation, both with two groups (right and left halves of the ring) with 20 agents each.](image)

We start by planting 2 messages at random among the agents, and observing who is getting the messages. The black-white experiment assumes that the messages are identical, and thus we obtain only the information of whether or not a given message was seen by a given agent. The colored experiment assumes that we can see which of the two messages (if any) a particular agent received. We assume that each agent forwards the first version received, ignoring subsequent receptions of what will seem like the same message to the agent.

In Internet traffic, the creation of different versions will usually be a simple matter, requiring only the alteration of a few otherwise inconsequential bits. Even in those cases where the agents cannot be influenced, but only observed, different versions will often arise naturally (perhaps as variations in the header) and we may be able to take advantage.

Spectral analysis of the colored (multi-version) scenario is exactly as described in Section ??, except that the matrix entries are $\{0, 1\}$-valued vectors whose length is the number of versions (2, for our experiments below). A “1” in the $i$th coordinate indicates that the $i$th version of the message was received.

Here are the results of our experiments, for varying $c_i$ and $c_o$. Green (or light grey) is used for the colored experiments, red (or dark grey) for the black-white experiments.

These results indicate that the colored variant of the method has a clear advantage over the black-white variant. This advantage is especially pronounced in the situation where the messages propagate with high probability in the population. Indeed, in the black-white variant, the difference between the groups (for a given message) would be manifested only if both planted messages landed in the same group and did not find their way into the opposite group. This is unlikely in the percolation regime. On the other hand, in the colored variant, the separating scenarios include the situation where the messages land in different groups. Here the groups will see different versions with high
probabilities.
Figure 10: Qualities of separation for the black-white and colored percolation experiments. The sizes of groups are 40 and 60; horizontal axis is $c_o$. 

16
Figure 11: Qualities of separation for the black-white and colored percolation experiments. The sizes of groups are 40 and 40; horizontal axis is $c_o$. 
13 Maintaining the Incidence Matrix

We have seen two techniques for the static problem of deducing clusters from a message incidence matrix. However, we envision a system which continually updates itself with new message data, and is capable of picking up new clusters, or revising old ones, as data improves and/or as the network itself changes.

Let us consider first the case where the network is not assumed to be changing. In that case \( m \) is increasing without bound and it becomes necessary to keep only the \( n \times n \) matrix \( A^T A \) and not the full \( m \times n \) message incidence matrix \( A \). It is true that the entries of \( A^T A \) will require increasing accuracy if we are to take advantage of the data, but this requires only space \( O(\log m) \) instead of linear in \( n \).

But maintaining \( A^T A \) is trivially easy. We merely keep track of \( m \), the total number of messages so far seen; for each agent \( j \), the sum \( S_j := \sum_{i=1}^{n} X_j(i) \); and for each pair of agents \( (j, k) \), the sum \( S_{jk} := \sum_{i=1}^{n} X_j(i) X_k(i) \). From these parameters the usual \( \ell^2 \) statistics such as mean and covariance can be recovered.

Only slightly more difficult is the issue of a changing network. The key is to choose a discount factor \( \beta \), which reduces the weight of a datum \( \alpha \) units old by \( \beta^\alpha \). (This classical method was used in very similar fashion to maintain web delay statistics in [6]). For example, if we take the unit of time to be one day and estimate that day-old message data is worth only 3/4 as much, we would take \( \beta = .75 \).

The time \( u \) of last update is maintained, along with a parameter \( m \) which now measures the sum of the weights of all messages previously recorded. When at time \( v \) a new vector \( X \) is to be entered, \( m \) is replaced by \( \beta^{v-u} m + 1 \), \( S_j \) by \( \beta^{v-u} S_j + X_j \), and \( S_{jk} \) by \( \beta^{v-u} S_{jk} + X_j X_k \).

At regular intervals or whenever a new partitioning of the agents is desired, the correlations are computed according to currently stored data and the methods described above are applied accordingly.

14 Summary and Conclusions

We have proposed a method for inferring network information, in particular clustering of accessible nodes, from records of which messages have been received by which such nodes (agents). The intention is that such data be maintained on line, and clustering be deduced when needed.

The amount of data required is linear in the number of agents and essentially insensitive to the number of messages. Moreover, as noted above, the data can conveniently be adjusted for obsolescence incurred by time.

We have proposed two approaches to producing the clusters, one using spectral methods, the other semidefinite programming (SDP). Both are polynomial-time heuristics which are not difficult to implement but come with no guarantees. Our experiments suggest that the simpler spectral methods will work well in many cases, but that the SDP approach is worth holding in reserve as it is sometimes very effective in cases where the spectral methods fail.

We have concentrated on the case where agents are partitioned into two classes, noting that the methods can be applied recursively to obtain finer partitions. However, both methods can be adapted to produce more than two clusters directly when desired.

The proposed methods rest on a simple model of agent behavior which itself is motivated by a bond percolation model for message-passing. The methods can also be applied when messages
exist in several versions, and, indeed, our experiments suggest that versions can greatly improve the determination of clusters, especially when many messages reach or exceed the percolation threshold and thus experience wide dissemination.

References


[4] F. Alizadeh, Interior point methods in semidefinite programming with applications to combinatorial optimization,


