Anthony Bonato
Jeannette Janssen (Eds.)

LNCS 7323

Algorithms and Models for the Web Graph

9th International Workshop, WAW 2012
Halifax, NS, Canada, June 2012
Proceedings

Springer

Lecture Notes in Computer Science 7323

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Anthony Bonato Jeannette Janssen (Eds.)

Algorithms and Models for the Web Graph

9th International Workshop, WAW 2012
Halifax, NS, Canada, June 22-23, 2012
Proceedings

Springer

Volume Editors

Anthony Bonato
Ryerson University
Department of Mathematics
Toronto, ON, M5B 2K3, Canada
E-mail: abonato@ryerson.ca

Jeannette Janssen
Dalhousie University
Department of Mathematics and Statistics
Halifax, NS, B3H 3J5, Canada
E-mail: janssen@mathstat.dal.ca

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-30540-5 e-ISBN 978-3-642-30541-2
DOI 10.1007/978-3-642-30541-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012937880

CR Subject Classification (1998): F.2, G.2, H.4, C.2, H.3, H.2.8, E.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume includes a selection of papers that were accepted to the 9th Workshop on Algorithms and Models for the Web Graph, WAW 2012, held at Dalhousie University in June 2012. The 13 accepted papers address a number of topics related to complex networks such as hypergraph coloring games and voter models, algorithms for detecting nodes with large degrees, random Appolonian networks, and a sublinear algorithm for Pagerank computations.

The last decade has seen intense growth in research on complex networks, ranging from the Web graph, to on-line social networks, to protein–protein interaction networks. Such research has been of great practical importance, and has also pushed the frontiers in pure mathematics and graph theory. One of the goals of the 2012 workshop was to present current research on the theory and applications of complex networks. The papers presented in this volume should stimulate new and exciting directions in research on complex networks.

We would like to thank the authors and reviewers for making the volume a reality.

June 2012

Anthony Bonato
Jeannette Janssen

Table of Contents

Hypergraph Coloring Games and Voter Models

Fan Chung and Alexander Tsiatas

Department of Computer Science and Engineering,
University of California, San Diego
{fan,atsiatas}@cs.ucsd.edu

Abstract. We analyze a network coloring game on hypergraphs which can also describe a voter model. Each node represents a voter and is colored according to its preferred candidate (or undecided). Each hyperedge is a subset of voters that can interact and influence one another. In each round of the game, one hyperedge is chosen randomly, and the voters in the hyperedge can change their colors according to some prescribed probability distribution. We analyze this *interaction model* based on random walks on the associated weighted, directed state graph. We consider three scenarios — a memoryless game, a partially memoryless game and the general game using the memoryless game for comparison and analysis. Under certain 'memoryless' restrictions, we can use semigroup spectral methods to explicitly determine the spectrum of the state graph, and the random walk on the state graph converges to its stationary distribution in $O(m \log n)$ steps, where n is the number of voters and m is the number of hyperedges. This can then be used to determine an appropriate cut-off time for voting: we can estimate probabilities that events occur within an error bound of ϵ by simulating the voting game for $O(\log(1/\epsilon)m \log n)$ rounds. Next, we consider a partially memoryless game whose associated random walk can be written as a linear combination of a memoryless random walk and another given random walk. In such a setting, we provide bounds on the convergence time to the stationary distribution, highlighting a tradeoff between the proportion of memorylessness and the time required. To analyze the general interaction model, we will first construct a companion memoryless process and then choose an appropriate *damping constant* β to build a partially memoryless process. The partially memoryless process can serve as an approximation of the actual interaction dynamics for determining the cut-off time if the damping constant is appropriately chosen either by using simulation or depending on the rules of interaction.

1 Introduction

We consider a network coloring game, motivated by human behavioral experiments [14,16] conducted in a network setting. The network coloring game can be formulated as the following interaction-based voter model:

A set of voters is modeled as the vertex set of a hypergraph $H = (V, E)$. Each hyperedge $g \in E$ represents a small, possibly overlapping group representing social interactions and discussions (such as lunchtime hallway or office discussions,

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 1–16, 2012.

blog commentary, television viewership, and Web forums). At time $t = 0$, each voter has a *color* representing an initial preference or is undecided. Then at each time step, one hyperedge g is randomly selected according to some prescribed probability distribution on E. After the voters in g interact with one another, the voters' preferences can change probabilistically. This process is repeated for many rounds, and the coloring configuration of the voters evolves.

Many natural questions arise. Will the coloring configurations converge under certain conditions? If the coloring patterns diverge or oscillate, what would be an appropriate time frame to stop the model? When the model is run for some prescribed number of rounds, how does the observed coloring configuration behave? Can the observed coloring configuration be something other than 'random'? What is the stationary distribution of the observed coloring configuration, if it exists?

The behavioral experiments of Kearns et al. [11,12,14,15] were the original motivation for our model. In these simulations, actual human agents were given a common graph coloring task with varying monetary rewards for their timely completion. This is a strategic component that is not necessarily built into our interaction-based model, but the probability distributions for color change can potentially be derived using game-theoretic analysis. The experiments allowed agents differing "views" of information about the specific graph problem structure. Thus, agents only had limited information and could only make decisions based on the color configurations of smaller subsets of nodes (but not just pairs). This gives a compelling reason to model the voters as nodes of a hypergraph and not a traditional graph where edges are limited to two endpoints.

These problems have been quite elusive in spite of extensive study either by simulation [11,12,14,15] or by using various other proposed voter models [7,10,18,19] for various types of networks [23]. The results vary widely with the specific set of rules or dynamics and point to evidence of intrinsic difficulties involved in this general voting model.

Although the general voting and coloring problems may be intractable to analyze tightly, we will consider several special cases that will help in evaluating the more general case. The first is a special case of *memoryless* voter models. In a memoryless voter model, the voters do not consider their current preferences when formulating new ones. Under these assumptions, we can show that the dynamics of coloring configurations can be analyzed quite precisely in several aspects.

Our coloring game is played on a hypergraph where each hyperedge is formed by a group to model multi-party simultaneous interaction. Although the coloring configurations of the nodes in the hypergraph do not converge in general, we can model the rounds of coloring games by conducting random walks on the associated directed state graph which contains coloring configurations as nodes. It turns out, for memoryless interactions, the spectrum of the state graph has elegant expressions. Using the formula for the eigenvalues, we can determine the rate of convergence for random walks on the state graph, which can then be used for determining a cut-off time for the voting game.

The eigenvectors of the random walk on the state graph contain rich information. In particular, the stationary distribution, which is the eigenvector

associated with eigenvalue 1, tells us the distribution of the observed coloring configuration after the cut-off time. In contrast with undirected graphs, the stationary distribution a random walk on a directed graph does not always exist and is not easy to determine, especially for state graphs with exponentially large size. Even for simple hypergraphs H such as the path or cycle, the stationary distribution for random walks on the state graph have complicated forms quite different from the uniform distribution [6]. Nevertheless, we will show that for events such as "red wins by at least 5% within an error bound ϵ" can be determined by simulating the voter model for $O(\log(1/\epsilon)m \log n)$ rounds, where m is the number of hyperedges in H, provided that the hyperedges are always chosen uniformly at random, and the interactions are memoryless.

Next, we proceed to consider the partially memoryless model for which, with some probability β, the interaction process is memoryless and with probability $1 - \beta$, the process is not required to be memoryless. Specifically, a partially memoryless voter model is one whose associated state-graph random walk can be decomposed into two parts: $P = \beta M + (1 - \beta)P'$, where M is a memoryless random walk and P' is any given random walk. Using results from the memoryless case and also from general directed Laplacians [5], we can analyze partially memoryless voter games as well.

The memoryless condition might seem to be too strict to be satisfied by real-world social interactions, but it can still be of interest to serve as a type of benchmark for the sake of comparison. In particular, the partially memoryless case provides a natural framework for evaluating a more general interaction voter process. For a given interaction voter process with its associated random walk P, we can construct a corresponding memoryless process $P' = \beta M + (1 - \beta)P$ where M is a memoryless process. The determination of M depends on β and simulation of P with the cut-off time depending on β. The details for choosing β and M will be given later in Section 7.

Related Work

The coordination or consensus game has been extensively studied in evolutionary network game theory [8,13,25]. In a set of controlled behavioral experiments, a simple voting game [12] as well as a biased version [14] were simulated on a set of small, constructed topologies. The rules varied, but involved small financial rewards for success and differing amounts of information visibility. The results showed that consensus was often reached within a certain timeframe, though there were many cases of failure as well. It is desirable to further explain these social network phenomena.

One method to analyze various consensus games is to use the standard voter model [7,10], where one agent is selected randomly during each round, assuming the vote of a randomly-selected neighbor. Some empirical work has shown that its performance depends on network topology [22]. Several proofs have been given [1,24] using coalescing random walks to show that the expected time to reach consensus under the voter model is $O(n^3 \log n)$, where n is the number of voters in the network. For the biased voting game, this method takes exponential time [16], but

it can be used as a subroutine for an algorithm converging in $O(n^8 \log n)$ expected time. However, elections in practice rarely end with a unanimous decision, and our model reflects this truth.

Other recent work consider alternative dynamics for the voting game, including Glauber dynamics [18] and pieces of "advice" [19] given to the voters. These models also result in consensus, and the time required is related to graph-theoretic quantities such as the tilted cutwidth, diameter, and broadcast time, often difficult to compute or reason about. Some of these models assume that the voters know approximate values for these quantities, which our model avoids. These models again result in unanimous coordination, whereas our proposed model is more likely to result in more realistic voting configurations.

The experiments of Kearns et al. [12,14] show that consensus is indeed not always reached, especially for biased voting games. But not much has been said thus far about the distribution of voting configurations should a true consensus be unattainable.

Furthermore, nearly all work done on the network coordination game has focused on pairwise relationships between voters. But voters often interact in groups larger than two, and most of the models described so far have not taken this possibility into account. Such interactions as town-hall debate, reader comments on news articles, and dinner-table discussions can all possibly be considered in the framework of our hypergraph model.

A Summary of Our Results

We consider a variation on the voter model that takes into account multi-voter interactions for the coordination game. Instead of limiting ourselves to pairwise relationships between voters, we model the interaction network as a hypergraph $H = (V, E)$, where each of n vertices is a voter and each of m hyperedges represents a group that can interact.

Our interaction model can be described as follows. At the beginning, all voters have some initial views, voting for one of several candidates, or starting undecided. In each round, one hyperedge g is selected randomly, and an *interaction* $X_{g,\tau}$ takes place: the randomly selected hyperedge g changes its pattern to τ. In the most general case, the probability of $X_{g,\tau}$ occurring can depend on the current coloring configuration σ of H. We denote by $p(g, \sigma, \tau)$ the probability of $X_{g,\tau}$ occurring when σ is the current coloring configuration on H, and for all σ, it must be the case that $\sum_{g,\tau} p_(g, \sigma, \tau) = 1$.

If the interactions are memoryless, then the probability of $X_{g,\tau}$ occurring is constant across all voting configurations σ on H. In this case, we can omit σ and denote by $p(g, \tau)$ the probability that $X_{g,\tau}$ occurs. When this probability only has two parameters, it can be assumed that the interaction is memoryless. The model is then simulated for some preset number of rounds.

While the game is taking place, the state can be described as a *voting configuration* or *coloring configuration* of the voters among the candidates or undecided. If there are r possible votes (including undecided), there are r^n possible configurations, and we can construct a *state graph* H^*, where the vertices are coloring

configurations, and a directed edge connects u to v if the state v is reachable from u in one round of the interaction model. Thus, a simulation of the coloring game using the interaction model can be completely described as a random walk on the state graph H^*.

Let an *event* $A \subset V(H^*)$ be a subset of the states in the state graph. These events can represent numerous scenarios: for example, the states where more than half of voters choose red, or the states which have a specific trend over some subsets of the voters, representing a voting district or municipality. In general, A can be any event of interest. We will show that for memoryless interactions, we can estimate $\Pr[A]$ within a probabilistic error bound by using a sufficient number of samplings, as long as A contains enough states.

Theorem 1. *Let A be an event or subset of all possible coloring configurations on a hypergraph H on n vertices and m edges. Suppose the interactions are memoryless, the probability of selecting a hyperedge g in any given round is at least $1/(\alpha m)$ for some α, and for any set $S \subseteq V(H)$ there are at least $|S|$ edges incident to vertices in H. The probability $\Pr[A]$ that A occurs at any cut-off time after $O(\alpha m \log n)$ rounds of simulation can be estimated within an error bound of ϵ using $O(\alpha \log(1/\epsilon) m \log(n)/\Pr[A])$ rounds of simulation.*

Note that for the special case that the hyperedges are chosen uniformly at random, the number of rounds of simulation for convergence is:

$$O(\log(1/\epsilon) m \log(n)/\Pr[A]).$$

The main tools that we use to prove the above theorem are sampling and fast mixing of random walks on the state graph H^* associated with memoryless strategies. In particular, the spectrum of such random walks on the state graph H^* can be determined by using spectral techniques originating in the analysis of card shuffling [3,4], self-organizing search [9], hyperplane arrangement [2] and semigroup random walks as well as the recent work on edge flipping games in graphs [6].

For the partially memoryless case, where the random walk P on the state graph H^* can be written as $P = \beta M + (1-\beta)P'$ with memoryless M, we have the following result showing a trade-off between the convergence time and β:

Theorem 2. *Let A be an event or subset of all possible coloring configurations on a hypergraph H on n vertices and m edges. Suppose the interactions are partially memoryless with parameter β, the probability of selecting a hyperedge g in any given round is at least $1/(\alpha m)$ for some α, and for any set $S \subseteq V(H)$ there are at least $|S|$ edges incident to vertices in H. The probability $\Pr[A]$ that A occurs at any cut-off time after $O(\alpha m \log n)$ rounds of simulation can be estimated within an error bound of ϵ using $O\left(\frac{\alpha \log \frac{1}{\epsilon} m \log n}{\beta \Pr[A]}\right)$ rounds of simulation.*

Finally, for the most general case, where the interactions are not memoryless at all, it is well known to be very difficult to deal with [6]. This is because the associated random walk can have exponentially small eigenvalues, and convergence time can vary widely based on the specific dynamics. Nevertheless, we will

show how to use insight from the memoryless and partially memoryless models to reason about the general voter model. By choosing an appropriate damping constant β and extracting the memoryless version of a given process, we can then construct a partially memoryless process which can be used to approximate the given interactive process.

2 The Voting Game on a Hypergraph as a Random Walk on the Associated State Graph

We first describe the interaction model as a coloring game on a hypergraph $H = (V, E)$. The set V of nodes consists of all voters and each hyperedge $g \in E$ represents a group of voters who can interact with one another. For a hyperedge g and coloring pattern τ on the voters in g, we let the *interaction* $X_{g,\tau}$ denote one step of the model that moves a coloring configuration of V to another by changing the color pattern to τ for voters in g.

The state graph H^* is a weighted directed graph whose vertices are coloring configurations of $V(H)$. To distinguish from nodes in H, we sometimes call a vertex in $V(H^*)$ a *state*. There is a directed edge from a state u in H^* to another state v if there is a hyperedge g and a coloring pattern τ on g such that the interaction $X_{g,\tau}$ moves u to v. The weight on the edge (u, v) is determined by the probability that $X_{g,\tau}$ occurs during state u. We denote by $p(g, u, \tau)$ the probability that, in state u, hyperedge g is selected and changes its color configuration to τ.

Note that for any directed graph with weighted edges, we can define a typical random walk with a transition from u to v occurring with probability $w(u, v)/\sum_z w(u, z)$. Therefore, the typical random walk associated with the state graph H^* simulates the evolving configurations in the voter interaction game.

Starting from a coloring configuration u, a sequence of interactions

$$X_{g_1,\tau_1} X_{g_2,\tau_2} \dots X_{g_t,\tau_t}$$

induces a series of changes in the coloring configuration and can then be viewed as a walk starting from u, traversing t directed edges on the state graph H^*. This correspondence leads to the following lemma whose proof follows from the Perron-Frobenius Theorem.

Lemma 1. *The voter interaction game with interactions as described above does not converge to an equilibrium in general. Instead, from any initial coloring configuration, the resulting configuration after t rounds of simulation, is s with probability approaching $\pi(s)$, where π is the stationary distribution of the random walk on the state graph H^*, as long as t is sufficiently large and H^* is strongly connected and aperiodic.*

An interaction $X_{g,\tau}$ is said to be *nontrivial* if the associated probability $p(g, \sigma, \tau)$ is nonzero. In the special case that all nontrivial strategies are consistent with some coloring pattern τ (for example, all red), then the coloring configuration

will reach an equilibrium. (Note that this is a special case where the interactions are memoryless.) Suppose there is a state s for which all the nontrivial interactions are of the form X_{g,s_g}, where s_g is simply the coloring configuration of s restricted to nodes in g. If this is the case, then starting from any initial configuration, the voting game will converge using standard coupon-collector probabilistic arguments:

Lemma 2. *In the voter interaction game, suppose there exists a coloring configuration s such that all the nontrivial interactions are of the form X_{g,s_g}, where s_g denotes the coloring pattern of s restricted to voters in g. Starting from any initial configuration, the voting game converges to s after t rounds of simulation with probability at least $1 - e^{-c}$ if*

$$t \geq \frac{\log n + c}{\min_{v \in V} \sum_{\substack{g \in E \\ g \ni v}} p(g, s_g)},$$

where n is the number of voters and $p(g, s_g)$ is the probability associated with the interaction. In the case that every vertex is incident to exactly d hyperedges and each hyperedge is chosen with equal probability, the above inequality is just $t \geq n(\log n + c)$.

For the remainder of this paper, we will assume that the given hypergraph and interaction dynamics yield a state graph H^* that is aperiodic and strongly connected. We will refer to the stationary distribution π accordingly.

We remark that Lemma 1 reduces the voter interaction game to random walks, and the rate of convergence depends on the eigenvalues of the directed state graph. These values can be complex, and in the most general case, the spectral gap can be exponentially small. Nevertheless, we will consider memoryless interactions which allow us to have real eigenvalues for the state graph, and we can use these techniques to derive some bounds for partially memoryless interactions as well.

3 Memoryless Interactions and Semigroup Spectral Graph Theory

For a random walk on the state graph H^*, we can describe a path in terms of the interactions $\{X_{g,\tau}\}$ that take place to follow the path. Thus, it is convenient to describe random walks as sequences of interactions. For a sequence $S = X_{g_1,\tau_1} X_{g_2,\tau_2} \dots X_{g_t,\tau_t}$ and a state u, we say $S = u$ if the interaction game ends up in state u after following the path described by S. For two sequences, we say $S_1 = S_2$ if both paths end at the same state.

We say the nontrivial interactions $\{X_{g,\tau}\}$ are *memoryless* if the probability of performing it does not depend on the current state. An equivalent definition is that for memoryless interactions, a repeated interaction $X_{g,\tau}$ means that earlier occurrences have no effect. Namely, for any three sequences of interactions S_1, S_2, S_3 of any length, then

$$S_1 X_{g,\tau} S_2 X_{g,\tau} S_3 = S_1 X_{g,\tau} S_2 S_3.$$

If the interaction strategies are memoryless, we can view them as members of a special type of semigroup known as a *left-regular band* or LRB, first studied in the 1940's [17,21]. An LRB is a semigroup where every element is idempotent, and for any two elements $x, y \in S$, $xyx = xy$. We define the product of two interactions $X_{g,\tau}$ and $X_{g',\tau'}$ to be the two interactions in sequence. If the interactions are memoryless, it is easy to see that the semigroup S generated by all nontrivial interactions $X_{g,\tau}$ is a LRB. This allows us to apply techniques in [2,3,4,6] to the voter interaction game. In particular, the associated random walk on the state graph for memoryless interactions has a clean form:

Theorem 3. *Suppose that the voter interaction game on a hypergraph $H = (V, E)$ in r colors has memoryless interactions $X_{g,\tau}$ for $g \in E$ and coloring patterns τ on g. If $p(g, \tau)$ is the probability of choosing g and coloring voters in g with the coloring pattern τ, then the random walk on the associated state graph H^* has an eigenvalue λ_T for every subset $T \subseteq V$:*

$$\lambda_T = \sum_{\substack{g,\tau \\ g \subseteq T}} p(g, \tau)$$

with multiplicity $(r-1)^{n-|T|}$.

For the specific case where the hyperedges are selected uniformly at random and there are only two colors, we note that the eigenvalues have an even cleaner form: for each subset $T \subseteq V$, there is an eigenvalue:

$$\lambda_T = \frac{|\{g \in E | g \subseteq T\}|}{m}$$

with multiplicity 1.

To prove Theorem 3, we need to explore further properties of LRB semigroups. The proof of the corollary follows from Theorem A in [6], and the theorem follows by generalizing the techniques to r colors. But in order to use these results, we must first interpret semigroup terminology in terms of the voter interaction game. These details appear in the appendix, and further details about the terminology can be found in [3].

4 The Cut-Off Time for Voter Interaction Games

Our methods also address some of the questions that arise in the recent human network experiments of Kearns et al. [16]. The voters are given a hard deadline, often arbitrarily set by various entities without justification. Furthermore, for different stopping times, the outcome of the network experiments varied widely from consensus to chaos. Using our interaction model and spectral techniques, we will prove the following theorems about the interaction model's convergence properties, as well as a mathematical interpretation of the resulting voting configuration after convergence is reached.

For voter interaction games with memoryless interactions, we have the following theorem. The proof follows by using the spectrum of H^* (derived in the previous section) to bound the total variation distance between the random walk and its stationary distribution after t steps:

Theorem 4. *Suppose the interaction model is simulated on a hypergraph $H = (V, E)$ with $|V| = n$, and each voter in V is colored with one of r colors. If the interactions are memoryless, then the total variation distance between the random walk on the state graph H^* denoted by the transition probability matrix P after t rounds of simulation and its stationary distribution π is given by*

$$\begin{aligned} \left|\left|P^t - \pi\right|\right|_{TV} &= \max_{A \subseteq V} \max_{y} \left| \sum_{x \in A} P^t(y, x) - \pi(x) \right| \\ &\leq \sum_{T \subseteq H} \lambda_T^t (r-1)^{n-|T|}. \end{aligned}$$

We remark that the above bound can be somewhat improved by restricting T to be the co-maximal subsets, although asymptotically the bound is still the same. Using this bound on the total variation distance, we can derive the convergence time for the interaction model:

Theorem 5. *On a hypergraph $H = (V, E)$ with $|V| = n, |E| = m$, suppose the voter interaction model is simulated with memoryless interactions, and the probability of choosing a hyperedge g in any round is uniform over E. Suppose for any set S of k voters, there are at least k hyperedges involving voters in S. The random walk on the state graph H^* converges to its stationary distribution in $O(m \log n)$ steps.*

To prove Theorem 5, we use the derived spectrum from Theorem 3 in our derived bound on the total variation distance from Theorem 4. The algebraic details are left for the appendix.

Theorems 4 and 5 imply a method for choosing a stopping point for the interaction model: enough time for a desired level of convergence to a stationary distribution π. This also indicates what the voting configuration among the agents looks like at any time after it has converged: the votes are a random sample from π from all the voting configurations in the state graph.

5 Estimating the Expected Value of a Given Event

Although estimating individual components of π can be computationally intractable, for memoryless interactions, we can effectively use sampling to estimate the probability of an event A, as long as the event has enough probability mass. For a general π, it can be difficult to reason about its components, since it contains exponentially many elements, corresponding to the state graph. Even for simple graphs such as the path of length k, the exact stationary distribution

on the state graph is quite complex [6]. Additionally, because most of the components of π are exponentially small, even estimating π can be quite difficult. But in practice, the exact stationary distribution is not of utmost importance. Instead, it is much more revealing and tractable to reason about larger events that capture a larger portion of π. We will use the following fact:

Theorem 6. *([20]). Let A be an event and $\pi(A) = \sum_{s \in A} \pi(s)$ be the probability that the outcome is in A. Let $\delta, \epsilon \in (0,1)$. Suppose that after N samplings, X is the proportion of times the outcome was in A. Then*

$$\Pr[(1-\delta)\pi(A) \leq X \leq (1+\delta)\pi(A)] \leq 1 - \epsilon,$$

as long as $N \geq O\left(\frac{\log(1/\epsilon)}{\pi(A)c(\delta)}\right)$, where $c(\delta)$ only depends on δ.

We can use this to prove Theorem 1. Here, we consider the case where each hyperedge is chosen with uniform probability $1/m$, but the same argument will hold using the looser $1/(\alpha m)$ bound.

Proof. (for Theorem 1) For an event A, suppose we are given an initial state f which we denote as a row vector indexed by states in $V(H^*)$. For any integer t, the coloring configuration we observe after t rounds of the voter interaction model is in A with probability

$$\mathrm{E}_t[fA] = \sum_{x \in A} fP^t(x)$$

where P denotes the transition probability matrix of the random walk on H^*.

By combining Theorems 3 and 4, we have

$$\begin{aligned} |\mathrm{E}_t[A] - \pi(A)| &\leq \max_y \left| \sum_{x \in A} P^t(y,x) - \pi(x) \right| \\ &= \left|\left| P^t - \pi \right|\right|_{TV} \\ &\leq \sum_{T \subseteq V} \lambda_T^t (r-1)^{n-|T|}. \end{aligned}$$

For the case where each hyperedge is chosen with probability $1/m$, where m is the total number of hyperedges, we have

$$\begin{aligned} |\mathrm{E}_t[A] - \pi(A)| &\leq n^2 \left(1 - \frac{1}{m}\right)^t \\ &\leq \epsilon \end{aligned}$$

if $t > 2m(\log n + \log(1/\epsilon))$.

Now we use Theorem 6, by breaking up the voter interaction game into N phases where $N = O\left(\frac{\log(1/\epsilon)}{\pi(A)c(\delta)}\right)$ and each phase consists of $t = O(m \log n)$ rounds. The proportion of phases where the outcome is in A satisfies:

$$\Pr[(1-\delta)\pi(A) < |X - \pi(A)|] < (1+\delta)\pi(A) \geq 1 - 2\epsilon. \tag{1}$$

6 The Interaction Model with Partially Memoryless Interactions

We say that a set of interactions is *partially memoryless* if the random walk on the state graph H^* can be decomposed into two parts, one of which is memoryless. Specifically, if the random walk transition matrix is P, there is a $\beta \in (0,1)$ such that we can write

$$P = \beta M + (1-\beta)P',$$

where the interactions described by M are memoryless and P' is another transition probability matrix without any restriction.

One way a partially memoryless interaction model can arise is if at each step, the voters in the selected hyperedge interact memorylessly with some probability β, allowing their actions to depend on the current step with probability $1-\beta$. It is also possible that the transition probability matrix P' is not explicitly built this way, but it can nevertheless be expressed as a partially memoryless model. In this sense, the parameter β can be viewed as describing how memoryless the model is, and its properties will depend on β:

Theorem 7. *On a hypergraph $H = (V, E)$ with $|V| = n, |E| = m$, suppose the voter interaction model is simulated with partially memoryless interactions, and the probability of choosing a hyperedge g in any round is uniform over E. Suppose for any set S of k voters, there are at least k hyperedges involving voters in S. The random walk on the state graph H^* converges to its stationary distribution in $O(\frac{m \log n}{\beta})$ steps.*

The partially memoryless structure allows us to use the results from the fully memoryless case, with an additional factor of $1/\beta$ when analyzing the spectrum. The details of the proof for Theorem 7 are left for the appendix. With this result, we can prove Theorem 2. The proof follows by using the same techniques as the proof for Theorem 1; we use Theorem 6 in the same manner. We have the same bound as (1), but the number of rounds required in each phase is now given as in Theorem 7 (Eq. (2)).

7 The General Interaction Model

Thus far, we have written about the interaction model with memoryless and partially memoryless interactions. The general version of the interaction model concerns interactions that all can depend on the current and previous states, where the random walk on the state graph H^* can be quite difficult to analyze. The eigenvalues of the random walk can be real or complex, and exponentially small [6]. With $O(2^n)$ states, computing the spectrum explicitly is too expensive or infeasible. Nevertheless, we can use the previous memoryless and partially memoryless models as a basis for comparison between a specific set of state-dependent interactions and memorylessness in general.

In particular, for any general interaction voter game, we can build a companion game which is partially memoryless with one scalar parameter β. Let M

denote the transition probability matrix of the random walk on H^*. We wish to construct a partially memoryless model

$$P' = \beta M + (1-\beta)P$$

where M is memoryless and can be constructed from P and β as follows: Instead of simulating blindly, we will use Theorem 2 which provides an upper bound on the convergence time to the stationary distribution for partially memoryless interactions. If the companion model serves as an approximation for the actual voter game, it is enough to simulate the more general model $O(\frac{m}{\beta}\log n)$ time. To construct M, we collect a sample probability distribution: when a hyperedge g is selected, keep track of the resulting coloring configurations. Then we can use these samples to build a partially memoryless model: first select a hyperedge g, then with probability β, randomly select one of the collected sample coloring configurations on g. Note that once the samples have been collected, this step is memoryless. With probability $1-\beta$, we just use the original memory-dependent dynamics.

The problem of determining the appropriate value of β can be quite difficult since the required length of time until convergence may be exponential. A feasible heuristic approach is a combination of a series of iteration and simulation. Such process also can be used to get a sense of how "memoryless" the more general model really is. In other words, an alternative interpretation of the damping constant β is just the ratio of the rate of convergence of the memoryless random walk and the actual random walk. In general, the value of β can range from 1 to some exponentially small values, reflecting the fact that some models have higher extent of memorylessness than others. So even though the general model is notoriously difficult to reason about, we can still quantify its convergence time empirically by comparing with the partially memoryless model.

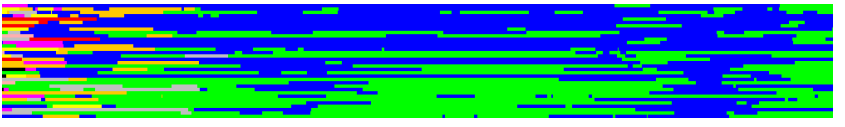
To illustrate this comparative process, we consider a traditional voting game on networks. We are given a graph G, and each node has a starting color. Then at each time step, a node is randomly selected, and it takes the color of one of its neighbors, selected uniformly at random. In our setting, there is a hyperedge for every node v, consisting of v and its neighbors, and whenever it is chosen, v changes color to one of its neighbors' colors. It should be clear that these dynamics are completely memory-dependent: the coloring configuration at time $t+1$ always depends on the state at time t. But we can build a partially memoryless process using the method described in the previous paragraph.

We demonstrate this method using Zachary's karate network [26] as an example graph. This graph has 34 nodes, and we initially assign one of 9 colors randomly to each node. Two sample runs of the consensus game are shown in Figs. 1a and 1b. Each row represents one node, and the colors change as time moves from left to right.

We note that the state graph can be as large as $9^{\binom{34}{2}}$ nodes. The convergence bound for a memoryless game of similar size is about $O(m\log n)$ steps with $n = 34$ and $m = \binom{34}{2}$. This is comparable to the time limit of 1000 steps, if each step is taken to be in the range of a fraction of a second. But in our simulations, consensus may or may not be reached.

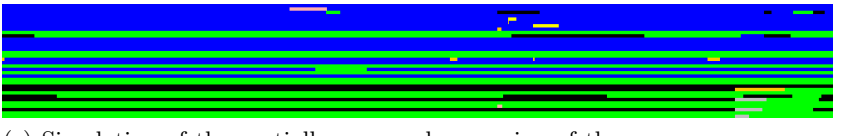
In Fig. 1c, we give an illustration of a partially memoryless version of the consensus game. Using only 100 rounds of simulation, we built the partially memoryless version and simulated it for 1000 rounds. In practice, β can be chosen empirically, by using binary search on $(0, 1)$. For Fig. 1c, we chose $\beta = 0.01$. One way to choose β is by iteratively adjusting β so that the proportion of cases that achieve consensus reaches the range of what is to be expected.

In this paper, we have used a one-parameter family of partially memoryless models to approximate a general voter game. Both the problems of finding a deterministic way to choose the parameter β and to rigorously analyze the sharpness of approximation remain open for future research.



(a) One simulation of the consensus game on Zachary's karate network for 1000 steps.

(b) Another simulation of the consensus game on Zachary's karate network for 1000 steps.



(c) Simulation of the partially memoryless version of the consensus game on Zachary's karate network for 1000 steps, with $\beta = 0.01$ and 100 steps of training.

Fig. 1. Simulations of the consensus game and its partially memoryless approximation

References

1. Aldous, D., Fill, J.A.: Reversible Markov chains and random walks on graphs (preprint), http://www.stat.berkeley.edu/~aldous/RWG/book.html
2. Bidigare, T.P., Hanlon, P., Rockmore, D.N.: A combinatorial description of the spectrum for the Tsetlin library and its generalization to hyperplane arrangements. Duke Mathematical Journal 99(1), 135–174 (1999)
3. Brown, K.S.: Semigroups, rings, and Markov chains. Journal of Theoretical Probability 13(3), 837–938 (2000)
4. Brown, K.S., Diaconis, P.: Random walks and hyperplane arrangements. Annals of Probability 26(4), 1813–1854 (1998)

5. Chung, F.: Laplacians and the Cheeger inequality for directed graphs. Annals of Combinatorics 9(1), 1–19 (2005)
6. Chung, F., Graham, R.: Edge flipping in graphs. Advances in Applied Mathematics 48(1), 37–63 (2012)
7. Clifford, P., Sudbury, A.: A model for spatial conflict. Biometrika 60(3), 581–588 (1973)
8. Ellison, G.: Learning, local interaction, and coordination. Econometrica 61(3), 1047–1071 (1993)
9. Fill, J.A.: An exact formula for the move-to-front rule for self-organizing lists. Journal of Theoretical Probability 9(1), 113–160 (1996)
10. Holley, R.A., Liggett, T.M.: Ergodic theorems for weakly interacting infinite systems and the voter model. Annals of Probability 3(4), 643–663 (1975)
11. Judd, S., Kearns, M.: Behavioral experiments in networked trade. In: Proceedings of the 9th ACM Conference on Electronic Commerce, pp. 150–159 (2008)
12. Judd, S., Kearns, M., Vorobeychik, Y.: Behavioral dynamics and influence in networked coloring and consensus. Proceedings of the National Academy of Sciences 107(34), 14978–14982 (2010)
13. Kandori, M., Mailath, G., Rob, R.: Learning, mutation, and long run equilibria in games. Econometrica 61(1), 29–56 (1993)
14. Kearns, M., Judd, S., Tan, T., Wortman, J.: Behavioral experiments on biased voting in networks. Proceedings of the National Academy of Sciences 106(5), 1347–1352 (2009)
15. Kearns, M., Suri, S., Montfort, N.: An experimental study of the coloring problem on human subject networks. Science 313(5788), 824–827 (2006)
16. Kearns, M., Tan, J.: Biased Voting and the Democratic Primary Problem. In: Papadimitriou, C., Zhang, S. (eds.) WINE 2008. LNCS, vol. 5385, pp. 639–652. Springer, Heidelberg (2008)
17. Klein-Barmen, F.: On a broader analysis of lattice theory. Mathematische Zeitschrift 46(1), 472–480 (1940) (in German)
18. Montanari, A., Saberi, A.: Convergence to equilibrium in local interaction games. In: Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 303–312 (2009)
19. Mossel, E., Schoenebeck, G.: Reaching consensus on social networks. In: Proceedings of the First Symposium on Innovations in Computer Science (ICS), pp. 214–229 (2010)
20. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press (1995)
21. Schützenberger, M.-P.: On left-regular bands. Comptes Rendus de l'Académie des Sciences 224, 777–778 (1947) (in French)
22. Suchecki, K., Eguíluz, V., San Miguel, M.: Voter model dynamics in complex networks: Role of dimensionality, disorder, and degree distribution. Physical Review E 72, 1–8 (2005)
23. Tahbaz-Salehi, A., Jadbabaie, J.: Consensus over ergodic stationary graph processes. IEEE Transactions on Automatic Control 55(1), 225–230 (2010)
24. Yildiz, M.E., Pagliari, A., Ozdaglar, A., Scaglione, A.: Voting models in random networks. In: Proceedings of the Information Theory and Applications Workshop (ITA), pp. 1–7 (2010)
25. Young, H.P.: The evolution of conventions. Econometrica 61(1), 57–84 (1993)
26. Zachary, W.W.: An information flow model for conflict and fission in small groups. Journal of Anthropological Research 33(4), 452–473 (1977)

A Semigroup Details for Proving Theorem 3

For our LRB semigroup S, there is a natural partial order defined by: $x \leq y \Leftrightarrow xy = y$. In other words, $x \leq y$ if the interaction or sequence of interactions represented by x is irrelevant after performing the sequence y. A semilattice $L(S)$ can be defined on S by considering the relation $\preceq$ on S as follows: $y \preceq x \Leftrightarrow xy = x$. The equivalence class under $\preceq$ which contains x is said to be the *support* of x, denoted by $\text{supp}(x)$, and for $x, y \in S$, $\text{supp}(xy) = \text{supp}(x) \cup \text{supp}(y)$. The support of x can be interpreted as the set of vertices whose colors were affected by the sequence of interactions given by x. The various elements of $L(S)$ are called the *flats*, and an element of S is said to be a *chamber* if its support is maximal. Therefore, the chambers of S are simply sequences of interactions x that affect the entire set of nodes V.

As given in [3,4,6], the eigenvalues of a random walk on chambers have an elegant form. For each flat $X \in L(S)$, there is an eigenvalue $\lambda_X = \sum_{x \in X} w_x$. Here, w_x is the probability of selecting the semigroup member x. In the voter interaction model, the chambers represent coloring configurations, and the probabilities w_x are simply the probabilities of choosing specific interactions. The flats are simply subsets of V, and it becomes clear how the eigenvalues of the random walk on H^* are derived. For the multiplicities, we note from [6] that the multiplicity of m_X of λ_X satisfies

$$\sum_{Y \geq X} m_Y = c_X,$$

where c_Y is the cardinality of $S_{\geq Y} = S_{\geq y} = \{z \in S : z \geq y\}$, where y is any element with support Y. (The cardinality is independent of the choice of y.) It can then be seen how the multiplicities in Theorem 3 were derived.

We note that these techniques as used in [6] were developed for an edge-flipping game with two colors. The semigroup techniques used do generalize to r colors, and the full proof for Theorem 3 can be derived in that manner.

B Proof of Theorem 5

Proof. Using Theorems 3 and 4:

$$\begin{aligned} \left|\left|P^t - \pi\right|\right|_{TV} &\leq \sum_{T \subseteq H} \lambda_T^t (r-1)^{n-|T|} \\ &= \sum_{T \subseteq V} \left(\frac{|\{g \in E | g \subseteq T\}|}{m} \right)^t (r-1)^{n-|T|} \\ &\leq \sum_{k=1}^{n} \left(\frac{\max_{|T|=k} |\{g \in E | g \subseteq T\}|}{m} \right)^t \binom{n}{k} (r-1)^{n-k}. \end{aligned}$$

Here, we indexed the subsets $T \subseteq V$ by their sizes.

For any node set T of size k, we can upper-bound the number of hyperedges contained within T. By using the fact that for any set T with $|T| = k$, there are at least k hyperedges incident to nodes in T, we have:

$$\begin{aligned}\left|\left|P^t - \pi\right|\right|_{TV} &\leq \sum_{k=1}^{n} \left(1 - \frac{k}{m}\right)^t \binom{n}{k} 2^{n-k} \\ &\leq n^2 \left(1 - \frac{1}{m}\right)^t \\ &\leq e^{-c}\end{aligned}$$

since for $f(k) = \left(1 - \frac{k}{m}\right)^t \binom{n}{k} 2^{n-k}$, we have $f(k) \geq f(k+1)$ for $k \geq 1$ and $t > 2m \log n + cn$. The theorem is proved.

C Proof of Theorem 7

Proof. Because the random walk on H^* is given by memoryless strategies, we can write its transition matrix as $\beta P_1 + (1-\beta)P_2$ for memoryless P_1. Theorem 3 allows us to analyze the eigenvalues of P_1; we can use results from [5] to see that P_2 has all eigenvalues between 0 and 1. Thus, if λ_T is an eigenvalue of P_1, then there is a corresponding eigenvalue of $P = \beta P_1 + (1-\beta)P_2$ satisfying:

$$\lambda \leq \beta \lambda_T + (1 - \beta).$$

Using Theorem 4, we have:

$$\begin{aligned}\left|\mathrm{E}_t[A] - \pi(A)\right| &\leq \max_y \left| \sum_{x \in A} P^t(y,x) - \pi(x) \right| \\ &= \left|\left|P^t - \pi\right|\right|_{TV} \\ &\leq \sum_{T \subseteq V} \lambda_T^t (r-1)^{n-|T|} \\ &\leq \sum_{k=1}^{n} \left(1 - \frac{\beta k}{m}\right)^t \binom{n}{k} (r-1)^{n-k} \\ &\leq n^2 \left(1 - \frac{\beta}{m}\right)^t \\ &\leq e^{-c}\end{aligned}$$

if

$$t \geq \frac{2m \log n + cn}{\beta}. \tag{2}$$

On a DAG Partitioning Problem

Soroush Alamdari[1] and Abbas Mehrabian[2]

[1] David R. Cheriton School of Computer Science
University of Waterloo
s26hosse@uwaterloo.ca
[2] Department of Combinatorics and Optimization
University of Waterloo
amehrabi@uwaterloo.ca

Abstract. We study the following DAG Partitioning problem: given a directed acyclic graph with arc weights, delete a set of arcs of minimum total weight so that each of the resulting connected components has exactly one sink. We prove that the problem is hard to approximate in a strong sense: If $\mathcal{P} \neq \mathcal{NP}$ then for every fixed $\epsilon > 0$, there is no $(n^{1-\epsilon})$-approximation algorithm, even if the input graph is restricted to have unit weight arcs, maximum out-degree three, and two sinks. We also present a polynomial time algorithm for solving the DAG Partitioning problem in graphs with bounded pathwidth.

Keywords: DAG Partitioning, Inapproximability, Reduction, 3-SAT, pathwidth, fixed parameter tractable.

1 Introduction

Tracking ideas and memes as they spread and evolve through the web has been studied extensively in recent years. Adar, Zhang, Adamic, and Lukose [2] studied the influence of blogs by analyzing the linking behavior of posts. They asked the question of finding a single source for each topic and assigning a topic to each post. Leskovec, Backstrom, and Kleinberg [9] formulated this question as the following DAG Partitioning problem: Given a directed acyclic graph with arc weights and n nodes, delete a set of arcs of minimum total weight so that each of the resulting connected components has a single sink. Here a *sink* refers to a vertex with no outgoing arc, and by connected components we mean weakly connected components.

In the information retrieval literature, there is a large interest on analyzing the spread of influence and topics throughout objects in the Web (see, e.g., [1,2,4,8,9,10]). Such objects can be blog posts, quotes by people, news headlines, or almost anything that appears on the Web. An immediate question is to assign a source of influence to each of these objects. We can model these objects by a directed graph, in which the vertices represent the objects and the arcs, which are weighted, represent traces of possible influence. These arcs can be extracted from the Web using different methods, such as studying linkage structure or

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 17–28, 2012.

studying occurrences of similar phrases; in the latter case, the weight of an arc is the degree of similarity between phrases. These objects influence each other in an acyclic manner in reality, but this might not be the case in the retrieved graph of influence. Yet, there are ways to find an acyclic subgraph of a given graph without dramatically distorting the structure of the graph (see [3] for instance). Once a directed acyclic graph is formed, the most natural way to assign a source to the objects, seems to be the DAG Partitioning problem we study here.

Leskovec et al. [9] proved the $\mathcal{NP}$-hardness of the DAG Partitioning problem, by reducing the Multiway Cut problem to it (see [6] for the definition of the latter problem). Their reduction converts an instance of the Multiway Cut Problem with k terminals to an instance of the DAG Partitioning problem with k sinks. The Multiway Cut problem can be solved efficiently for 2 terminals, and when there are k terminals, a $(\frac{3}{2} - \frac{1}{k})$-approximation algorithm is known [6]. Thus one might expect similar approximation algorithms for the DAG Partitioning problem as well. In section 2 we show that this is far from being true, and even the simplest case of this problem is hard to approximate. Indeed, assuming $\mathcal{P} \neq \mathcal{NP}$, for every fixed $\epsilon > 0$, there is no $(n^{1-\epsilon})$-approximation algorithm for the DAG Partitioning problem. It is a standard assumption to restrict the graph to have vertices with only constant number of outgoing arcs, since it is true for real instances of the problem. We show that our hardness result holds, even if the input graph is restricted to have unit weights, maximum out-degree three, and just two sinks. We use reduction from the 3-SAT problem.

The *pathwidth* of a directed graph is simply the pathwidth of its undirected underlying graph. In section 3 we study the parameterized version of the problem, with pathwidth chosen as the parameter. We show that the problem is fixed parameter tractable. More precisely, we present an algorithm that given a path decomposition of the graph with width k, solves the DAG Partitioning problem optimally and has running time $2^{O(k^2)}n$. Thus, for graphs with bounded pathwidth, the problem is solvable in linear time. We conclude with an open problem in section 4.

2 The Hardness Result

For a directed acyclic graph D, a *good cut* is a subset C of arcs such that when deleted from D, each of the resulting connected components has a single sink. So the DAG Partitioning problem is just the problem of finding a good cut with minimum weight. The *size* of an instance of a 3-SAT problem is simply the number of clauses in the instance, and in the following we will assume that this number is sufficiently large.

The basic construction we use is the following.

Definition. Let $\mathcal{I}$ be an instance of the 3-SAT problem, and M be a positive integer. Arcs of weight M are called the *heavy* arcs, and the rest of the arcs are called the *light* arcs. The directed graph $D = D(\mathcal{I}, M)$ is the following:

- There are four special vertices t, t', f, f' in D, where (t', t) and (f', f) are heavy arcs.

- For each variable x of $\mathcal{I}$, there are
  - four vertices $x, \overline{x}, x_t, x_f$,
  - two heavy arcs (t', x_t) and (f', x_f), and
  - eight arcs (x_t, x), $(x_t, \overline{x})$, (x, t), $(\overline{x}, t)$, (x_f, x), $(x_f, \overline{x})$, (x, f), and $(\overline{x}, f)$ of weight one (see Figure 1).
- For each clause $L = (a \vee b \vee c)$, there is
  - a vertex L,
  - a heavy arc (t', L), and
  - three arcs $(L, a), (L, b), (L, c)$ of weight one (see Figure 2).

  Note that a, b, c denote literals here, and a vertex is already associated with each of them.

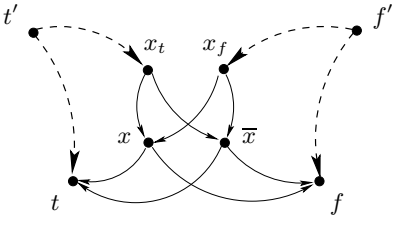


Fig. 1. The gadget corresponding to a variable: heavy arcs are dashed and light arcs are solid.

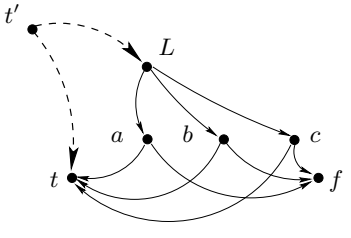


Fig. 2. The gadget corresponding to a clause: heavy arcs are dashed and light arcs are solid.

It is easy to verify that if $\mathcal{I}$ has size $s > 3$, then $D(\mathcal{I}, M)$ has at most $14s$ vertices, $27s$ light arcs and $8s$ heavy arcs. Moreover, $D(\mathcal{I}, M)$ is acyclic and has just two sinks t and f.

Lemma 1. *The instance $\mathcal{I}$ is satisfiable if and only if $D = D(\mathcal{I}, M)$ has a good cut that does not contain any heavy arcs.*

Proof. ($\Rightarrow$) Consider a satisfying assignment of the variables. Build the subset T of vertices of D as following: Put t, t' in T. For each clause L, put L in T. For each variable x, put x_t in T. If x is true then put x in T, otherwise put $\overline{x}$ in T. Do not put any other vertex in T.

Let C be the set of arcs with exactly one endpoint in T. Then C does not contain any heavy arcs. Deleting C results in a directed acyclic graph H with two connected components, with one of them containing the sink t and the other one containing the sink f. For each variable x, exactly one of x and $\overline{x}$ is in T, so none of x_t and x_f is a sink in H. If x is true, then $(x,t),(\overline{x},f)$ are arcs in H, and if x is false, then $(x,f),(\overline{x},t)$ are arcs in H, so none of $x,\overline{x}$ is a sink in H. For each clause $L=(a\vee b\vee c)$, at least one of a,b,c is in T, so L is not a sink in H. Thus it can be verified that C is a good cut that does not contain any heavy arcs.

($\Leftarrow$) Let C be a good cut of minimum size that does not contain any heavy arcs. Let H be the directed acyclic graph obtained from deleting C. We claim that H has two connected components. First, it has at least two components as t and f are sinks in H. If it has a third component (other than the components containing t and f), then let r be a sink in the third component. Let e be an arbitrary outgoing arc from r in G. Then $C\setminus\{e\}$ is a good cut with a smaller size that does not contain any heavy arcs, contradicting the choice of C. Hence H has two connected components and so t and f are the only sinks in H. Denote the components containing t and f by T and F, respectively.

For each variable x, if $x\in T$ then let x be true, and let x be false otherwise. Observe that:

- Since C has no heavy arcs, $t'\in T$ and $f'\in F$.
- For each variable x, we have $x_t\in T$ and $x_f\in F$.
- Since for each variable x, none of x_t or x_f is a sink in H, exactly one of $x,\overline{x}$ is in T and the other one is in F (see Figure 1).
- Since C has no heavy arc, for each clause L, vertex L is in T.
- As vertex L is not a sink in H, at least one of the vertices a,b,c is in T (see Figure 2).

Therefore, this is a satisfying assignment, and the proof is complete. □

Corollary 1. *Let $\mathcal{I}$ be an instance of 3-SAT of size s, where $s>3$. If $\mathcal{I}$ is satisfiable then the optimal value of $D(\mathcal{I},M)$ is at most $27s$. Otherwise, the optimal value of $D(\mathcal{I},M)$ is at least M.*

Now, we alter the construction so that we just have unit weights and out-degrees at most 3.

Definition. Let $\mathcal{I}$ be an instance of the 3-SAT problem, and M be a positive integer. Note that the only arcs of non-unit weight, are those going out from t' and f'. Also, the only vertices with out-degree more than three are t' and f'. The directed graph $\overline{D}(\mathcal{I},M)$ is obtained from $D(\mathcal{I},M)$ as follows. For each heavy arc (t',v), add M new vertices $x_1,x_2,\ldots,x_M$, and add the arcs $(x_1,v),(x_1,t),(x_2,v),(x_2,t),\ldots,(x_M,v),(x_M,t)$. Perform the same procedure for all heavy outgoing arcs from f'. Finally, delete t',f', and all adjacent arcs.

Note that the directed graph $\overline{D}(\mathcal{I},M)$ has $\Theta(sM)$ vertices and $\Theta(sM)$ arcs, and can be constructed in time polynomial in s and M. Moreover, it is acyclic

and all of its arcs have unit weights. Also, all vertices have out-degree at most three. It is easy to check that Corollary 1 remains true for $\overline{D}(\mathcal{I}, M)$.

Corollary 2. *Let $\mathcal{I}$ be an instance of 3-SAT of size s, where $s > 3$. If $\mathcal{I}$ is satisfiable then the optimal value of $\overline{D}(\mathcal{I}, M)$ is at most $27s$. Otherwise, the optimal value of $\overline{D}(\mathcal{I}, M)$ is at least M.*

Definition. The Restricted DAG Partitioning problem is the following problem. The input is a directed acyclic graph G with arc weights, such that the weight of each arc is one, the out-degree of each vertex is at most three, and the graph has exactly two sinks. The output is a set C of arcs of minimum total weight such that each of the connected components of $G - C$ has a single sink.

We are now ready to prove our hardness result.

Theorem 1. *Assume that $\epsilon > 0$ is fixed and there is a polynomial-time $(n^{1-\epsilon})$-approximation algorithm for the Restricted DAG Partitioning problem. Then the 3-SAT problem is in $\mathcal{P}$.*

Proof. Let $\mathcal{I}$ be an instance of 3-SAT of size s, where $s > 3$. Take $M = \Theta(s^{2/\epsilon})$, so that $s^{2-\epsilon} = o(M^{\epsilon})$, and construct the instance $\overline{D}(\mathcal{I}, M)$ of the Restricted DAG Partitioning problem, which has size $\Theta(sM)$. Let w be the weight of the solution found by the approximation algorithm. If $\mathcal{I}$ is satisfiable, then the optimal value is $O(s)$, so we would have $w = O\left(s(sM)^{1-\epsilon}\right) = o(M)$. Otherwise, the optimal value is at least M, so $M \leq w$. Hence one can decide whether $\mathcal{I}$ is satisfiable in polynomial time. □

3 Linear-Time Algorithm for Bounded-Pathwidth Graphs

A *tree decomposition* of a directed graph G is a pair (T, W), where T is a tree and $W = (W_t : t \in V(T))$ is a family of (not necessarily induced) directed subgraphs of G such that

(i) $\bigcup_{t \in V(T)} V(W_t) = V(G)$, and every arc of G has both endpoints in some W_t, and
(ii) For every $v \in V(G)$, the set $\{t : v \in V(W_t)\}$ induces a subtree of T.

The *width* of (T, W) is

$$\max\{|V(W_t)| - 1 : t \in V(T)\},$$

and the *treewidth* of G is the minimum width of a tree decomposition of G. If each $t \in V(T)$ has degree at most 2, then (T, W) is called a *path decomposition* of G. The *pathwidth* of a graph G, written $pw(G)$, is the minimum width of a path decomposition of G. For clarity we will refer to the vertices of T as the *nodes*. W_t is called the *bag* of the decomposition corresponding to the node t. Note that sometimes the bags are simply defined as subsets of vertices of G, but here we assume that each bag also contains a subset of arcs of G. The value of pathwidth is the same in either definition.

In this section we present an algorithm that, given a DAG and a path decomposition of width k, solves the DAG Partitioning problem optimally, and has running time $2^{O(k^2)}n$. The algorithm first chooses an arbitrary degree one node of the decomposition as its root, and then (to simplify the main part) augments the decomposition so that it has the following properties:

1. Every arc appears in exactly one bag, and every bag has at most one arc.
2. If a bag W_t has an arc, then the corresponding node t has a child t_2. Let t_1 be the parent of t. Then W_{t_1}, W_t and W_{t_2} have the same vertex set, and W_{t_1} and W_{t_2} do not have an arc.
3. If node t has a child t', then the vertex sets of W_t and $W_{t'}$ differ by at most one vertex. That is,

$$|(V(W_t) \cup V(W_{t'})) \setminus (V(W_t) \cap V(W_{t'}))| \leq 1.$$

4. The bags corresponding to the root and the (only) leaf of the decomposition have empty vertex sets.

It is not hard to see that given a path decomposition with $O(n)$ bags, it is possible to augment it to a desired one with $O(k^2 n)$ bags.

An important observation is that in a DAG G, if for every $v \in V(G)$ there is a unique sink $s \in V(G)$ such that v has a directed path to s, then every connected component of G has a unique sink. Hence, if C is any solution for the DAG Partitioning problem for DAG G, then in $G - C$, to every vertex v is assigned a unique sink s, which we will sometimes call "the sink" of v in this solution.

We use dynamic programming on the path decomposition. Let us define the subproblems. Let H be a subgraph of G, and let $\mathcal{X} \subseteq V(H)$ be such that there is no arc from $V(G) \setminus V(H)$ to $V(H) \setminus \mathcal{X}$ or vice versa. Let $G - H$ denote the subgraph obtained from G by removing the *arcs* of H. Let $\mathcal{P} = \{P_1, \ldots, P_s\}$ be a partition of $\mathcal{X}$. Let $\mathcal{F} \subseteq \mathcal{X}$, and let $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{X}$ be a set of ordered pairs of distinct elements of $\mathcal{X}$ that does not induce a cycle in $\mathcal{X}$. One can build a DAG H' from H by adding $2s$ vertices $a_1, \ldots, a_s, b_1, \ldots, b_s$, and adding the arcs

$$\{(a_i, u) : u \in P_i, i = 1 \ldots s\} \cup \{(u, b_i) : u \in P_i \cap \mathcal{F}, i = 1 \ldots s\} \cup \{(u, v) : (u, v) \in \mathcal{D}\}$$

with weight ∞. The subproblem defined by $(H, \mathcal{X}, \mathcal{P}, \mathcal{F}, \mathcal{D})$ is the DAG Partitioning problem on H'. Informally speaking, this is the DAG Partitioning problem, confined to H, with the following extra restrictions:

- Vertices in $\mathcal{F}$ should not have their sink in H.
- Vertices in the same element of $\mathcal{P}$ should have the same sink, and vertices in different elements should have distinct sinks.

and the following assumptions about $G - H$:

- Vertex $v \in \mathcal{X}$ is in $\mathcal{F}$ if and only if v has a path in $G - H$ to a sink, and this sink is out of H. Note that by definition of the problem, in any solution there is a many-to-one mapping from vertices to sinks. Therefore, we refer to such a sink as the sink of v.

- For any pair $\{u, v\}$ of vertices in $\mathcal{X} \cap \mathcal{F}$, if u and v are in the same element of $\mathcal{P}$ then their sink (which is in $V(G) \setminus V(H)$) is the same, otherwise their sinks are distinct.
- For every pair $(u, v) \in \mathcal{X} \times \mathcal{X}$, we have $(u, v) \in \mathcal{D}$ if and only if there is a (u, v)-path in $G - H$.
- For every pair $(u, v) \in \mathcal{D}$, u and v are in the same element of $\mathcal{P}$.

In general, there can be exponentially many subproblems. However, using the path decomposition, one can choose a polynomial number of them, solving which obtains the optimal solution for the main problem. Let t be a node of the path decomposition, whose corresponding bag, W_t, does not have an arc. Let $\mathcal{X}_t = V(W_t)$, let H_t be the subgraph of G formed by taking the union of the bag W_t with all bags whose nodes are the descendants of t. Let $\mathcal{P} = \{P_1, \ldots, P_s\}$ be a partition of $\mathcal{X}_t$. Let $\mathcal{F} \subseteq \mathcal{X}_t$, and let $\mathcal{D} \subseteq \mathcal{X}_t \times \mathcal{X}_t$ be a set of ordered pairs of distinct elements of $\mathcal{X}_t$ that does not induce a cycle in $\mathcal{X}_t$. The subproblem defined by $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ is equivalent to the subproblem defined as $(H_t, \mathcal{X}_t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ above, and we denote its optimal values by $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$. Next we illustrate an algorithm for calculating $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ based on the optimal values found for the descendants of t in the decomposition. But before doing so, let us define some notation.

Let $\mathcal{P} = \{P_1, \ldots, P_s\}$ be a partition of a set $\mathcal{X}$. For an element $v \notin \mathcal{X}$, $\mathcal{P} * v$ is the following family of partitions of $\mathcal{X} \cup \{v\}$:

$$\mathcal{P} * v = \{\{\{v\}, P_1, P_2, \ldots, P_s\}, \{P_1 \cup \{v\}, P_2, \ldots, P_s\}, \ldots, \{P_1, P_2, \ldots, P_s \cup \{v\}\}\}.$$

For an element $v \in X$, $\mathcal{P}/v$ is the following partition of $\mathcal{X} \setminus \{v\}$:

$$\mathcal{P}/v = \{P_1 \setminus \{v\}, \ldots, P_s \setminus \{v\}\}.$$

For a set $\mathcal{D}$ of pair of vertices of G and a vertex v, $\mathcal{D}/v$ is obtained from $\mathcal{D}$ by deleting all pairs in which v appears.

Theorem 2. *Algorithm 1 calculates* $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ *correctly in time* $O(k^3)$*; therefore, the DAG Partitioning problem on* G *can be solved in time* $2^{O(k^2)}n$.

Proof. We prove correctness by induction. For any node t of the path decomposition and any valid tuple $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$, we show that the value of $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ is calculated correctly, assuming this has been the case for the child of t, if any. The induction base is true since the bag corresponding to the unique leaf of the path decomposition is an empty subgraph so its optimal value is 0 [lines 1-2].

Now, assume that t is not a leaf, and t' is its unique child. Recall that H_t is the subgraph of G formed by taking the union of the bag W_t with all bags whose nodes are the descendants of t, and $H_{t'}$ is defined similarly. Let $H = H_t$, $\mathcal{X} = V(W_t)$, $H' = H_{t'}$ and $\mathcal{X}' = V(W_{t'})$. Recall that if $\mathcal{P}$ is a partition of $\mathcal{X}$,

Algorithm 1. Calculate $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$

if t has no child **then**
  $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = 0$
else
  $t' \leftarrow$ the child of t
  if $V(W_{t'})$ is $V(W_t) \cup \{v\}$ for some vertex v **then**
    $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \min\{\mathrm{OPT}(t', \mathcal{Q}, \mathcal{F}, \mathcal{D}) : Q \in \mathcal{P} * v\}$
  else if $V(W_{t'})$ is $V(W_t) \setminus \{v\}$ for some vertex $v \in V(W_t)$ **then**
    $P \leftarrow$ the element of $\mathcal{P}$ containing v
    if $P = \{v\}$ **then**
      $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P} \setminus \{\{v\}\}, \mathcal{F} \setminus \{v\}, \mathcal{D}/v)$
    else if $v \in \mathcal{F}$ **then**
      if there is a $u \in P \setminus \{v\}$ with $u \in \mathcal{F}$ **then**
        $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v)$
      else
        $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \infty$
      end if
    else if there is a $u \in P \setminus \{v\}$ with $(v, u) \in \mathcal{D}$ **then**
      $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v)$
    else
      if for all $u \in P \setminus \{v\}$ we have $u \notin \mathcal{F}$
      and for some $u \in P \setminus \{v\}$ we have $(u, v) \in \mathcal{D}$ **then**
        $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \cup \{u \in P \setminus \{v\} : (u, v) \in \mathcal{D}\}, \mathcal{D}/v)$
      else
        $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \infty$
      end if
    end if
  else if $W_{t'}$ is $W_t \cup \{(u, v)\}$ for some arc (u, v) **then**
    $t'' \leftarrow$ the child of t'
    if u and v are in different elements of $\mathcal{P}$ **then**
      $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = w(u, v) + \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}, \mathcal{D})$
    else
      $S \leftarrow \{x \in V(W_t) : (x, u) \in \mathcal{D}\} \cup \{u\}$
      $T \leftarrow \{y \in V(W_t) : (v, y) \in \mathcal{D}\} \cup \{v\}$
      $\mathcal{D}' \leftarrow \mathcal{D} \cup \{(x, y) : x \in S, y \in T\}$
      if $v \in \mathcal{F}$ **then**
        $\mathcal{F}' \leftarrow \mathcal{F} \cup S$
      else
        $\mathcal{F}' \leftarrow \mathcal{F}$
      end if
      $\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \min\{w(u, v) + \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}, \mathcal{D}), \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}', \mathcal{D}')\}$
    end if
  end if
end if

$\mathcal{F} \subseteq \mathcal{X}$ and $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{X}$, then $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ is the DAG Partitioning problem confined to H with the following extra restrictions:

- Vertices in $\mathcal{F}$ should not have their sink in H;
- Vertices in the same element of $\mathcal{P}$ should have the same sink, and vertices in different elements should have distinct sinks;

and the following assumptions about $G - H$:

- Vertex $v \in \mathcal{X}$ is in $\mathcal{F}$ if and only if v has a path in $G - H$ to its sink, and the sink of v is out of H.
- For any pair $\{u, v\}$ of vertices in $\mathcal{X} \cap \mathcal{F}$, if u and v are in the same element of $\mathcal{P}$ then their sink (which is in $V(G) \setminus V(H)$) is the same, otherwise their sinks are distinct.
- For every pair $(u, v) \in \mathcal{X} \times \mathcal{X}$, we have $(u, v) \in \mathcal{D}$ if and only if there is a (u, v)-path in $G - H$.
- For every pair $(u, v) \in \mathcal{D}$, u and v are in the same element of $\mathcal{P}$.

Because of the augmentation of the path decomposition, the relation between W_t and $W_{t'}$ is of one of the following three types.

Type 1: $\mathcal{X}' = \mathcal{X} \cup \{\mathbf{v}\}$ (LINES 5-6) In this case H and H' are the same. Any solution for $(t', \mathcal{P}', \mathcal{F}, \mathcal{D})$ gives a solution with the same value for $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$, as long as the partitions $\mathcal{P}$ and $\mathcal{P}'$ are consistent. That is, $\mathcal{P}'$ should keep the partitioning of $\mathcal{X}$, and then either throw the new vertex v into one of the elements of the partition, or put it in a new singleton element. Moreover, all solutions for $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$ are obtained this way. Hence [see lines 5-6]

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \min\{\mathrm{OPT}(t', \mathcal{Q}, \mathcal{F}, \mathcal{D}) : \mathcal{Q} \in \mathcal{P} * v\}.$$

Type 2: $\mathcal{X} = \mathcal{X}' \cup \{\mathbf{v}\}$ (LINES 7-25) In this case H has an isolated vertex v which H' does not have. Let P be the element of the partition containing v. Four cases may happen:

(a) P is a singleton (LINES 9-10). First, assume that $v \in \mathcal{F}$. So v has a path in $G - H$ to its sink, and its sink is in $V(G) \setminus V(H)$. Also v is in a different element of partition from any other vertex $u \in \mathcal{X} \setminus \{v\}$, hence the sinks of u and v are different. So to solve this subproblem, one just needs to remove v and solve the resulting subproblem [see lines 9-10]:

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P} \setminus \{\{v\}\}, \mathcal{F} \setminus \{v\}, \mathcal{D}/v).$$

Now, assume that $v \notin \mathcal{F}$. Therefore v does not have a path in $G - H$ to its sink. Note that the sink of v can not be in H' since P is a singleton and $v \notin \mathcal{X}'$. Thus v is a sink itself, so for any other vertex $u \in \mathcal{X} \setminus \{v\}$, the sinks of u and v are different. So to solve this subproblem, again one just needs to remove v and solve the resulting subproblem [see lines 9-10]:

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P} \setminus \{\{v\}\}, \mathcal{F} \setminus \{v\}, \mathcal{D}/v).$$

(b) P is not a singleton, and $v \in \mathcal{F}$ (LINES 11-16). In this case v has a path in $G - H$ to its sink, which is in $V(G) \setminus V(H)$. First, assume that there is no $u \in P \setminus \{v\}$ with $u \in \mathcal{F}$. Then the sink of v is in $V(G) \setminus V(H)$ while each vertex $u \in P \setminus \{v\}$ has either its sink in $V(H)$, or has no path to its sink in $G - H$. Thus the subproblem is infeasible [see lines 14-15].

Now, assume that there is some $u \in P \setminus \{v\}$ with $u \in \mathcal{F}$. Then we know that the sink of u and v is the same. Hence in any solution for $(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v)$, a vertex $w \in \mathcal{X}$ has the same sink as v, if and only if it has the same sink as u, if and only if it is in P. Hence we have [see lines 12-13]

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v).$$

(c) P is not a singleton, $v \notin \mathcal{F}$, and there is a $u \in P \setminus \{v\}$ with $(v, u) \in \mathcal{D}$ (LINES 17-18). In any solution for $(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v)$, the sink of v is the same as the sink of u; thus a vertex $w \in \mathcal{X}$ has the same sink as v, if and only if it has the same sink as u, if and only if it is in P. Hence we have [see lines 17-18]

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \setminus \{v\}, \mathcal{D}/v).$$

(d) P is not a singleton, $v \notin \mathcal{F}$, and there is no $u \in P \setminus \{v\}$ with $(v, u) \in \mathcal{D}$ (LINES 19-25). Since v has no outgoing arc in H or $G - H$, it is a sink in any solution for $(t, \mathcal{P}, \mathcal{F}, \mathcal{D})$. However, if there is a $u \in P \setminus \{v\}$ with $u \in \mathcal{F}$, then the sink of u is in $V(G) \setminus V(H)$, so the sink of u and the sink of v are distinct, and the subproblem is infeasible. Otherwise, if there is no $u \in P \setminus \{v\}$ with $(u, v) \in \mathcal{D}$, then for any $u \in P \setminus \{v\}$, the sink of u will be in $V(H) \setminus \{v\}$ while the sink of v is v, and again the subproblem would be infeasible [see lines 20-24].

So, assume that for all $u \in P \setminus \{v\}$ we have $u \notin \mathcal{F}$, and for some $u \in P \setminus \{v\}$ we have $(u, v) \in \mathcal{D}$. In this case one can remove v and solve the remaining subproblem. but it should be taken into account that vertices $u \in P \setminus \{v\}$ with $(u, v) \in \mathcal{D}$ have a path in $G - H'$ to their sink (which is $v \in V(G) \setminus V(H')$). Consequently [see lines 20-21],

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \mathrm{OPT}(t', \mathcal{P}/v, \mathcal{F} \cup \{u \in P \setminus \{v\} : (u, v) \in \mathcal{D}\}, \mathcal{D}/v).$$

Type 3: $\mathcal{X} = \mathcal{X}'$ and $W_{t'} = W_t \cup \{(u, v)\}$ (LINES 26-42) Let t'' be the child of t', and let $H'' = H_{t''}$. Here H is the same as $H'' \cup (u, v)$. If u and v are in distinct elements of $\mathcal{P}$, then clearly the arc (u, v) should be in solution set (the set of arcs that are cut), that is [see lines 28-29],

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = w(u, v) + \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}, \mathcal{D}).$$

Otherwise, one has the choice of putting the arc (u, v) in the solution set or not. If not, then when reducing the subproblem to a smaller one corresponding to H'', one should take into account the arc (u, v) which is not in H''. Let

$$S = \{x \in \mathcal{X} : (x, u) \in \mathcal{D}\} \cup \{u\}, \quad T = \{y \in \mathcal{X} : (v, y) \in \mathcal{D}\} \cup \{v\}.$$

Then for any $(x, y) \in S \times T$, the arc (u, v) creates an (x, y)-path in $G - H''$. Thus when solving the subproblem corresponding to H'', the set

$$\mathcal{D}' = \mathcal{D} \cup \{(x, y) : x \in S, y \in T\}$$

is precisely the set of pairs (x, y) such that there is an (x, y)-path in $G - H''$ [lines 31-33]. Moreover, if $v \in \mathcal{F}$, then any vertex in S has its sink in $V(G) \setminus V(H) = V(G) \setminus V(H'')$, and a path to its sink in $G - H''$. Thus, letting

$$\mathcal{F}' = \begin{cases} \mathcal{F} \cup S & v \in \mathcal{F} \\ \mathcal{F} & v \notin \mathcal{F}, \end{cases}$$

we have [see lines 34-39]

$$\mathrm{OPT}(t, \mathcal{P}, \mathcal{F}, \mathcal{D}) = \min\{w(u, v) + \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}, \mathcal{D}), \mathrm{OPT}(t'', \mathcal{P}, \mathcal{F}', \mathcal{D}')\},$$

where the first term in the minimum corresponds to putting the arc (u, v) in the solution, and the second term corresponds to not doing so.

Finally, we analyze the running time. First, observe that Algorithm 1 has running time $O(k^3)$ given that the size of the bags is at most $k + 1$. Let x be the root of the path decomposition. Note that the vertex set of W_x is empty, thus the optimal value of the DAG Partitioning problem on G is simply $\mathrm{OPT}(x, \emptyset, \emptyset, \emptyset)$. The total number of subproblems is $2^{O(k^2)}n$, and each of them can be solved in time $O(k^3)$ (provided the solutions to smaller subproblems have been calculated), which gives a total running time of $2^{O(k^2)}n$. □

4 Concluding Remarks

We showed that even the simplest instances of the DAG Partitioning problem are hard to approximate. Our result implies that the hardness arises from the global structure of the graph, and not from the weight of the arcs, the number of sinks, or the "local complexity" arising from a vertex with large out-degree. Thus, when encountered with this problem, one should naturally try to use heuristics that behave well in practice. This was the approach taken by the authors of [9].

We proved the problem is fixed parameter tractable, when the pathwidth of the graph is chosen as the parameter. A natural question is, what is the complexity if treewidth is chosen instead? It is known (see [7]) that a graph with treewidth k, has pathwidth $O(k \log n)$, so our algorithm has running time $n^{O(k^2 \log n)}$ on such a graph. Unfortunately our dynamic programming does not work correctly on tree decompositions, but it might be possible to alter it and come up with an algorithm with running time linear in n. Courcelle [5] proved that any property of graphs definable in monadic second-order logic (MSO2) can be decided in linear time on any class of graphs with bounded treewidth. So, another approach may be to formulate this problem using monadic second-order logic.

References

1. Adar, E., Adamic, L.A.: Tracking information epidemics in blogspace. In: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2005, pp. 207–214. IEEE Computer Society, Washington, DC (2005), http://dx.doi.org/10.1109/WI.2005.151
2. Adar, E., Zhang, L., Adamic, L.A., Lukose, R.M.: Implicit Structure and the Dynamics of Blogspace. In: WWW 2004 Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics. ACM Press, New York (2004)
3. Berger, B., Shor, P.W.: Approximation alogorithms for the maximum acyclic subgraph problem. In: Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 1990, pp. 236–243. Society for Industrial and Applied Mathematics, Philadelphia (1990), http://dl.acm.org/citation.cfm?id=320176.320203
4. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: Proceedings of the 23rd International Conference on Machine Learning, ICML 2006, pp. 113–120. ACM, New York (2006), http://doi.acm.org/10.1145/1143844.1143859
5. Courcelle, B.: The monadic second-order logic of graphs. i. recognizable sets of finite graphs. Inf. Comput. 85, 12–75 (1990), http://dl.acm.org/citation.cfm?id=81253.81255
6. Călinescu, G., Karloff, H., Rabani, Y.: An improved approximation algorithm for multiway cut. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, STOC 1998, pp. 48–52. ACM, New York (1998), http://doi.acm.org/10.1145/276698.276711
7. Korach, E., Solel, N.: Tree-width, path-width, and cutwidth. Discrete Appl. Math. 43, 97–101 (1993), http://dl.acm.org/citation.cfm?id=153610.153618
8. Kwon, Y.S., Kim, S.W., Park, S., Lim, S.H., Lee, J.B.: The information diffusion model in the blog world. In: Proceedings of the 3rd Workshop on Social Network Mining and Analysis, SNA-KDD 2009, pp. 4:1–4:9. ACM, New York (2009), http://doi.acm.org/10.1145/1731011.1731015
9. Leskovec, J., Backstrom, L., Kleinberg, J.: Meme-tracking and the dynamics of the news cycle. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, pp. 497–506. ACM, New York (2009), http://doi.acm.org/10.1145/1557019.1557077
10. Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N., Hurst, M.: Cascading behavior in large blog graphs: Patterns and a model. In: Society of Applied and Industrial Mathematics: Data Mining, SDM 2007 (2007)

Some Typical Properties of the Spatial Preferred Attachment Model

Colin Cooper[1], Alan Frieze[2], and Paweł Prałat[3]

[1] Department of Computer Science, Kings College, University of London, London WC2R 2LS, UK
[2] Department of Mathematical Sciences, Carnegie Mellon University, 5000 Forbes Av., 15213, Pittsburgh, PA, U.S.A
[3] Department of Mathematics, Ryerson University, Toronto, ON, Canada, M5B 2K3

Abstract. We investigate a stochastic model for complex networks, based on a spatial embedding of the nodes, called the Spatial Preferred Attachment (SPA) model. In the SPA model, nodes have spheres of influence of varying size, and new nodes may only link to a node if they fall within its influence region. The spatial embedding of the nodes models the background knowledge or identity of the node, which influences its link environment. In this paper, we focus on the (directed) diameter, small separators, and the (weak) giant component of the model.

1 Introduction

Discrete random graph processes exhibiting power law properties have been studied by many authors and in many contexts. The study of such processes dates back at least, to Yule [28] in 1924. Recent interest in preferential attachment models follows from the work of Barabási and Albert [5] who observed a power law degree sequence for a subgraph of the World Wide Web, and of Faloutsos, Faloutsos and Faloutsos [14] who observed power law behaviour for the internet graph. Many models of such process exist. For details see, for example, the surveys [7,27] and the monographs [9,12].

In networked information spaces, vertices are not only defined by their link environment, but also by the information entity they represent. More recently, attempts have been made to model this alternative view of the vertices through *spatial models*. In a spatial model, vertices are embedded in a metric space, and link formation is influenced by the metric distance between vertices. The metric space is meant to be like a feature space, so that the coordinates of a vertex in this space represent the information associated with the vertex. For example, in text mining, documents are commonly represented as vectors in a word space. The metric is chosen so that metric distance represents similarity, i.e. vertices whose information entities are closely related will be at a short distance from each other in the metric space. A number of spatial models have been proposed up to date [10,11,15,16,17,24]. We direct the reader to the recent survey for more details [18].

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 29–40, 2012.

In this paper, we focus on the Spatial Preferred Attachment (SPA) model, proposed in [3,4]. The SPA model generates directed graphs according to the following principle. Vertices are points in a given metric space. Each vertex v has a *sphere of influence*. The volume of the sphere of influence of a vertex is a function of its in-degree. A new vertex u can only link to an existing vertex v if u falls inside the sphere of influence of v. In the latter case, u links to v with probability p. The SPA model incorporates the principle of preferential attachment, since vertices with a higher in-degree will have a larger sphere of influence. We investigate the (directed) diameter, small separators, and the (weak) giant component of the model.

2 The SPA Model

We start by giving a precise description of the SPA model, presenting some known properties, and deriving some facts about the model, which we will need to prove our results. In [3] (see also [4] for a proceeding version of this paper), the model is defined for a variety of metric spaces S. In this paper, we let S be the unit hypercube in $\mathbb{R}^m$, equipped with the torus metric derived from any of the L_p norms. This means that for any two points x and y in S,

$$d(x,y) = \min\left\{||x-y+u||_p \,:\, u \in \{-1,0,1\}^m\right\}.$$

The torus metric thus "wraps around" the boundaries of the unit square; this metric was chosen to eliminate boundary effects.

The parameters of the model consist of the *link probability* $p \in [0,1]$, and two positive constants A_1 and A_2, which, in order to avoid the resulting graph becoming too dense, must be chosen so that $pA_1 < 1$. The original model as presented in [3] has a third parameter, A_3, which is assumed to be zero here. This causes no loss of generality, since all asymptotic results presented here are unaffected by A_3.

The SPA model generates stochastic sequences of graphs $(G_t : t \geq 0)$, where $G_t = (V_t, E_t)$, and $V_t \subseteq S$. Let $\deg^-(v,t)$ be the in-degree of vertex v in G_t, and $\deg^+(v,t)$ its out-degree. We define the *sphere of influence* $S(v,t)$ of vertex v at time $t \geq 1$ to be the ball centered at v with volume $|S(v,t)|$ defined as follows:

$$|S(v,t)| = \frac{A_1 \deg^-(v,t) + A_2}{t}, \tag{2.1}$$

or $S(v,t) = S$ and $|S(v,t)| = 1$ if the right-hand-side of (2.1) is greater than 1.

The process begins at $t = 0$, with G_0 being the null graph. Time-step t, $t \geq 1$, is defined to be the transition between G_{t-1} and G_t. At the beginning of each time-step t, a new vertex v_t is chosen *uniformly at random* from S, and added to V_{t-1} to create V_t. Next, independently, for each vertex $u \in V_{t-1}$ such that $v_t \in S(u, t-1)$, a directed link (v_t, u) is created with probability p. Thus, the probability that a link (v_t, u) is added in time-step t equals $p\,|S(u, t-1)|$.

We say that an event holds asymptotically almost surely (a.a.s.) if the probability that it holds tends to one as t goes to infinity. It was shown in [3] that

a.a.s. the SPA model produces graphs with a power law degree distribution, with exponent $1 + 1/(pA_1)$. Moreover, a precise expression for the probability distribution of the in-degree of the individual vertex v_i born at time i was given. In [19] (see also [20]) the relationship between the link structure of graphs produced by the model and the relative positions of the vertices in the metric space was analyzed. See Figure 1 for a drawing of a simulation of the SPA model.

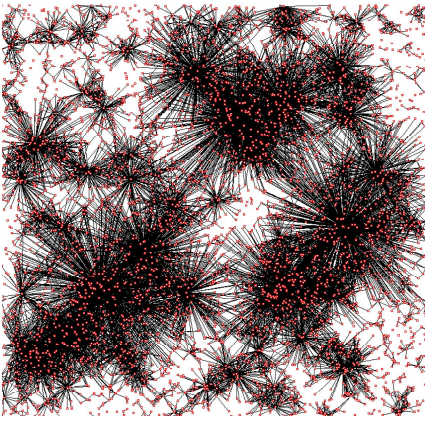


Fig. 1. A simulation on the unit square with $t = 5,000$ and $p = A_1 = A_2 = 1$

Now, let us discuss a few simple new facts about the model. Knowing the expected in-degree of a node, given its age, will help us to analyze geometric properties of the SPA Model. Let us note that the result for $i \gg 1$ was proved in [19] (see (2.2)); we extend it here to all $i \geq 1$ (see (2.3)). As before, let v_i be the node added at time i.

Theorem 1. *Suppose that $i = i(t) \gg 1$ as $t \to \infty$. Then,*

$$\begin{aligned} \mathbb{E}(\deg^-(v_i, t)) &= (1 + o(1))\frac{A_2}{A_1}\left(\frac{t}{i}\right)^{pA_1} - \frac{A_2}{A_1}, \\ \mathbb{E}(|S(v_i, t)|) &= (1 + o(1))A_2 t^{pA_1 - 1} i^{-pA_1}. \end{aligned} \tag{2.2}$$

Moreover, for all $i \geq 1$,

$$\begin{aligned} \mathbb{E}(\deg^-(v_i, t)) &\leq \frac{eA_2}{A_1}\left(\frac{t}{i}\right)^{pA_1} - \frac{A_2}{A_1}. \\ \mathbb{E}(|S(v_i, t)|) &\leq (1 + o(1))eA_2 t^{pA_1 - 1} i^{-pA_1}. \end{aligned} \tag{2.3}$$

Proof. In order to simplify calculations, we make the following substitution:

$$X(v_i, t) = \deg^-(v_i, t) + \frac{A_2}{A_1}. \tag{2.4}$$

It follows from the definition of the process that

$$X(v_i, t+1) = \begin{cases} X(v_i,t)+1, & \text{with probability } \frac{pA_1X(v_i,t)}{t} \\ X(v_i,t), & \text{otherwise.} \end{cases}$$

Finding the conditional expectation,

$$\begin{aligned} \mathbb{E}(X(v_i,t+1) \mid X(v_i,t)) &= (X(v_i,t)+1)\frac{pA_1X(v_i,t)}{t} + X(v_i,t)\left(1 - \frac{pA_1X(v_i,t)}{t}\right) \\ &= X(v_i,t)\left(1+\frac{pA_1}{t}\right). \end{aligned}$$

Taking expectations again, we get

$$\mathbb{E}(X(v_i,t+1)) = \mathbb{E}(X(v_i,t))\left(1+\frac{pA_1}{t}\right).$$

Since all nodes start with in-degree zero, $X(v_i,i) = \frac{A_2}{A_1}$. Note that, for $0 < x < 1$, $\log(1+x) = x - O(x^2)$. If $i \gg 1$, one can use this to get

$$\mathbb{E}(X(v_i,t)) = \frac{A_2}{A_1}\prod_{j=i}^{t-1}\left(1+\frac{pA_1}{j}\right) = (1+o(1))\frac{A_2}{A_1}\exp\left(\sum_{j=i}^{t-1}\frac{pA_1}{j}\right),$$

but in all cases $i \geq 1$,

$$\mathbb{E}(X(v_i,t)) \leq \frac{A_2}{A_1}\exp\left(\sum_{j=i}^{t-1}\frac{pA_1}{j}\right).$$

Therefore, when $i \gg 1$,

$$\mathbb{E}(X(v_i,t)) = (1+o(1))\frac{A_2}{A_1}\exp\left(pA_1\log\left(\frac{t}{i}\right)\right) = (1+o(1))\frac{A_2}{A_1}\left(\frac{t}{i}\right)^{pA_1},$$

and (2.2) follows from (2.4) and (2.1). Moreover, for any $i \geq 1$

$$\mathbb{E}(X(v_i,t)) \leq \frac{A_2}{A_1}\exp\left(pA_1\left(\log\left(\frac{t}{i}\right)+1/i\right)\right) \leq \frac{eA_2}{A_1}\left(\frac{t}{i}\right)^{pA_1},$$

and (2.3) follows from (2.4) and (2.1) as before which completes the proof.

Another fact that we will need follows directly from the following result proved in [19]. The degree of an individual vertex is not concentrated, due to variation happening shortly after birth. (That is, a.a.s. there are vertices that have smaller/larger degrees than what we would expect.) However, provided that the degree of the vertex at end time t is large enough (that is, is tending to infinity faster than $\log t$), sharp bounds on the degree of the vertex during most of the

process were obtained. This is expressed in the following theorem. First, define a injective function $f : \mathbb{R} \to \mathbb{R}$ by

$$f(i) = \frac{A_2}{A_1}\left(\frac{t}{i}\right)^{pA_1},$$

so $f(i)$ is the expected degree, at time t, of a vertex born at time i (up to a factor of $(1 + o(1))$). Thus, $f^{-1}(k)$ is the birth time of a vertex of final degree k, assuming the degree of the vertex is close to the expected value during its lifetime. Hence, if a vertex of final degree k has behaviour close to its expected degree, then

$$t_a = f^{-1}\left(\frac{A_2 k}{A_1 a}\right)$$

will be the time when that vertex has degree a. Indeed, for a vertex born at time $f^{-1}(k)$, the expected degree at time t_a is equal to

$$\begin{aligned}\frac{A_2}{A_1}\left(\frac{t_a}{f^{-1}(k)}\right)^{pA_1} &= \frac{A_2}{A_1}\left(\frac{A_2}{A_1}\left(\frac{t}{f^{-1}(k)}\right)^{pA_1}\right) / \left(\frac{A_2}{A_1}\left(\frac{t}{t_a}\right)^{pA_1}\right)\\ &= \frac{A_2}{A_1}k / \left(\frac{A_2 k}{A_1 a}\right) = a.\end{aligned}$$

Theorem 2 ([19]). *Let $\omega = \omega(t)$ be any function tending to infinity together with t. The following statement holds a.a.s. for every vertex v for which $\deg^-(v,t) = k = k(t) \geq \omega \log t$. Let $i = f^{-1}(k)$, and let t_k be*

$$t_k = f^{-1}\left(\frac{A_2 k}{A_1 \omega \log t}\right).$$

Then, for all values of s such that $t_k \leq s \leq t$,

$$\deg^-(v,s) = (1 + o(1))\frac{A_2}{A_1}\left(\frac{s}{i}\right)^{pA_1} = (1 + o(1))k\left(\frac{s}{t}\right)^{pA_1}. \tag{2.5}$$

The theorem implies that once a given vertex accumulates $\omega \log t$ neighbours, the rest of the process (until time-step t) can be predicted with high probability; in fact, a.a.s. we get a concentration around the expected value.

Now, with Theorem 2 in hand, we get immediately the following.

Theorem 3. *Let $\omega = \omega(t)$ is a function that goes to infinity together with t. The following holds a.a.s. for every vertex v_i added at time i. For all $i \leq s \leq t$ we have*

$$\begin{aligned}\deg^-(v_i,s) &= O\left((\omega \log t)\left(\frac{s}{i}\right)^{pA_1}\right),\\ |S(v_i,s)| &= O\left(\frac{\omega \log t}{i}\right),\end{aligned}$$

Proof. For a contradiction suppose that $k = \deg^-(v_i, s) \geq (2\omega \log t)(s/i)^{pA_1}$ for some value of s $(i \leq s \leq t)$. Since $k \geq \omega \log t$, Theorem 2 can be applied to get that

$$\deg^-(v_i, i) = (1 + o(1))\frac{A_2}{A_1}\left(\frac{i}{f^{-1}(k)}\right)^{pA_1} = (1+o(1))\frac{A_2}{A_1}\left(\frac{s}{f^{-1}(k)}\right)^{pA_1}\left(\frac{s}{i}\right)^{-pA_1}$$
$$= (1 + o(1))k\left(\frac{s}{i}\right)^{-pA_1} \geq (2 + o(1))\omega \log t,$$

which is clearly a contradiction (in fact, $\deg^-(v_i, i) = 0$).

3 Directed Diameter

The small world property, introduced by Watts and Strogatz [29], is a central notion in the study of complex networks (see also [22]). The small world property demands a low diameter of $O(\log t)$, and a higher clustering coefficient than found in a binomial random graph with the same number of nodes and same average degree. Adamic et al. [1] provided an early study of a social network at Stanford University, and found that the network has the small world property. Similar results were found in [2] which studied Cyworld, MySpace, and Orkut, and in [26] which examined data collected from Flickr, YouTube, LiveJournal, and Orkut. Low diameter (of 6) and high clustering coefficient were reported in the Twitter by both Java et al. [21] and Kwak et al. [23]. Many well-known models for complex networks, including the preferential attachment model by Barabási and Albert [5], have diameters growing at most logarithmically with time. (In fact, in [8] Bollobás and Riordan showed that a.a.s. the diameter of the preferential attachment model is asymptotic to $\log t / \log \log t$.)

Consider a graph G_t produced by the SPA model. For a given pair of vertices $v_i, v_j \in V_t$ $(1 \leq i < j \leq t)$, let $l(v_i, v_j)$ denote the length of the shortest directed path from v_j to v_i if such a path exists, and let $l(v_i, v_j) = 0$ otherwise. The directed diameter of a graph G_t is defined as

$$D(G_t) = \max_{1 \leq i < j \leq t} l(v_i, v_j).$$

The next subsection (Subsection 3.1) is devoted to proving the following result:

Theorem 4. *A.a.s.* $D(G_t) = O(\log t)$.

In fact, we conjecture that this result is best possible; that is, the following holds:

Conjecture 1. A.a.s. $D(G_t) = \Theta(\log t)$.

We will try to settle this down in the journal version of this paper. We mention the approach we plan to use to solve it in the Subsection 3.2.

3.1 Upper Bound

An $O(\log t)$ upper bound on the directed diameter is obtained as follows.

Theorem 5. *Let $C = 18\max(A_2, 1)$. With probability $1 - o(t^{-2})$ we have that for any $1 \le i < j \le t$, G_t does not contain a directed (v_i, v_j)-path of length at least $k^* = C \log t$.*

As there are at most t^2 pairs v_i, v_j, the Theorem 4 will follow as well.

Proof. In order to simplify the notation, we use v to denote the vertex added at step $v \le t$. Let vPu be a directed (v,u)-path of length given by $vPu = (v, t_{k-1}, t_{k-2}, \dots, t_1, u)$, let $t_0 = u, t_k = v$.

$$\Pr(vPu) = \prod_{i=1}^{k} p\left(\frac{A_1 \deg^-(t_{i-1}, t_i) + A_2}{t_i}\right).$$

Let $N(v,u,k)$ be the number of directed (v,u)-paths of length k, then

$$\mathbb{E}N(v,u,k) = \sum_{u<t_1<\cdots<t_{k-1}<v} p^k \mathbb{E}\left(\prod_{i=1}^{k}\left(\frac{A_1 \deg^-(t_{i-1}, t_i) + A_2}{t_i}\right)\right).$$

However

$$\mathbb{E}(\deg^-(t_i, t_{i+1}) \mid \deg^-(t_{j-1}, t_j) \text{ and } (t_{j-1}, t_j) \in E_t,\, j \le i) = \mathbb{E}(\deg^-(t_i, t_{i+1})).$$

We first consider the case where u tends to infinity together with t. From Theorem 1 it follows that

$$\mathbb{E}(\deg^-(t_{i-1}, t_i)) = (1 + o(1))\frac{A_2}{A_1}\left(\frac{t_i}{t_{i-1}}\right)^{pA_1} - \frac{A_2}{A_1}.$$

Thus

$$\begin{aligned}
\mathbb{E}N(v,u,k) &= \sum_{u<t_1<\cdots<t_{k-1}<v} p^k \prod_{i=1}^{k}\frac{1}{t_i}\left(A_1\mathbb{E}(d^-(t_{i-1}, t_i)) + A_2\right) \\
&= \sum_{u<t_1<\cdots<t_{k-1}<v} (1 + o(1))^k (A_2 p)^k \prod_{i=1}^{k}\frac{1}{t_i}\left(\frac{t_i}{t_{i-1}}\right)^{pA_1} \\
&= (1 + o(1))^k (A_2 p)^k \left(\frac{v}{u}\right)^{pA_1}\frac{1}{v}\sum_{u<t_1<\cdots<t_{k-1}<v}\prod_{i=1}^{k-1}\frac{1}{t_i}.
\end{aligned}$$

However

$$\begin{aligned}
\sum_{u<t_1<\cdots<t_{k-1}<v}\prod_{i=1}^{k-1}\frac{1}{t_i} &\le \frac{1}{(k-1)!}\left(\sum_{u<s<v}\frac{1}{s}\right)^{k-1} \\
&\le \frac{1}{(k-1)!}\left(\log v/u + 1/u\right)^{k-1} \\
&\le \left(\frac{e(\log v/u + 1/u)}{k-1}\right)^{k-1}.
\end{aligned}$$

Let $k^* = C \log t$, where $C = 18 \max(1, A_2)$. Assuming t sufficiently large, and recalling that $pA_1 < 1$, we have

$$\begin{aligned}\sum_{k>k^*} \mathbb{E}N(v,u,k) &\le 2A_2 \sum_{k>k^*} \left(\frac{(1+o(1))A_2 pe(\log v/u + 1/u)}{k-1} \right)^{k-1} \\ &\le 2A_2 \left(\frac{(1+o(1))A_2 e(\log v/u + 1/u)}{C \log t} \right)^{k^*} \frac{1}{1 - 3A_2/C} \\ &= O(6^{-18 \log t}) = o(t^{-4}).\end{aligned}$$

The result follows for u tending to infinity. In the case where u is a constant, it follows from Theorem 1 that a multiplicative correction of e can be used in $\mathbb{E}(\deg^-(t_{i-1}, t_i))$, leading to an error term of $O(t^{-18 \log 2}) = o(t^{-4})$, as before.

3.2 Lower Bound

It follows from Theorem 1 that a vertex v_i added at time $i \gg 1$ has the expected degree of 1 at time

$$t_i = (1 + o(1)) \left(\frac{A_1 + A_2}{A_2} \right)^{1/pA_1} i = \Theta(i).$$

(Note that the constant in the $\Theta()$ notation does not depend on i.) Hence, the path constructed by considering the first neighbours only is expected to have a logarithmic length. However, such a path is not necessarily the shortest path. In order to show that a path of the desired length that is also the shortest one exists a.a.s., we plan to create a path that does not extend by joining to the first in-neighbour of v_i but instead we wait for a neighbour that is both far away from v_i and is in the desired direction. Since the spheres of influences are usually shrinking, this should be enough to guarantee that no 'shortcut' in this path can be created.

4 Small Separators

Let us note that there are some significant differences between graphs generated by the preferential attachment model and those found in the real world. One major difference is found in their expansion properties. Mihail, Papadimitriou, and Saberi [25] showed that a.a.s. the preferential attachment model has conductance bounded below by a constant. On the other hand, Blandford, Blelloch and Kash [6] found that some WWW related graphs have smaller separators than the preferential attachment model predicts. This observation is consistent with observations due to Estrada [13], who found that half of the real-world networks he looked at were good expanders and the other half were not so good. In this subsection, we show that the SPA model has small separators.

Let us recall that $V_t \subseteq S$ where S is the unit hypercube $[0,1]^m$. We use the geometry of the model to obtain a sparse cut. Let

$$S' = \left\{ s = (s_1, s_2, \ldots, s_m) \in S : s_1 < \frac{1}{2} \right\}.$$

Let us partition the vertex set V_t as follows: $V_t' = V_t \cap S'$, $V_t'' = V_t \cap (S \setminus S') = V_t \setminus V_t'$. The next theorem shows that this partition yields a sparse cut.

Theorem 6. *A.a.s. the following holds* $|V_t'| = (1+o(1))t/2$, $|V_t''| = (1+o(1))t/2$, *and*

$$|E(V_t', V_t'')| = O(t^{\max\{1-1/m, pA_1\}} \log^5 t) = o(t).$$

Proof. Clearly, we expect $t/2$ vertices in each set V_t' and V_t''. The concentration follows immediately from Chernoff bound. It remains to show that an upper bound for the size of the cut holds a.a.s.

It follows from Theorem 3 (by taking $\omega = \log t$) that a.a.s. for every $i \in [t]$ the maximum sphere of influence of a vertex v_i added at time i is $O(i^{-1} \log^2 t)$ (during the whole process). Since we aim for a result that holds a.a.s., we may assume that this property holds for all i. Therefore, the maximum radius of influence of v_i is $O((\log^2 t/i)^{1/m})$.

We will investigate how many edges are in the cut by counting (independently) edges in this cut directed to vertices of similar age. For a given integer k such that $0 \le k < \log t$, let

$$\begin{aligned} V^{(k)} &= \{v_i \in V_t : e^k \le i < \min\{e^{k+1}, t\}\}, \\ E^{(k)} &= \{(v_i, v_j) \in E_t : v_i \in V^{(k)} \quad \text{and} \quad i < j \le t\} \\ C^{(k)} &= E^{(k)} \cap E(V_t', V_t''). \end{aligned}$$

It is clear that $\{E^{(k)} : 0 \le k < \log t\}$ is a partition of the edge set and so $\{C^{(k)} : 0 \le k < \log t\}$ is a partition of the cut $E(V_t', V_t'')$. It remains to estimate the size of $C^{(k)}$ for a given value of k.

Fix $0 \le k < \log t$, and let $v_i \in V^{(k)}$. Note that the maximum radius of influence of v_i is $O((e^{-k} \log^2 t)^{1/m})$. Therefore, if there is an edge in the cut directed to $v_i = (s_1, s_2, \ldots, s_m)$, then v_i must fall into a strip within distance $O((e^{-k} \log^2 t)^{1/m})$ from the cutting hyperplane; that is, $|s_1 - 1/2| = O((e^{-k} \log^2 t)^{1/m})$. Since $|V^{(k)}| = O(e^k)$, we get that

$$O((e^{-k} \log^2 t)^{1/m}) \cdot |V^{(k)}| = O(e^{k(1-1/m)} (\log t)^{2/m})$$

vertices of $V^{(k)}$ are expected to appear in this strip during the whole process. Hence, it follows from Chernoff bound that with probability at least $1 - \exp(-\Theta(\log^2 t))$ there are $O(e^{k(1-1/m)} \log^2 t)$ vertices in this strip at the end of the process. (Note that the exponent of $\log t$ has changed from $2/m$ to 2 in order to guarantee the value at least $\log^2 t$ which is required for a bound to hold with the desired probability.) By Theorem 3 (again, by taking $\omega = \log t$),

a.a.s. all vertices introduced in this time period have (final) in-degree at most $(t/e^k)^{pA_1}\log^2 t$, we get that

$$|C^{(k)}| = O(e^{k(1-1/m)}\log^2 t)\cdot(t/e^k)^{pA_1}\log^2 t = O(t^{pA_1}e^{k(1-1/m-pA_1)}\log^4 t)$$

edges in the cut a.a.s.

Finally, we get that a.a.s.

$$|E(V_t', V_t'')| = \sum_{k=0}^{\lceil \log t\rceil -1} |C^{(k)}| = \sum_{k=0}^{\lceil \log t\rceil -1} O(t^{pA_1}e^{k(1-1/m-pA_1)}\log^4 t)$$
$$\leq \begin{cases} \log t\cdot O(t^{pA_1}t^{1-1/m-pA_1}\log^4 t) = O(t^{1-1/m}\log^5 t), & \text{if } pA_1 < 1-1/m;\\ \log t\cdot O(t^{pA_1}\log^4 t) = O(t^{pA_1}\log^5 t), & \text{otherwise,}\end{cases}$$

which finishes the proof.

5 Emergence of Giant Component

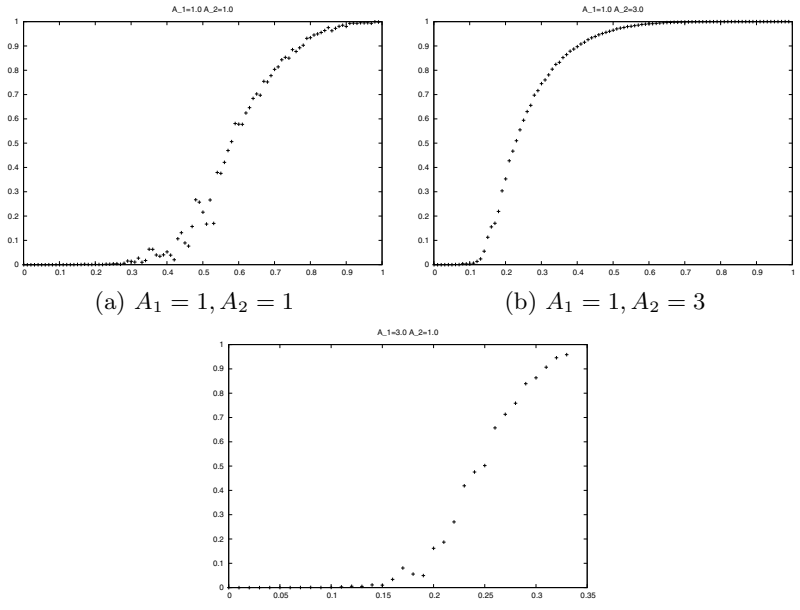
Let us note that all edges in G_t are from younger vertices to older ones; that is, denoting by v_i the vertex added at time i we get that if $(v_j, v_i) \in E_t$, then $j > i$. This implies that G_t has t strongly connected components, each of which consists of one vertex.

On the other hand, it seems that investigating the size of the largest weak connected component is a non-trivial task. Let $\hat{G}_t = (V_t, \hat{E}_t)$ be the underlying graph of G_t; that is, $\hat{G}$ is an undirected graph on the vertex set V_t and $\{v_j, v_i\} \in \hat{E}_t$ if and only if $(v_j, v_i) \in E_t$. We wish to know the size of the largest component in $\hat{G}_t$.

One can show that the expected number of edges added at time t of the process is $\deg^+(v_t, t) = \frac{pA_2}{1-pA_1}$. Therefore, if $p > p_1 := (A_1 + A_2)^{-1}$, then the expected out degree in G_t is larger than 1, and so is the expected degree in $\hat{G}_t$. By looking at the 'branching factor' of the breadth-first search process it is natural to conjecture that a.a.s. there exists a giant component if $p > p_1$. On the other hand, if $p < p_1$, then the expected out-degree in G_t is smaller than one, but this fact does not in itself guarantee absence of the giant component in $\hat{G}_t$. Is p_1 the threshold we search for? If $p < p_2 := (A_1 + 2A_2)^{-1}$, then $\deg^+(v_t, t) < 1/2$ and so the average degree in $\hat{G}_t$ is smaller than one. Perhaps p_2 is the threshold for the giant component? Clearly, more sophisticated argument is required to solve this problem and we will try to settle this down in the journal version of this paper.

We performed a number of simulations to make a better prediction (see Figure 2). For a given set of parameters A_1, A_2, we performed a number of simulations ($p = i/100, 0 \leq i < 1/A_1$). Unfortunately, it seems that $t = 100,000$ is still too small to observe a clear trend. However, based on these numerical results, one can conjecture the following.

Conjecture 2. $p_3 := (2A_1 + 2A_2)^{-1}$ is the threshold for the giant component.



(a) $A_1 = 1, A_2 = 1$ (b) $A_1 = 1, A_2 = 3$

(c) $A_1 = 3, A_2 = 1$

Fig. 2. A simulation of the SPA model on the unit 2-dimensional torus with $t = 100,000$. (The x-axis is p, y-axis is the fraction of vertices in the largest component of $\hat{G}_t$.)

References

1. Adamic, L.A., Buyukkokten, O., Adar, E.: A social network caught in the web. First Monday 8 (2003)
2. Ahn, Y., Han, S., Kwak, H., Moon, S., Jeong, H.: Analysis of topological characteristics of huge on-line social networking services. In: Proceedings of the 16th International Conference on World Wide Web (2007)
3. Aiello, W., Bonato, A., Cooper, C., Janssen, J., Prałat, P.: A spatial web graph model with local influence regions. Internet Mathematics 5, 175–196 (2009)
4. Aiello, W., Bonato, A., Cooper, C., Janssen, J., Prałat, P.: A Spatial Web Graph Model with Local Influence Regions. In: Bonato, A., Chung, F.R.K. (eds.) WAW 2007. LNCS, vol. 4863, pp. 96–107. Springer, Heidelberg (2007)
5. Barabási, A., Albert, R.: Emergence of scaling in random networks. Science 286, 509–512 (1999)
6. Blandford, D., Blelloch, G.E., Kash, I.: Compact Representations of Separable Graphs. In: Proc. of ACM/SIAM Symposium on Discrete Algorithms, pp. 679–688 (2003)
7. Bollobás, B., Riordan, O.: Mathematical results on scale-free graphs. In: Bornholdt, S., Schuster, H. (eds.) Handbook of Graphs and Networks. Wiley-VCH, Berlin (2002)
8. Bollobás, B., Riordan, O.: The diameter of a scale-free random graph. Combinatorica 4, 5–34 (2004)

9. Bonato, A.: A course on the web graph. American Mathematical Society Graduate Studies in Mathematics 89 (2008)
10. Bonato, A., Janssen, J., Prałat, P.: Geometric Protean Graphs. Internet Mathematics 8, 2–28 (2012)
11. Bradonjic, M., Hagberg, A., Percus, A.G.: The structure of geographical threshold graphs. Internet Mathematics 4, 113–139 (2009)
12. Chung, F.R.K., Lu, L.: Complex Graphs and Networks. American Mathematical Society (2006)
13. Estrada, E.: Spectral scaling and good expansion properties in complex networks. Europhysics Letters 73(4), 649–655 (2006)
14. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On Power-law Relationships of the Internet Topology. In: SIGCOMM, pp. 251–262 (1999)
15. Flaxman, A., Frieze, A.M., Vera, J.: A geometric preferential attachment model of networks. Internet Mathematics 3(2), 187–206 (2006)
16. Flaxman, A., Frieze, A.M., Vera, J.: A geometric preferential attachment model of networks II. Internet Mathematics 4(1), 87–111 (2008)
17. Higham, D.J., Rasajski, M., Przulj, N.: Fitting a geometric graph to a protein-protein interaction network. Bioinformatics 24(8), 1093–1099 (2008)
18. Janssen, J.: Spatial Models for Virtual Networks. In: Ferreira, F., Löwe, B., Mayordomo, E., Mendes Gomes, L. (eds.) CiE 2010. LNCS, vol. 6158, pp. 201–210. Springer, Heidelberg (2010)
19. Janssen, J., Prałat, P., Wilson, R.: Geometric Graph Properties of the Spatial Preferred Attachment model (preprint)
20. Janssen, J., Prałat, P., Wilson, R.: Estimating node similarity from co-citation in a spatial graph model. In: Proceedings of the 2010 ACM Symposium on Applied Computing–Special Track on Self-organizing Complex Systems, pp. 1329–1333 (2010)
21. Java, A., Song, X., Finin, T., Tseng, B.: Why we twitter: understanding microblogging usage and communities. In: Proceedings of the Joint 9th WEBKDD and 1st SNA-KDD Workshop 2007 (2007)
22. Kleinberg, J.: The small-world phenomenon: An algorithmic perspective. In: Proceedings of the 32nd ACM Symposium on Theory of Computing (2000)
23. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: Proceedings of the 19th International World Wide Web Conference (2010)
24. Masuda, N., Miwa, M., Konno, N.: Geographical threshold graphs with small-world and scale-free properties. Phys. Rev. E 71(3), 036108 (2005)
25. Mihail, M., Papadimitriou, C.H., Saberi, A.: On Certain Connectivity Properties of the Internet Topology. In: Proc. IEEE Symposium on Foundations of Computer Science, p. 28 (2003)
26. Mislove, A., Marcon, M., Gummadi, K., Druschel, P., Bhattacharjee, B.: Measurement and analysis of on-line social networks. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (2007)
27. Mitzenmacher, M.: A brief history of generative models for power law and lognormal distributions. In: Proc. of the 39th Annual Allerton Conf. on Communication, Control, and Computing, pp. 182–191 (2001)
28. Yule, G.: A mathematical theory of evolution based on the conclusions of Dr. J.C. Willis. Philosophical Transactions of the Royal Society of London (Series B) 213, 21–87 (1924)
29. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)

A Sublinear Time Algorithm for PageRank Computations

Christian Borgs[1], Michael Brautbar[2], Jennifer Chayes[1], and Shang-Hua Teng[3]

[1] Microsoft Research New England, One Memorial Drive, Cambridge, MA 02142
{borgs,jchayes}@microsoft.com
[2] Computer and Information Science Department, University of Pennsylvania, 3330 Walnut Street, Philadelphia, PA 19104
brautbar@cis.upenn.edu
[3] Computer Science Department, University of Southern California, 941 Bloom Walk, Los Angeles, CA 90089
shanghua@usc.edu

Abstract. In a network, identifying all vertices whose PageRank is more than a given threshold value Δ is a basic problem that has arisen in Web and social network analyses. In this paper, we develop a nearly optimal, sublinear time, randomized algorithm for a close variant of this problem. When given a directed network $G = (V, E)$, a threshold value Δ, and a positive constant $c > 3$, with probability $1 - o(1)$, our algorithm will return a subset $S \subseteq V$ with the property that S contains all vertices of PageRank at least Δ and no vertex with PageRank less than Δ/c. The running time of our algorithm is always $\tilde{O}(\frac{n}{\Delta})$. In addition, our algorithm can be efficiently implemented in various network access models including the Jump and Crawl query model recently studied by [6], making it suitable for dealing with large social and information networks.

As part of our analysis, we show that any algorithm for solving this problem must have expected time complexity of $\Omega(\frac{n}{\Delta})$. Thus, our algorithm is optimal up to logarithmic factors. Our algorithm (for identifying vertices with significant PageRank) applies a multi-scale sampling scheme that uses a fast personalized PageRank estimator as its main subroutine. For that, we develop a new local randomized algorithm for approximating personalized PageRank which is more robust than the earlier ones developed by Jeh and Widom [9] and by Andersen, Chung, and Lang [2].

1 Introduction

A basic problem in network analysis is to identify the set of its vertices that are "significant." For example, the significant nodes in the web graph defined by a query could provide the authoritative content in web search; they could be the critical proteins in a protein interaction network; and they could be the set of people (in a social network) most effective to seed the influence for online advertising. As the networks become larger, we need more efficient algorithms to identify these "significant" nodes.

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 41–53, 2012.

1.1 Identifying Nodes with Significant PageRanks: Our Results

The meaning of 'significant' vertices depend on the semantics of the network and the applications. In this paper, we focus on a particular measure of significance — the PageRanks of the vertices. PageRank was introduced by Page and Brin in their seminal work for ranking webpages [11]. Mathematically, the PageRank (with restart constant, also known as the teleportation constant, α) of a web-page is proportional to the the probability that the page is visited by a random surfer who explores the web using the following simple random walk: at each step, with probability $(1-\alpha)$ go to a random webpage linked to from the current page, and with probability α, restarts the process from a randomly chosen page. For reasons to be cleared shortly, we consider a normalization of the PageRank so that the sum of the PageRank values over all vertices is equal to n, the number of vertices in the network. In other words, suppose PageRank(u) denote the PageRank of vertex u in the network $G = (V, E)$. Then,

$$\sum_{u \in V} \text{PageRank}(u) = n.$$

PageRank has been used by the Google search engine and has found applications in wide range of data analysis problems [4,7]. In this context, the problem of identifying "significant" vertices could be illustrated by the following search problem: Let Top PageRanks denote the problem of identifying all vertices whose PageRanks in a network $G = (V, E)$ are more than a given threshold value $1 \leq \Delta \leq |V|$.

In this paper, we consider for the following close variant of Top PageRanks:

> Significant PageRanks**:** *Given a network $G = (V, E)$, a threshold value $1 \leq \Delta \leq |V|$ and a positive constant $c > 1$, compute, with success probability $1 - o(1)$, a subset $S \subseteq V$ with the property that S contains all vertices of PageRank at least Δ and no vertex with PageRank less than Δ/c.*

We develop a nearly optimal, sublinear time randomized algorithm for Significant PageRanks for any fixed $c > 3$. The running time of our algorithm is always $\tilde{O}(\frac{n}{\Delta})$. We show that any algorithm for Significant PageRanks must have time complexity of $\Omega(\frac{n}{\Delta})$. Thus, our algorithm is optimal up to logarithmic factors. Our Significant PageRanks algorithm applies a multi-scale sampling scheme that uses a fast personalized PageRank estimator (see below) as its main subroutine.

1.2 Matrix Sampling and Personalized PageRank Approximation

While the PageRank of a vertex captures the importance of the vertex collectively assigned by all vertices in the network, as pointed out by Haveliwala [8], one can use the distributions of the following random walk to define the pairwise contributions of significances: Given a teleportation probability α and a starting

vertex u in a network $G = (V, E)$, at each step, with probability $(1 - \alpha)$ go to a random neighboring vertex, and with probability α, restarts the process from u. For $v \in V$, the probability that v is visited by this random process, denoted by $\text{PersonalizedPageRank}_u(v)$, is the u's personal PageRank contribution of significance to v. It is not hard to verify that

$$\forall u \in V, \quad \sum_{v \in V} \text{PersonalizedPageRank}_u(v) = 1; \text{ and}$$

$$\forall v \in V, \quad \text{PageRank}(v) = \sum_{u \in V} \text{PersonalizedPageRank}_u(v).$$

Personalized PageRanks has been widely used to describe personalized behavior of web-users [11] as well as for designing good network clustering techniques [2]. As a result, fast algorithms for computing or approximating personalized PageRank can be very useful. One can approximate PageRanks and personalized PageRanks by the power method [4], which involves costly matrix multiplications for large scale networks. Applying effective truncation, Jeh and Widom [9] and Andersen, Chung, and Lang [2] developed personalized PageRank approximation algorithms that can find an ϵ-additive approximation in time proportional to the product of ϵ^{-1} and the maximum in-degree in the graph.

Our sublinear-time algorithm for Significant PageRanks also requires fast subroutines for estimating personalized PageRanks. It uses a multi-scale sampling approach by selecting a set of precision parameters $\{\epsilon_1, ..., \epsilon_h\}$ where h depends on n and Δ, $\epsilon_i = 1/2^i$. Then, for each i in range $1 \leq i \leq h$, it computes the ϵ_i-precise personalized PageRanks defined by a sample of $\tilde{O}(\epsilon_i n/\Delta)$ vertices. For networks with constant maximum degrees, we can simply use the Jeh-Widom or Andersen-Chung-Lang personalized PageRank approximation algorithms in our multi-scale sampling scheme. However, for networks such as web graphs and social networks that may have nodes with large degrees, these two earlier algorithms are not robust enough for our purpose.

We develop a new local algorithm for approximating personalized PageRank that satisfies the desirable robust property that the multi-scale sample scheme requires. Given $\rho, \epsilon > 0$ and a starting vertex u in a network $G = (V, E)$, our algorithm estimates each entry in the personalized PageRank vector,

$$\text{PersonalizedPageRank}_u(.)$$

defined by u to a multiplicative factor of at most $(1+\rho)$ plus an additive precision error of at most ϵ [1]. The time complexity of our algorithm is $O(\frac{\log(|V|)\log(\epsilon^{-1})}{\epsilon\rho^2})$. Our algorithm requires a careful simulation of random walks from the starting node u to ensure that its complexity does not depends on the degree of any node.

Our algorithms can be efficiently implemented in various network querying models assuming no direct global access to the network. In particular, our algorithms can be efficiently implemented in the Jump and Crawl query model [6],

[1] Formally, estimated value $\hat{val}$ of val would have the property that $(1-\rho) \cdot \text{val} - \epsilon \leq \hat{\text{val}} \leq (1+\rho) \cdot \text{val} + \epsilon$.

making the algorithm suitable for processing large social and information networks.

In particular, our sublinear algorithm for SIGNIFICANT PAGERANKS could be used in Web search engines, which often need to build a core of web-pages to be later used for web-search. It is desirable that pages in the core have high PageRank values. These search engines usually apply crawling to discover new significant pages and add them to the core. The property that our sublinear-time algorithms have a natural implementation in the Jump and Crawl model may make them useful in a search engine for selecting pages with high PageRank values to update the current core by using them to replace the existing core pages that have relatively low PageRank values. We anticipate that our algorithm for SIGNIFICANT PAGERANKS will be useful for many other network analysis tasks.

1.3 Additional Related Work

For personalized PageRanks approximation, in addition to the work of [2, 4, 5, 9, 11], Andersen *et al* [1] developed a 'backward' version of the local algorithm of [2]. Their algorithm finds all nodes that contribute at least some fixed fraction ρ to a page's PageRank in time $O(d_{\text{max-out}}/\rho)$ where $d_{\text{max-out}}$ is the maximum out-degree in the network. This algorithm can be used to provide some reliable estimate to a node's PageRank. For example, for a given k, in time $\tilde{\Theta}(k)$ it can bound the total contribution from the k highest contributors to the node's PageRank. However, for networks with large $d_{\text{max-out}}$, its complexity may not be sublinear.

As suggested in [1], one can view the entire set of personalized PageRanks (defined by all vertices in a network) as an $|V| \times |V|$ matrix, which is referred to as the *PageRank matrix* of the graph. In the PageRank matrix, each row represents the personalized PageRanks from a particular vertex, and each column represents the contributions to its PageRank from all vertices in the network. Note that the sum of each row is 1 and the sum of the u^{th} column is the PageRank of u.

In light of this, the problem of SIGNIFICANT PAGERANKS can be viewed as a matrix sparsification or matrix approximation problem. There has been a large body of work of finding a low complexity approximation to a matrix that preserves some of its properties. Perhaps the most relevant one to our goal is a low rank matrix approximation under the l_2 matrix norm.

All current methods for finding such low rank approximations runs in time at least linear in the size of the input matrix. See [10] for a survey of recent results.

Next, a linear time Monte Carlo based method to estimate PageRank of all nodes is devised in [3]. The method is based on running constant number of random walks from each of the nodes in the network.

Last, in the context of sublinear time graph algorithms, our research is related to the work of [12], in which sublinear time algorithms are presented for estimating several quantities. Our implementation of the Jump and Crawl query model can be viewed as a stringent type of the adjacency-list graph model used in [12].

1.4 Organization

Section 2 contains the needed definitions and notations. Section 3 presents our multi-scale sampling algorithm for SIGNIFICANT PAGERANKS. In section 4 we provide a lower bound construction for SIGNIFICANT PAGERANKS. In Section 5 we give a robust local algorithm for approximating personalized PageRank vectors.

2 Preliminaries

We consider a network which is defined as a directed graph $G = (V, E)$ with n nodes and m edges. Usually, a network is massive. Our algorithms access a network using a rather natural implementation of the Jump-and-Crawl query model of [6] developed for processing large social and information networks. The Jump and Crawl model is concerned with informational complexity of nodes, where each node access reveal its full list of adjacent neighbors at no extra cost. Our algorithms shall be designed to work under the following compelling implementation of the Jump-and-Crawl query model. We allow two types of queries:

- *Jump*: A call to the Jump query needs no input and returns a uniformly at random node from the network.
- *RandomCrawl*: A call to the RandomCrawl query requires a vertex v as input. RandomCrawl(v) returns a uniformly at random out-neighbor of v.

Note for example, that the random surfer procedure used in the definition of PageRank is itself a natural algorithm under our implementation of the Jump-and-Crawl query model.

We now move to define personalized PageRank as well as PageRank. Mathematically, the personalized PageRank vector of a node v is the stationary point of the following equation:

$$\text{PersonalizedPageRank}_v(\cdot) = \alpha \mathbf{1}_v + (1-\alpha)\text{PersonalizedPageRank}_v(\cdot) \cdot D^{-1}A,$$

where α is the teleportation probability, A is the adjacency matrix of the directed network $G = (V, E)$ so $A(i, j) = 1$ iff $(i, j) \in E$. In this notation, D is a diagonal matrix with $d_{out}(v)$ at entry (v, v) and $\mathbf{1}_v$ is the indicator vector of v. We will follow the standard [4] by assuming that each node has at least one out-link[2].

Then, one can define the RageRank vector as

$$\text{PageRank}(\cdot) = \sum_{v \in V} \text{PersonalizedPageRank}_v(\cdot)$$

Note that in this definition, the sum of the all PageRank values is equal to n.

Following [1], we define a matrix PPR (short for personalized PageRank) to be the $n \times n$ matrix, whose v^{th} row is $\text{PersonalizedPageRank}_v(\cdot)$.

Unless stated otherwise, for any x, $\log(x)$ would mean $\log_2(x)$.

[2] Otherwise, as commonly done [4], consider that node as having out links into all nodes in the network.

3 Multi-scale Matrix Sampling and Approximation of PageRank

In this section, we present our nearly optimal, sublinear time algorithm for Significant PageRanks. Recall that

> Significant PageRanks: *Given a network $G = (V, E)$, a threshold value $1 \leq \Delta \leq |V|$ and a positive constant $c > 1$, compute, with probability $1 - o(1)$, a subset $S \subseteq V$ with the property that S contains all vertices of PageRank at least Δ and no vertex with PageRank less than Δ/c.*

Note that the PageRank value of each vertex is at least α and at most n. Instrumental to our algorithm, we present a multi-scale algorithm for sampling the PageRank matrix PPR that achieves, for any fixed $c > 3$, the following goals: The algorithm makes $\tilde{O}(\frac{n}{\Delta})$ total queries and updates, and with high probability,

1. For each vertex with PageRank value at least Δ, the sum of the sampled entries of of the column corresponding to the vertex will provide a quality estimate to the PageRank value of that vertex.
2. The algorithm does not return any vertex whose PageRank value is less than Δ/c.

In our algorithm, we will use a new local algorithm *ApproxRow* for personalized PageRank approximation. Algorithm *ApproxRow* takes three input parameters: $v \in V$, an additive error factor $\epsilon \in (0, 1)$ and a multiplicative factor $\rho \in (0, 1)$. It returns an approximation to $\text{PersonalizedPageRank}_v(\cdot)$ such that for every $\text{PPR}(v, j) > \epsilon$, it returns a non-negative estimated value between $(1 - \rho)\text{PPR}(v, j) - \epsilon$ to $(1 + \rho)\text{PPR}(v, j) + \epsilon$. The running time of *ApproxRow* is essentially $O(\frac{\log(n)\log(\epsilon^{-1})}{\epsilon\rho^2})$. *ApproxRow* and its analysis will be presented in Section 5.

We start with some high-level idea of our multi-scale sampling algorithm. To assist our exposition, we will present our algorithm and its analysis for $c = 6$. Both are easily extended to any other constant value $c > 3$. Our algorithm will use $O(\log n)$ precision scales: $\epsilon_t = 2^{-t}$ for $0 \leq t \leq \log(\frac{4n}{\Delta})$. We conceptually divide each column of the PPR matrix into *chunks*, where the chunk corresponding to ϵ_t contains its entries with values between ϵ_t to $2\epsilon_t$. Thus, we ignore all entries in the PPR matrix column of value less than $\frac{\Delta}{4n}$, the finest scale. Note that entries with value at most $\frac{\Delta}{4n}$ can contribute to at most a quarter to the PageRank of a vertex whose PageRank value is least Δ.

If the sum of a chunk's entries is at least $\Delta/(2\log(n))$, we will refer to it as a *heavy chunk*. The central idea of our algorithm is to efficiently generate robust estimates of the sums for all heavy chunks, as we shall show that it is also sufficient to only provide estimates to heavy chunks.

As the entries in each chunk are within a factor of 2 of each other, we then reduce the task of estimating the sum in a chunk to the problem of approximately counting the size of the chunk. Then conceptually, we estimate the size of each heavy chunk at scale ϵ_t by taking $\tilde{O}(\epsilon_t 4n/\Delta)$ random entries from its column

and counting the numbers of samples in this chunk. The challenge we need to overcome is to efficiently sample all heavy chunks at a scale simultaneously.

This is where we will use our local PageRank approximation algorithm ApproxRow, which in $O(\frac{\log(n)\log(\epsilon^{-1})}{\epsilon})$ time when given a vertex v, returns robust estimates to all entries of values at least ϵ in v's row in the PPR matrix. To achieve $\tilde{O}(n/\Delta)$ queries and running time, we call ApproxRow $\tilde{\Theta}(\frac{n}{\Delta}\epsilon_t)$ times at scale ϵ_t, and we will show that it is sufficient to sample this much (or little).

In the last step of the algorithm, for each node j, we will simply sum up over all scales ϵ_t, its estimated values weighted by a normalizing factor $\frac{\Delta}{\epsilon_t 2\log^2(n)}$. Then the algorithm will output only those j's where the sum is at least $\frac{\Delta}{4}$ and their estimated PageRank values.

A detailed pseudo-code of our algorithm, *ApproximatePageRank*, is given below.

Algorithm 1. ApproximatePageRank

Require: PageRank threshold Δ, a network $G = (V, E)$ on n nodes accessible only by Jump and RandomCrawl queries.

// **First-Part** //

Initialize a binary search tree, ChunkTree, indexed lexicographically by a two-tuple key (nodeID, ϵ).
for $t = 0$ to $\log(\frac{n}{4\Delta})$ **do**
  Set the additive error $\epsilon_t = 2^{-t}$.
  for $(\frac{n}{\Delta}\epsilon_t 4\log^2(n))$ times **do**
    Jump to a random node, call it v.
    Call list = ApproxRow($v, \frac{\epsilon_t}{2}, \frac{1}{2}$) and update the chunk size estimate affiliated vertices in list as the following:
    for each pair (nodeID, ϵ_t) in the list **do**
      if there exists an entry e with key (nodeID, ϵ_t) in ChunkTree **then**
        Update entry e's value by adding 1 to its current value.
      else
        Create an entry in ChunkTree with key (nodeID, ϵ_t) and value 1.
      end if
    end for
  end for (at scale ϵ_t).
end for (for all scales)

// **Second-Part** //

Initialize a final tree, called TreeofPageRankValues, indexed by key (nodeID).
for all elements (chunks) in ChunkTree that all belong to same node i (namely, have i as the first part of their key) **do**
  if chunk has value, val, at least $\frac{1}{2}\log(n)$ **then**
    Let ϵ be the second part of the chunk's key.
    Add $\frac{\Delta}{2\epsilon\log^2(n)}$ to the entry indexed by (i) in TreeofPageRankValues.
  end if
  Output all elements in TreeofPageRankValues with at least $\Delta/4$
end for

In the proofs for the following two theorems, we will analyze the performance of this algorithm. Note that we will ignore the dependence of the running time on α as for all standard PageRank computations, it is taken to be a fixed constant independent of input size [4].

Theorem 1 (Complexity of ApproximatePageRank). *The runtime of algorithm ApproximatePageRank is upper bounded by* $\tilde{O}(n/\Delta)$.

Proof. The algorithm uses $O(\log(n/\Delta))$ scales. In *First-Part* of the algorithm, for scale ϵ_t, it makes $\frac{n}{\Delta}\epsilon_t 4\log^2(n)$ Jump queries and for each query it runs ApproxRow$(v, \epsilon_t/2, 1/2)$, where v represents the random vertex returned by the query. ApproxRow then has a runtime of $O(\frac{\log(n)\log(\epsilon_t^{-1})}{\epsilon_t})$. Thus, the total runtime complexity is $\tilde{O}(\frac{n}{\Delta})$ as the finest scale is Δ/n and there are at most $\log n$ scales. In addition to the time spent on querying the network, the algorithm takes $\Theta(\log(n))$ per step overhead for each access/update in its data structure.

In *Second-Part* of the algorithm, it makes no new queries. As there are only $\tilde{O}(n/\Delta)$ items in the data structure ChunkTree and then TreeofPageRankValues, the complexity of this summation part is $\tilde{O}(n/\Delta)$. The last step of outputting all nodes in the tree with value bigger than a threshold can easily be done in linear time in the size of the tree, which is $\tilde{O}(n/\Delta)$. □

Theorem 2 (Correctness of ApproximatePageRank). *Given* Δ *and constant* $c > 3$, *ApproximatePageRank outputs, with probability* $1 - o(1)$, *all nodes with PageRank at least* Δ *but no node with PageRank smaller than*[3] Δ/c.

Proof. For $v \in V$, let $(p_1^v, p_2^v, \ldots, p_n^v)$ be v's column in the PPR matrix. Let ChunkSet$(v, \epsilon) = \{i : \epsilon \le p_i^v < 2\epsilon\}$, ChunkSize$(v, \epsilon) = |\text{ChunkSet}(v, \epsilon)|$, and ChunkSum$(v, \epsilon) = \sum_{i=1}^{n} \{p_i^v : \epsilon \le p_i^v < 2\epsilon\}$.

Recall a chunk is heavy is its chunksum $\Delta/\log(n)$. We now prove that at the end of First-Part in Algorithm ApproximatePageRank, all heavy chunks are well approximated.

To focus on the essence of the proof for multi-scale matrix sampling, we first assume that all the values returned by ApproxRow are exact (with no error at all). We call this assumption, the *perfect row approximation assumption*. We will later show that when removing this assumption the approximation scheme would only be affected by a multiplicative factor of three, namely the effective value of c in Significant PageRanks would be one third its value under perfect row approximations.

Lemma 1 (key lemma). *Let* $\epsilon_t = 2^{-t}$, *for* $1 \le t \le \frac{n}{4\Delta}$. *The following holds with probability* $1 - o(1)$*:*

- *If ChunkSum*$(v, \epsilon_t) \ge \frac{\Delta}{2\log(n)}$ *then at the end of First-Part in the algorithm the entry in ChunkTree with key* (v, ϵ_t), *namely the algorithm's approximation of ChunkSize*(v, ϵ_t), *is at least* $\log n/2$ *and is between* $\frac{ChunkSize(v,\epsilon_t)}{\Delta} \cdot \epsilon_t 2\log^2(n)$ *to* $\frac{ChunkSize(v,\epsilon_t)}{\Delta} \cdot \epsilon_t 8\log^2(n)$.

[3] Again for exposition, we present our algorithm and its analysis for $c = 6$. We later show that the theorem on a slightly modified algorithm holds for any constant $c > 3$.

– *If* $\text{ChunkSum}(v, \epsilon_t) \leq \frac{\Delta}{4\log(n)}$ *then at the end of First-Part in the algorithm the entry in ChunkTree with key* (v, ϵ_t)*, namely the algorithm's approximation of* $\text{ChunkSize}(v, \epsilon_t)$*, is smaller than* $\frac{\log(n)}{2}$*.*

Proof. Note that $\frac{1}{2\epsilon_t}\text{ChunkSum}(v, \epsilon_t) \leq \text{ChunkSize}(v, \epsilon_t) \leq \frac{1}{\epsilon_t}\text{ChunkSum}(v, \epsilon_t)$. So, if $\text{ChunkSum}(v, \epsilon_t) \geq \frac{\Delta}{2\log(n)}$ then $\text{ChunkSize}(v, \epsilon_t)/n \geq \Delta/(4\epsilon_t n \log n)$.

Thus, when sampling $4\epsilon_t n \log^2(n)/\Delta$ random rows (as in line 5 of the algorithm), the expected number of entries in the chunk that ApproxRow discovers is at least $\text{ChunkSize}(v, \epsilon_t)\epsilon_t 4\log^2(n)/\Delta \geq \log(n)$. By a standard multiplicative Chernoff bound (see appendix), with probability $1 - o(1)$, after multiplying the count by $\Delta/(2\epsilon_t \log^2 n)$, we can approximate $\text{ChunkSize}(v, \epsilon_t)$ within a multiplicative factor of 2. Moreover, if $\text{ChunkSum}(v, \epsilon_t) \leq \frac{\Delta}{4\log(n)}$ then, its estimated value is at most twice its value, namely smaller than $\frac{\log(n)}{2}$. □

Lemma 2. *The following holds with probability* $1 - o(1)$ *under the perfect row approximation assumption:*

– *If* $\text{PageRank}(v) \geq \Delta$*, then the algorithm will output* v *and will estimate its PageRank value to a value between* $\text{PageRank}(v)/4$ *to* $2\text{PageRank}(v)$*.*
– *If* $\text{PageRank}(v) < \Delta/8$*, then the algorithm will not output* v*.*
– *If* $\Delta/8 \leq \text{PageRank}(v) < \Delta$*, then the algorithm might output* v*. If* v *is outputted, then its estimated PageRank value is between* $\text{PageRank}(v)/16$ *to* $2\text{PageRank}(v)$*.*

Proof. By lemma 1, that the sums of each heavy chunk are well estimated to within a multiplicative factor of 2.

Since there are at most $\log n$ chunks in column, the contribution from all non-heavy chunks is at most $\log n(\Delta/(2\log n)) = \Delta/2$. Thus, if v's PageRank is at least Δ, then the contribution from its heavy chunks is at least $\Delta/2$. Consequently, and the algorithm's approximation to v's PageRank will be at least $\Delta/4$ and at most 2Δ, and this vertex will be outputted.

We can similarly establish the other two cases as stated in the lemma. □

We now turn to discuss the effect of having only approximate values computed in ApproxRow calls on the guarantees of ApproxmatePageRank.

Lemma 3. *Given parameters* $0 < \epsilon, \rho < 1$*, removing the perfect row approximation assumption changes the approximation constant* c *by at most* 3 *times its value as well as changes the estimated PageRank values computed by ApproximatePageRank to be at most three times their value.*

Proof. The PPR matrix is effectively computed using calls to ApproxRow by the algorithm.

Given $\epsilon > 0$, consider an element $\epsilon \leq PPR(v, j) \leq 2\epsilon$ for some nodes v, j. There are two sources for having this element estimator differ from it real value. First, ApproxRow (with parameters $\epsilon = \epsilon/2$ and $\rho = 1/2$) computes approximate values so the estimated value is between $(1 - \rho - 1/4)$ its real value to $(1 + \rho + 1/4)$

(we could put the additive $\epsilon/4$ error in the multiplicative approximation factor since $\epsilon \leq PPR(v,j) \leq 2\epsilon$). In particular, in the algorithm we pick $\rho = 1/2$. However, one could replace both ρ and ϵ with smaller values to get an approximation that gets closer to the true value: Replacing ρ by $k_1\rho$ and ϵ by $k_2\epsilon$ for any integral k_1, k_2 would increase the total runtime by only a factor of $k_1^2 k_2 \frac{\log(k_2)}{\log(\epsilon^{-1})}$. The second source why the estimator differs from its true value is double counting. An element with a true value between $\epsilon/2$ to ϵ as well as one with a true value between 2ϵ to 4ϵ could appear in a realization as an element with a value between ϵ to 2ϵ. However (by applying Chernoff bound), elements with true value smaller than $\epsilon/2$ as well as those with value bigger than 4ϵ would not appear as such. Thus due to double counting the sum of elements in each column can be at most three times its real value. If we denote the PageRank of node j by $\Delta(j)$ and the value it gets from the realized column values by $\Delta'(j)$ then,

$$(1 - k_1\rho - k_2/2)\Delta(j) \leq \Delta'(j) \leq 3(1 + k_1\rho + k_2/2)\Delta(j).$$

In particular, algorithm ApproximatePageRank uses $\rho = \frac{1}{2}$, $k_1 = 1$ and $k_2 = \frac{1}{2}$ which gives

$$\frac{1}{4}\Delta(j) \leq \Delta'(j) < 6\Delta(j).$$

□

This ends the proof of Theorem 2. □

4 Lower Bound Construction for PageRank Approximations

We now turn to prove a corresponding lower bound for PageRank approximations.

The lower bound construction will show that, any algorithm, making less than $\Omega(\frac{n}{\Delta})$ Jump and Crawl queries, will fail, with constant probability, to find any node with PageRank at least Δ on the graph. This holds true for any type of implementation of a Crawl query (including the RandomCrawl one). Given positive integers n and $\Delta < \frac{n}{2} - 1$, we construct an undirected graph on n nodes made of a path subgraph on $n - d - 1$ nodes and an isolated star subgraph on $d + 1$ nodes, where $d = 2\Delta$. See figure 1 for an illustration. Fix $0 < \alpha < 1$, the teleportation probability. By solving the PageRank equations it is not hard to check that each node on the path subgraph has PageRank value of 1, the hub of the subgraph has PageRank $\frac{d}{2} + \frac{1}{2(1-\alpha)}$ and each leaf of the star subgraph has PageRank of $\frac{1}{d}(d + 1 - \frac{d}{2} - \frac{1}{2(1-\alpha)}) \leq \frac{1}{2} + \frac{1}{d}$. As $\Delta = \frac{d}{2}$, the only node with PageRank at least Δ is the hub of the star subgraph. However, for any $\epsilon > 0$, in order to find any node that belongs to the star subgraph one needs to make, with probability at least $1 - \frac{1}{e} - \epsilon$, at least $\frac{n}{d} = \Omega(\frac{n}{\Delta})$ Jump queries.

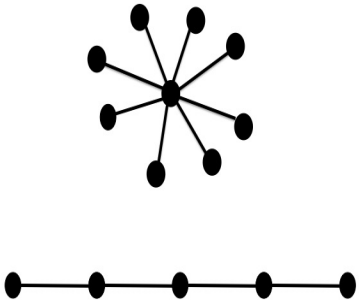


Fig. 1. An example illustrating the path-star graph of the lower bound construction for PageRank computations

5 Local Robust Computation of Personalized PageRank

We now describe a method, *ApproxRow*, based only on local computations that approximates a node's personalized PageRank vector. The pseudo-code is given on the next page.

Theorem 3 (Complexity of ApproxRow). *For any node v and values $0 < \epsilon, \rho < 1$, the runtime of ApproxRow(v, ϵ, ρ) is upper bounded by $O(\frac{\log(n)\log(\epsilon^{-1})}{\epsilon\rho^2 \log_2(\frac{1}{1-\alpha})})$.*

Proof. The algorithm performs $\frac{1}{\epsilon\rho^2} \cdot 16\log(n)$ rounds where at each round it simulates a random walk with termination probability of α for at most *length* steps. Each step is simulated by taking a Jump ('termination' step) with probability α and taking a RandomCrawl step with probability $1-\alpha$. Thus the total number of queries used is $\frac{16\log(n)}{\epsilon\rho^2} \cdot \log_{\frac{1}{(1-\alpha)}}(\frac{3}{\epsilon}) = O(\frac{\log(n)\log(\epsilon^{-1})}{\epsilon\rho^2 \log_2(\frac{1}{1-\alpha})})$. □

Theorem 4 (Correctness of ApproxRow). *For any node v and values $0 < \epsilon, \rho < 1$, with probability of at least $1 - \Theta(\frac{1}{n^2})$, ApproxRow$(v, \epsilon, \rho)$ computes a list l with the following properties:*

- *Every node j that is outputted in the list l has an estimated value which is non-negative and lies between $(1-\rho)PPR(v,j) - \frac{\epsilon}{4}$ to $(1+\rho)PPR(v,j)$.*
- *Every node not in the list l has $PPR(v,j) \leq \epsilon/2$.*

Proof. We start with an observation. The personalized PageRank contribution from a node v to node j is exactly the probability that a random walk that starts at v, and at each time step terminates with probability α, and with probability $1-\alpha$ moves to a random out-link of the node it is currently at, was at node j one step before termination. Define 1_v to be the indicator vector of v. The proof of the observation follows from a series of algebraic manipulations on the definition of the PersonalizedPageRankVector of v:

$$\text{PersonalizedPageRank}_v(\cdot) = \alpha \mathbf{1}_v + (1-\alpha)\text{PersonalizedPageRank}_v(\cdot) \cdot D^{-1}A.$$

Algorithm 2. ApproxRow

Require: A node v in $G = (V, E)$, additive error parameter $0 < \epsilon < 1$, multiplicative approximation parameter $0 < \rho < 1$, teleportation probability $0 < \alpha < 1$.

1: Initialize a binary search tree NodeCountTree where the key is a node's identity.
2: Set $length = \log_{\frac{1}{(1-\alpha)}}(\frac{4}{\epsilon})$.
3: Set $r = \frac{1}{\epsilon\rho^2} \cdot 16\log(n)$.
4: **for** r times **do**
5: Run one realization of a random walk with restart probability α: the walk starts at v and at each time makes, with probability α a 'termination' step by returning to v and terminating, and with probability $1-\alpha$ a RandomCrawl step. The walk is artificially stopped after *length* steps if it has not terminated already.
6: **if** the walk visited a node u just before making a termination step **then**
7: Add 1 to the count stored at u's entry in NodeCountTree.
8: **end if**
9: Output all nodes in NodeCountTree together with their average count (over the r rounds).
10: **end for**

Solving the system gives $\text{PersonalizedPageRank}_v(\cdot) = \alpha\mathbf{1}_v(I-(1-\alpha)D^{-1}A)^{-1} = \alpha\mathbf{1}_v\sum_{i=0}^{\infty}((1-\alpha)D^{-1}A)^i$. This last equation makes the observation clear.

Given a node j, denote by $p_k(v, j)$ the contribution to v's Personalized PageRank vector from walks that are of length at most k. By the above observation, $p_k(v, j) = \alpha\mathbf{1}_v\sum_{i=0}^{k}((1-\alpha)D^{-1}A)^i$.

We ask how much is contributed to j's entry in the Personalized PageRank vector of v from walks of length bigger or equal to k. The contribution is at most $(1-\alpha)^k$ since the walk needs to survive at least k consecutive steps. Taking $(1-\alpha)^k \le \frac{\epsilon}{4}$ will guarantee that at most $\frac{\epsilon}{4}$ is lost by only considering walks of length smaller than k, namely: $\text{PPR}(v, j) - \frac{\epsilon}{4} \le p_k(v, j) \le \text{PPR}(v, j)$.

For that it suffices to take $k = \log_{\frac{1}{(1-\alpha)}}(\frac{4}{\epsilon})$. This is exactly *length*, the length of each walk the algorithm simulates, is set to.

Next, the algorithm provide a estimate of $p_k(v, j)$ by realizing walks of length at most k. The algorithm does so by taking the average count over $\frac{1}{\epsilon\rho^2} \cdot 16\log(n)$ trials. Denote the algorithm's output by $\hat{p}_k(v, j)$. Then, if $p_k(v, j) \ge \frac{\epsilon}{4}$, by the multiplicative Chernoff bound, $Pr(\hat{p}_k(v, j) > (1+\rho)p_k(v, j)) \le \exp(-2\log(n))$ and $Pr(\hat{p}_k(v, j) < (1-\rho)p_k(v, j)) \le \exp(-2\log(n))$.

We can conclude that $(1-\rho)(\text{PPR}(v, j) - \frac{\epsilon}{4}) \le \hat{p}_k(v, j) \le (1+\rho)\text{PPR}(v, j)$.

In particular, nodes with $\text{PPR}(v, j) > \epsilon$ will be estimated to a positive value and outputted as claimed.

Similarly, if $p_k(v, j) \le \frac{\epsilon}{4}$ then by the multiplicative Chernoff bound, $Pr(\hat{p}_k(v, j) > \frac{\epsilon}{2}) \le \exp(-2\log(n))$. In this case we conclude that $\hat{p_k}(v, j) \le \frac{\epsilon}{2}$. Also, $\text{PPR}(v, j) \le \frac{\epsilon}{4} + \frac{\epsilon}{4} \le \frac{\epsilon}{2}$ so $|\text{PPR}(v, j) - \hat{p}_k(v, j)| \le \frac{\epsilon}{2}$. □

Acknowledgments. We thank Brendan Lucier, Elchanan Mossel and Eugene Vorobeychik for their suggestions and the anonymous reviewers for their helpful comments.

References

1. Andersen, R., Borgs, C., Chayes, J.T., Hopcroft, J.E., Mirrokni, V.S., Teng, S.-H.: Local computation of pagerank contributions. Internet Mathematics 5(1), 23–45 (2008)
2. Andersen, R., Chung, F.R.K., Lang, K.J.: Local graph partitioning using pagerank vectors. In: FOCS, pp. 475–486 (2006)
3. Avrachenkov, K., Litvak, N., Nemirovsky, D., Osipova, N.: Monte carlo methods in pagerank computation: When one iteration is sufficient. SIAM Journal on Numerical Analysis 45 (2007)
4. Berkhin, P.: Survey: A survey on pagerank computing. Internet Mathematics 2(1) (2005)
5. Berkhin, P.: Bookmark-coloring approach to personalized pagerank computing. Internet Mathematics 3(1) (2006)
6. Brautbar, M., Kearns, M.: Local algorithms for finding interesting individuals in large networks. In: ICS, pp. 188–199 (2010)
7. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks 30(1-7), 107–117 (1998)
8. Haveliwala, T.H.: Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. Trans. Knowl. Data Eng. 15(4), 784–796 (2003)
9. Jeh, G., Widom, J.: Scaling personalized web search. In: WWW, pp. 271–279 (2003)
10. Kannan, R.: Spectral methods for matrices and tensors. In: STOC, pp. 1–12 (2010)
11. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Stanford University (1998)
12. Rubinfeld, R., Shapira, A.: Sublinear time algorithms. SIAM Journal on Discrete Math. 25, 1562–1588 (2011)

Appendix: Concentration Bounds

Lemma 4 (multiplicative Chernoff bound). *Let X_i be i.i.d. Bernoulli random variables with expectation μ each. Define $X = \sum_{i=1}^{n} X_i$. Then,*

- *For* $0 < \lambda < 1, Pr[X < (1-\lambda)\mu n] < \exp(-\mu n \lambda^2/2)$.
- *For* $0 < \lambda < 1, Pr[X > (1+\lambda)\mu n] < \exp(-\mu n \lambda^2/4)$.
- *For* $\lambda \geq 1, Pr[X > (1+\lambda)\mu n] < \exp(-\mu n \lambda/2)$.

Quick Detection of Nodes with Large Degrees*

Konstantin Avrachenkov[1], Nelly Litvak[2], Marina Sokol[1], and Don Towsley[3]

[1] INRIA, 2004 Route des Lucioles, Sophia-Antipolis, France
[2] University of Twente, The Netherlands
[3] University of Massachusetts Amherst, USA

Abstract. Our goal is to quickly find top k lists of nodes with the largest degrees in large complex networks. If the adjacency list of the network is known (not often the case in complex networks), a deterministic algorithm to find the top k list of nodes with the largest degrees requires an average complexity of $O(n)$, where n is the number of nodes in the network. Even this modest complexity can be very high for large complex networks. We propose to use the random walk based method. We show theoretically and by numerical experiments that for large networks the random walk method finds good quality top lists of nodes with high probability and with computational savings of orders of magnitude. We also propose stopping criteria for the random walk method which requires very little knowledge about the structure of the network.

1 Introduction

We are interested in quickly detecting nodes with large degrees in very large networks. Firstly, node degree is one of centrality measures used for the analysis of complex networks. Secondly, large degree nodes can serve as proxies for central nodes corresponding to the other centrality measures as betweenness centrality or closeness centrality [8,9]. In the present work we restrict ourself to undirected networks or symmetrized versions of directed networks. In particular, this assumption is well justified in social networks. Typically, friendship or acquaintance is a symmetric relation. If the adjacency list of the network is known (not often the case in complex networks), a deterministic algorithm to find the top k list of nodes with the largest degrees requires an average complexity of $O(n)$, where n is the number of nodes in the network. We assume that the degree is available when accessing a node (if this is not the case, the complexity should be counted in terms of links). However, even linear complexity can be very high for very large, possibly varying, complex networks. In the present work we suggest using random walk based methods for detecting a small number of nodes with the largest degree. The main idea is that the random walk very quickly comes across large degree nodes. In our numerical experiments random walks outperform the standard deterministic algorithms by orders of magnitude in terms of

* This research was sponsored by INRIA Alcatel-Lucent Joint Lab, by the NSF under CNS-1065133, and the U.S. Army Research Laboratory under Cooperative Agreement W911NF-09-2-0053.

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 54–65, 2012.

computational complexity. For instance, in our experiments with the web graph of the UK domain (about 18 500 000 nodes) the random walk method spends on average only about 5 400 steps to detect the largest degree node. Potential memory savings are also significant since the method does not require knowledge of the entire network. In many practical applications we do not need a complete ordering of the nodes and even can tolerate some errors in the top list of nodes. We observe that the random walk method obtains many nodes in the top list correctly and even those nodes that are erroneously placed in the top list have large degrees. Therefore, as typically happens in randomized algorithms [12,13], we trade off exact results for very good approximate results or for exact results with high probability and gain significantly in computational efficiency.

The paper is organized as follows: in the next section we introduce our basic random walk with uniform jumps and demonstrate that it is able to quickly find large degree nodes. Then, in Section 3 using configuration model we provide an estimate for the necessary number of steps for the random walk. In Section 4 we propose stopping criteria that use very little information about the network. In Section 5 we show the benefits of allowing few erroneous elements in the top k list. Finally, we conclude the paper in Section 6.

2 Random Walk with Uniform Jumps

Let us consider a random walk with uniform jumps which serves as a basic algorithm for quick detection of large degree nodes. The random walk with uniform jumps is described by the following transition probabilities [1]

$$p_{ij} = \begin{cases} \frac{\alpha/n+1}{d_i+\alpha}, & \text{if } i \text{ has a link to } j, \\ \frac{\alpha/n}{d_i+\alpha}, & \text{if } i \text{ does not have a link to } j, \end{cases} \tag{1}$$

where d_i is the degree of node i. The random walk with uniform jumps can be regarded as a random walk on a modified graph where all the nodes in the graph are connected by artificial edges with a weight α/n. The parameter α controls the rate of jumps. Introduction of jumps helps in a number of ways. As was shown in [1], it reduces the mixing time to stationarity. It also solves a problem encountered by a random walk on a graph consisting of two or more components, namely the inability to visit all nodes. The random walk with jumps also reduces the variance of the network function estimator [1]. This random walk resembles the PageRank random walk. However, unlike the PageRank random walk, the introduced random walk is reversible. One important consequence of the reversibility of the random walk is that its stationary distribution is given by a simple formula

$$\pi_i(\alpha) = \frac{d_i + \alpha}{2|E| + n\alpha} \qquad \forall i \in V, \tag{2}$$

from which the stationary distribution of the unperturbed random walk can easily be retrieved. We observe that the modification preserves the monotonicity of

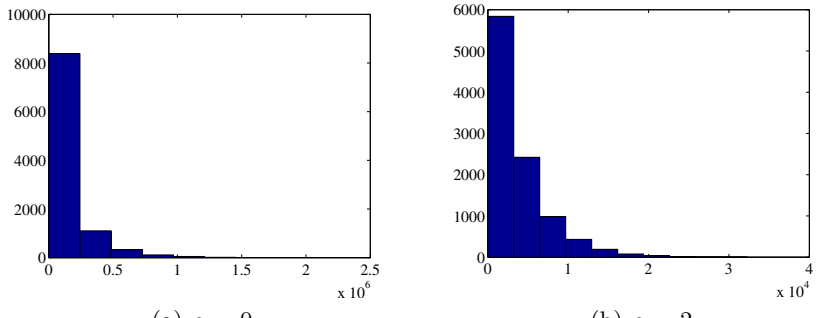
the stationary distribution with respect to the node degree, which is particularly important for our application.

We illustrate on several network examples how the random walk helps us quickly detect large degree nodes. We consider as examples one synthetic network generated by the preferential attachment rule and two natural large networks. The Preferential Attachment (PA) network combines 100 000 nodes. It has been generated according to the generalized preferential attachment mechanism [6]. The average degree of the PA network is two and the power law exponent is 2.5. The first natural example is the symmetrized web graph of the whole UK domain crawled in 2002 [4]. The UK network has 18 520 486 nodes and its average degree is 28.6. The second natural example is the network of co-authorships of DBLP [5]. Each node represents an author and each link represents a co-authorship of at least one article. The DBLP network has 986 324 nodes and its average degree is 6.8.

We carry out the following experiment: we initialize the random walk (1) at a node chosen according to the uniform distribution and continue the random walk until we hit the largest degree node. The largest degrees for the PA, UK and DBLP networks are 138, 194 955, and 979, respectively. For the PA network we have made 10 000 experiments and for the UK and DBLP networks we performed 1 000 experiments (these networks were too large to perform more experiments).

In Figue 1 we plot the histograms of hitting times for the PA network. The first remarkable observation is that when $\alpha = 0$ (no restart) the average hitting time, which is equal to 123 000 steps, is nearly three orders of magnitude larger than 3 720, the hitting time when $\alpha = 2$. The second remarkable observation is that 3 720 is of the same order of magnitude as the value $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 2\,857$, which corresponds to the average return time to the largest degree node in the random walk with jumps.

We were not able to collect a representative number of experiments for the UK and DBLP networks when $\alpha = 0$. The reason for this is that the random walk gets stuck either in disconected or weakly connected components of the networks. For the UK network we were able to make 1 000 experiments with



(a) $\alpha = 0$ (b) $\alpha = 2$

Fig. 1. Histograms of hitting times in the PA network

$\alpha = 0.001$ and obtain the average hitting time 30 750. Whereas if we take $\alpha = 28.6$ for the UK network, we obtain the average hitting time 5 800. Note that the expected return time to the largest degree node in the UK network is given by $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 5\,432$. For the DBLP graph we conducted 1 000 experiments with $\alpha = 0.00001$ and obtained an average hitting time of 41 131. Whereas if we take $\alpha = 6.8$, we obtain an average hitting time of 14 200. The expected return time to the largest degree node in the DBLP network is given by $1/\pi_{max}(\alpha) = (2|E| + n\alpha)/(d_{max} + \alpha) = 13\,607$. The two natural network examples confirm our guess that the average hitting time for the largest degree node is fairly close to the average return time to the largest degree node. Let us also confirm our guess with asymptotic analysis.

Theorem 1. *Without loss of generality, index the nodes such that node 1 has the largest degree,* $(1, i) \in E, i = 2, ..., s, s = d_1 + 1$*, and let* ν *denote the initial distribution of the random walk with jumps. Then, the expected hitting time to node 1 starting from any initial distribution is given by*

$$E_\nu[T_1] = \frac{\sum_{i=2}^{n} d_i + (n-1)\alpha}{d_1 + 2\alpha(1 - 1/n)} + o\left(\min_{i=2,...,s}\{(d_i + \alpha), n\}\right), \tag{3}$$

Proof: The expected hitting time from distribution ν to node 1 is given by the formula

$$E_\nu[T_1] = \nu[I - P_{-1}]^{-1}\underline{1}, \tag{4}$$

where P_{-1} is a taboo probability matrix (i.e., matrix P with the 1-st row and 1-st column removed). The matrix P_{-1} is substochastic but is very close to stochastic. Let us represent it as a stochastic matrix minus some perturbation term:

$$P_{-1} = \tilde{P} - \varepsilon Q = \tilde{P} - \begin{bmatrix} \frac{1+2\alpha/n}{d_2+\alpha} & 0 & & & & 0 \\ 0 & \ddots & & & & \\ & & \frac{1+2\alpha/n}{d_s+\alpha} & & & \\ & & & \frac{2\alpha/n}{d_{s+1}+\alpha} & & \\ & & & & \ddots & 0 \\ 0 & & & & 0 & \frac{2\alpha/n}{d_n+\alpha} \end{bmatrix}$$

We add missing probability mass to the diagonal of $\tilde{P}$, which corresponds to an increase in the weights for self-loops. The matrix $\tilde{P}$ represents a reversible Markov chain with the stationary distribution

$$\tilde{\pi}_j = \frac{d_j + \alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha}.$$

Now we can use the following result from the perturbation theory (see Lemma 1 in [2]):

$$[I - \tilde{P} + \varepsilon Q]^{-1} = \frac{\underline{1}\tilde{\pi}}{\tilde{\pi}(\varepsilon Q)\underline{1}} + X_0 + \varepsilon X_1 + ... \,, \tag{5}$$

where $\tilde{\pi}$ is the stationary distribution of the stochastic matrix $\tilde{P}$. In our case, the quantity $\max_{i=2,\ldots,s}\{1/(d_i+\alpha), 1/n\}$ will play the role of ε. We apply the series (5) to approximate the expected hitting time. Towards this goal, we calculate

$$\tilde{\pi}(\varepsilon Q)\underline{1} = \sum_{j=2}^{n} \tilde{\pi}_j \varepsilon q_{jj}$$

$$= \sum_{j=2}^{s} \frac{d_j+\alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha} \frac{1+2\alpha/n}{d_j+\alpha} + \sum_{j=s+1}^{n} \frac{d_j+\alpha}{\sum_{i=2}^{n} d_i + (n-1)\alpha} \frac{2\alpha/n}{d_j+\alpha}$$

$$= \frac{d_1(1+2\alpha/n) + (n-d_1-1)(2\alpha/n)}{\sum_{i=2}^{n} d_i + (n-1)\alpha} = \frac{d_1 + 2\alpha(1-1/n)}{\sum_{i=2}^{n} d_i + (n-1)\alpha}.$$

Observing that $\nu\underline{1}\tilde{\pi}\underline{1} = 1$, we obtain (3). □

Indeed, the asymptotic expression (3) is very close to $(2|E|+n\alpha)/(d_1+\alpha)$, which is the expected return time to node 1.

Based on the notion of the hitting time we propose an efficient method for quick detection of the top k list of largest degree nodes. The algorithm maintains a top k candidate list. Note that once one of the k nodes with the largest degrees appears in this candidate list, it remains there subsequently. Thus, we are interested in hitting events. We propose the following algorithm for detecting the top k list of largest degree nodes.

Algorithm 1. Random walk with jumps and candidate list

1. *Set k, α and m.*
2. *Execute a random walk step according to (1). If it is the first step, start from the uniform distribution.*
3. *Check if the current node has a larger degree than one of the nodes in the current top k candidate list. If it is the case, insert the new node in the top-k candidate list and remove the worst node out of the list.*
4. *If the number of random walk steps is less than m, return to Step 2 of the algorithm. Stop, otherwise.*

The value of parameter α is not crucial. In our experiments, we have observed that as long as the value of α is neither too small nor not too big, the algorithm performs well. A good option for the choice of α is a value slightly smaller than the average node degree. Let us explain this choice by calculating a probability of jump in the steady state

$$\sum_{j=1}^{n} \pi_j(\alpha) \frac{\alpha}{d_j+\alpha} = \sum_{j=1}^{n} \frac{d_j+\alpha}{2|E|+n\alpha} \frac{\alpha}{d_j+\alpha} = \frac{n\alpha}{2|E|+n\alpha} = \frac{\alpha}{2|E|/n+\alpha}.$$

If α is equal to $2|E|/n$, the average degree, the random walk will jump in the steady state on average every two steps. Thus, if we set α to the average degree or to a slightly smaller value, on one hand the random walk will quickly converge

to the steady state and on the other hand we will not sample too much from the uniform distribution.

The number of random walk steps, m, is a crucial parameter. Our experiments indicate that we obtain a top k list with many correct elements with high probability if we take the number of random walk steps to be twice or thrice as large as the expected hitting time of the nodes in the top k list. From Theorem 1 we know that the hitting time of the large degree node is related to the value of the node's degree. Thus, the problem of choosing m reduces to the problem of estimating the values of the largest degrees. We address this problem in the following section.

3 Estimating the Largest Degrees in the Configuration Network Model

The estimations for the values of the largest degrees can be derived in the configuration network model [7] with a power law degree distribution. In some applications the knowledge of the power law parameters might be available to us. For instance, it is known that web graphs have power law degree distribution and we know typical ranges for the power law parameters.

We assume that the node degrees $D_1, \ldots, D_n$ are i.i.d. random variables with a power law distribution F and finite expectation $E[D]$. Let us determine the number of links contained in the top k nodes. Denote

$$F(x) = P[D \leq x], \quad \bar{F}(x) = 1 - F(x), \quad x \geq 0.$$

Further let $D_{(1)} \geq \ldots \geq D_{(n)}$ be the order statistics of $D_1, \ldots, D_n$. Under the assumption that D_j's obey a power law, we use the results from the extreme value theory as presented in [11], to state that there exist sequences of constants (a_n) and (b_n) and a constant δ such that

$$\lim_{n\to\infty} n\bar{F}(a_n x + b_n) = (1 + \delta x)^{-1/\delta}. \tag{6}$$

This implies the following approximation for high quantiles of F, with exceedance probability close to zero [11]:

$$x_p \approx a_n \frac{(pn)^{-\delta} - 1}{\delta} + b_n.$$

For the jth largest degree, where $j = 2, \ldots, k$, the estimated exceedance probability equals $(j-1)/n$, and thus we can use the quantile $x_{(j-1)/n}$ to approximate the degree $D_{(j)}$ of this node:

$$D_{(j)} \approx a_n \frac{(j-1)^{-\delta} - 1}{\delta} + b_n. \tag{7}$$

The sequences (a_n) and (b_n) are easy to find for a given shape of the tail of F. Below we derive the corresponding results for the commonly accepted Pareto tail distribution of D, that is,

$$\bar{F}(t) = Cx^{-\gamma} \quad \text{for } x > x', \tag{8}$$

where $\gamma > 1$ and x' is a fixed sufficiently large number so that the power law degree distribution is observed for nodes with degree larger than x'. In that case we have

$$\lim_{n\to\infty} n\bar{F}(a_n x+b_n) = \lim_{n\to\infty} nC(a_n x+b_n)^{-\gamma} = \lim_{n\to\infty} (C^{-1/\gamma} n^{-1/\gamma} a_n x + C^{-1/\gamma} n^{-1/\gamma} b_n)^{-\gamma},$$

which directly gives (6) with

$$\delta = 1/\gamma, \quad a_n = \delta C^{\delta} n^{\delta}, \quad b_n = C^{\delta} n^{\delta}. \tag{9}$$

Substituting (9) into (7) we obtain the following prediction for $D_{(j)}$, $j = 2, \ldots, k$, in the case of the Pareto tail of the degree distribution:

$$D_{(j)} \approx n^{1/\gamma}[C^{1/\gamma}(j-1)^{-1/\gamma} - C^{1/\gamma} + 1]. \tag{10}$$

It remains to find an approximation for $D_{(1)}$, the maximal degree in the graph. From the extreme value theory it is well known that if $D_1, \ldots, D_n$ obey a power law then

$$\lim_{n\to\infty} P\left(\frac{D_{(1)} - b_n}{a_n} \le x\right) = H_\delta(x) = \exp(-(1+\delta x)^{-1/\delta}),$$

where, for Pareto tail, a_n, b_n and δ are defined in (9). Thus, as an approximation for the maximal node degree we can choose $a_n x + b_n$ where x can be chosen as either an expectation, a median or a mode of $H_\delta(x)$. If we choose the mode, $((1+\delta)^{-\delta} - 1)/\delta$, then we obtain an approximation, which is smaller than the one for the 2nd largest degree. Further, the expectation $(\Gamma(1-\delta) - 1)/\delta$ is very sensitive to the value of $\delta = 1/\gamma$, especially when γ is close to one, which is often the case in complex networks. Besides, the parameter γ is hard to estimate with high precision. Thus, we choose the median $(\log(2))^{-\delta} - 1)/\delta$, which yields

$$D_{(1)} \approx a_n \frac{(\log(2))^{-\delta} - 1}{\delta} + b_n = n^{1/\gamma}[C^{1/\gamma}(\log(2))^{-1/\gamma} - C^{1/\gamma} + 1]. \tag{11}$$

For instance, in the PA network $\gamma = 2.5$ and $C = 3.7$, which gives according to (11) $D_{(1)} \approx 127$. (This is a good prediction even though the PA network is not generated according to the configuration model. We also note that even though the extremum distribution in the preferential attachment model is different from that of the configuration model their ranges seem to be very close [10].) This in turn suggests that for the PA network m should be chosen in the range 6 000-18 000 if $\alpha = 2$. As we can see from Figure 2 this is indeed a good range for the number of random walk steps. In the UK network $\gamma = 1.7$ and $C = 90$, which gives $D_{(1)} \approx 82\,805$ and suggests a range of 20 000-30 000 for m if $\alpha = 28.6$. Figure 3 confirms that this is a good choice. The degree distribution of the DBLP network does not follow a power law so we cannot apply the above reasoning to it.

4 Stopping Criteria

Suppose now that we do not have any information about the range for the largest k degrees. In this section we design stopping criteria that do not require knowledge about the structure of the network. As we shall see, knowledge of the order of magnitude of the average degree might help, but this knowledge is not imperative for a practical implementation of the algorithm.

Let us now assume that node j can be sampled independently with probability $\pi_j(\alpha)$ as in (2). There are at least two ways to achieve this practically. The first approach is to run the random walk for a significant number of steps until it reaches the stationary distribution. If one chooses α reasonably large, say the same order of magnitude as the average degree, then the mixing time becomes quite small [1] and we can be sure to reach the stationary distribution in a small number of steps. Then, the last step of a run of the random walk will produce an i.i.d. sample from a distribution very close to (2). The second approach is to run the random walk uninterruptedly, also with a significant value of α, and then perform Bernoulli sampling with probability q after a small initial transient phase. If q is not too large, we shall have nearly independent samples following the stationary distribution (2). In our experiment, $q \in [0.2, 0.5]$ gives good results when α has the same order of magnitude as the average degree.

We now estimate the probability of detecting correctly the top k list of nodes after m i.i.d. samples from (2). Denote by X_i the number of hits at node i after m i.i.d. samples. We note that if we use the second approach to generate i.i.d. samples, we spend approximately m/q steps of the random walk. We correctly detect the top k list with the probability given by the multinomial distribution

$$P[X_1 \geq 1, ..., X_k \geq 1] =$$

$$\sum_{i_1 \geq 1, ..., i_1 \geq 1} \frac{m!}{i_1! \cdots i_k! (m - i_1 - ... - i_k)!} \pi_1^{i_1} \cdots \pi_k^{i_k} (1 - \sum_{i=1}^{k} \pi_i)^{m - i_1 - ... - i_k}$$

but it is not feasible for any realistic computations. Therefore, we propose to use the Poisson approximation. Let Y_j, $j = 1, ..., n$ be independent Poisson random variables with means $\pi_j m$. That is, the random variable Y_j has the following probability mass function $P[Y_j = r] = e^{-m\pi_j}(m\pi_j)^r / r!$. It is convenient to work with the complementary event of not detecting correctly the top k list. Then, we have

$$P[\{X_1 = 0\} \cup ... \cup \{X_k = 0\}] \leq 2P[\{Y_1 = 0\} \cup ... \cup \{Y_k = 0\}]$$

$$= 2(1 - P[\{Y_1 \geq 1\} \cap ... \cap \{Y_k \geq 1\}]) = 2(1 - \prod_{j=1}^{k} P[\{Y_j \geq 1\}])$$

$$= 2(1 - \prod_{j=1}^{k} (1 - P[\{Y_j = 0\}])) = 2(1 - \prod_{j=1}^{k} (1 - e^{-m\pi_j})) =: a, \qquad (12)$$

where the first inequality follows from [12, Thm 5.10]. In fact, in our numerical experiments we observed that the factor 2 in the first inequality is very conservative. For large values of m, the Poisson bound works very well as proper approximation.

For example, if we would like to obtain the top 10 list with at most 10% probability of error, we need to have on average 4.5 hits per each top element. This can be used to design the stopping criteria for our random walk algorithm. Let $\bar{a} \in (0,1)$ be the admissible probability of an error in the top k list. Now the idea is to stop the algorithm after m steps when the estimated value of a for the first time is lower than the critical number $\bar{a}$. Clearly,

$$\hat{a}_m = 2(1 - \prod_{j=1}^{k}(1 - e^{-X_j}))$$

is the maximum likelihood estimator for a, so we would like to choose m such that $\hat{a}_m \leq \bar{a}$. The problem, however, is that we do not know which X_j's are the realisations of the number of visits to the top k nodes. Then let $X_{j_1}, ..., X_{j_k}$ be the number of hits to the current elements in the top k candidate list and consider the estimator

$$\hat{a}_{m,0} = 2(1 - \prod_{i=1}^{k}(1 - e^{-X_{j_i}})),$$

which is the maximum likelihood estimator of the quantity

$$2(1 - \prod_{i=1}^{k}(1 - e^{-m\pi_{j_i}})) \geq a.$$

(Here π_{j_i} is a stationary probability of the node with the score X_{j_i}, $i = 1, \ldots, k$). The estimator $\hat{a}_{m,0}$ is computed without knowledge of the top k nodes or their degrees, and it is an estimator of an upper bound of the estimated probability that there are errors in the top k list. This leads to the following stopping rule.

Stopping rule 0. *Stop at* $m = m_0$*, where*

$$m_0 = \arg\min\{m : \hat{a}_{m,0} \leq \bar{a}\}.$$

The above stopping criterion can be simplified even further to avoid computation of $\hat{a}_{m,0}$. Since

$$\hat{a}_{m,1} := 2(1 - (1 - e^{-X_{j_k}})^k) \geq \hat{a}_{m,0} \geq \hat{a},$$

where X_{j_k} is the number of hits of the worst element in the candidate list. The inequality $\hat{a}_m \leq \bar{a}$ is guaranteed if $\hat{a}_{m,1} \leq \bar{a}$. This leads to the following stopping rule for the random walk algorithm.

Stopping rule 1. *Compute* $x_0 = \arg\min\{x \in \mathbb{N} : (1 - e^{-x})^k \geq 1 - \bar{a}/2.\}$ *Stop at*

$$m_1 = \arg\min\{m : X_{j_k} = x_0\}.$$

We have observed in our numerical experiments that we obtain the best trade off between the number of steps of the random walk and the accuracy if we take α around the average degree and the sampling probability q around 0.5. Specifically, if we take $\bar{a}/2 = 0.15$ ($x_0 = 4$) in Stopping rule 1 for top 10 list, we obtain 87% accuracy for an average of 47 000 random walk steps for the PA network; 92% accuracy for an average of 174 468 random walk steps for the DBLP network; and 94% accuracy for an average of 247 166 random walk steps for the UK network. We have averaged over 1000 experiments to obtain tight confidence intervals.

5 Relaxation of Top k Lists

In the stopping criteria of the previous section we have strived to detect all nodes in the top k list. This costs us a lot of steps of the random walk. We can significantly gain in performance by relaxing this strict requirement. For instance, we could just ask for list of k nodes that contains 80% of top k nodes [3]. This way we can take an advantage of a generic 80/20 rule that 80% of result can be achieved with 20% of effort.

Let us calculate the expected number of top k elements observed in the candidate list up to trial m. Define by X_j the number of times we have observed node j after m trials and

$$H_j = \begin{cases} 1, \text{node } j \text{ has been observed at least once,} \\ 0, \text{node } j \text{ has not been observed.} \end{cases}$$

Assuming we sample in i.i.d. fashion from the distribution (2), we can write

$$E[\sum_{j=1}^{k} H_j] = \sum_{j=1}^{k} E[H_j] = \sum_{j=1}^{k} P[X_j \geq 1] = \sum_{j=1}^{k} (1 - P[X_j = 0]) = \sum_{j=1}^{k} (1 - (1-\pi_j)^m). \tag{13}$$

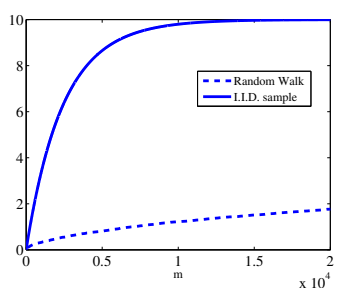
In Figure 2 we plot $E[\sum_{j=1}^{k} H_j]$ (the curve "I.I.D. sample") as a function of m for $k = 10$ for the PA network with $\alpha = 0$ and $\alpha = 2$. In Figure 3 we plot $E[\sum_{j=1}^{k} H_j]$ as a function of m for $k = 10$ for the UK network with $\alpha = 0.001$ and $\alpha = 28.6$. The results for the UK and DBLP networks are similar in spirit.

Here again we can use the Poisson approximation

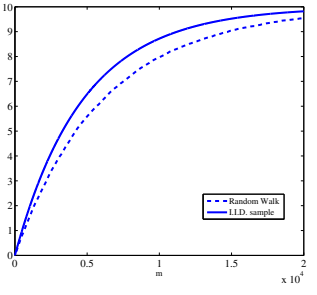
$$E[\sum_{j=1}^{k} H_j] \approx \sum_{j=1}^{k} (1 - e^{-m\pi_j}).$$

In fact, the Poisson approximation is so good that if we plot it on Figures 2 and 3, it nearly covers exactly the curves labeled "I.I.D. sample", which correspond to the exact formula (13). Similarly to the previous section, we can propose stopping criteria based on the Poisson approximation. Denote

$$b_m = \sum_{i=1}^{k} (1 - e^{-X_{j_i}}).$$

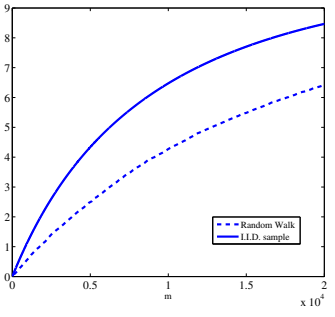


(a) $\alpha = 0$

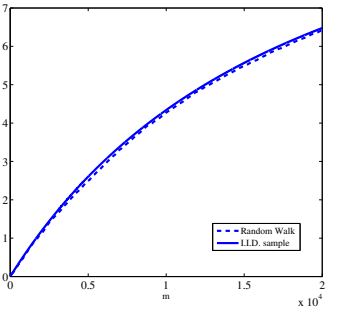


(b) $\alpha = 2$

Fig. 2. Average number of correctly detected elements in top-10 for PA



(a) $\alpha = 0.001$



(b) $\alpha = 28.6$

Fig. 3. Average number of correctly detected elements in top-10 for UK

Stopping rule 2. *Stop at $m = m_2$, where*

$$m_2 = \arg\min\{m : b_m \geq \bar{b}\}.$$

Now if we take $\bar{b} = 7$ in Stopping rule 2 for top-10 list, we obtain on average 8.89 correct elements for an average of 16 725 random walk steps for the PA network; we obtain on average 9.28 correct elements for an average of 66 860 random walk steps for the DBLP network; and we obtain on average 9.22 correct elements for an average of 65 802 random walk steps for the UK network. (We have averaged over 1000 experiments for each network.) This makes for the UK network the gain of more than two orders of magnitude in computational complexity with respect to the deterministic algorithm.

6 Conclusions and Future Research

We have proposed the random walk method with the candidate list for quick detection of largest degree nodes. We have also supplied stopping criteria which

do not require knowledge of the graph structure. In the case of large networks, our algorithm finds top k list of largest degree nodes with few mistakes with the running time orders of magnitude faster than the deterministic algorithms. In future research we plan to obtain estimates for the required number of steps for various types of complex networks.

References

1. Avrachenkov, K., Ribeiro, B., Towsley, D.: Improving Random Walk Estimation Accuracy with Uniform Restarts. In: Kumar, R., Sivakumar, D. (eds.) WAW 2010. LNCS, vol. 6516, pp. 98–109. Springer, Heidelberg (2010)
2. Avrachenkov, K., Borkar, V., Nemirovsky, D.: Quasi-stationary distributions as centrality measures for the giant strongly connected component of a reducible graph. Journal of Comp. and Appl. Mathematics 234, 3075–3090 (2010)
3. Avrachenkov, K., Litvak, N., Nemirovsky, D., Smirnova, E., Sokol, M.: Quick Detection of Top-k Personalized PageRank Lists. In: Frieze, A., Horn, P., Prałat, P. (eds.) WAW 2011. LNCS, vol. 6732, pp. 50–61. Springer, Heidelberg (2011)
4. Boldi, P., Vigna, S.: The WebGraph framework I: Compression techniques. In: Proceedings of WWW 2004 (2004)
5. Boldi, P., Rosa, M., Santini, M., Vigna, S.: Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In: Proceedings of WWW 2011 (2011)
6. Dorogovtsev, S.N., Mendes, J.F.F., Samukhin, A.N.: Structure of growing networks: Exact solution of the Barabasi-Albert model. Phys. Rev. Lett. 85, 4633–4636 (2000)
7. van der Hofstad, R.: Random graphs and complex networks. Lecture Notes (2009), http://www.win.tue.nl/rhofstad/NotesRGCN.pdf
8. Lim, Y., Menasche, D.S., Ribeiro, B., Towsley, D., Basu, P.: Online estimating the k central nodes of a network. In: Proceedings of IEEE NSW 2011 (2011)
9. Maiya, A.S., Berger-Wolf, T.Y.: Online Sampling of High Centrality Individuals in Social Networks. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS, vol. 6118, pp. 91–98. Springer, Heidelberg (2010)
10. Moreira, A.A., Andrade Jr., J.S., Amaral, L.A.N.: Extremum statistics in scale-free network models. Phys. Rev. Lett. 89, 268703, 4 pages (2002)
11. Matthys, G., Beirlant, J.: Estimating the extreme value index and high quantiles with exponential regression models. Statistica Sinica 13(3), 853–880 (2003)
12. Mitzenmacher, M., Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press (2005)
13. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press (1995)

Ranking and Sparsifying a Connection Graph

Fan Chung[1,2] and Wenbo Zhao[2]

[1] Department of Mathematics
[2] Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093, USA
{fan,w3zhao}@ucsd.edu

Abstract. Many problems arising in dealing with high-dimensional data sets involve connection graphs in which each edge is associated with both an edge weight and a d-dimensional linear transformation. We consider vectorized versions of the PageRank and effective resistance which can be used as basic tools for organizing and analyzing complex data sets. For example, the generalized PageRank and effective resistance can be utilized to derive and modify diffusion distances for vector diffusion maps in data and image processing. Furthermore, the edge ranking of the connection graphs determined by the vectorized PageRank and effective resistance are an essential part of sparsification algorithms which simplify and preserve the global structure of connection graphs.

1 Introduction

In this paper, we consider a generalization of graphs, called connection graphs, in which each edge of the graph is associated with a weight and also a "rotation" (which is a linear orthogonal transformation acting on a d-dimensional vector space for some positive integer d). The adjacency matrix and the discrete Laplace operator are acting on the space of vector-valued functions (instead of the usual real-valued functions) and therefore can be represented by matrices of size $dn \times dn$ where n is the number of vertices in the graph.

Connection graphs arise in numerous applications, in particular for data and image processing involving high-dimensional data sets. To quantify the affinities between two data points, it is often not enough to use only a scalar edge weight. For example, if the high-dimensional data set can be represented or approximated by a low-dimensional manifold, the patterns associated with nearby data points are likely to related by certain rotations [29]. There are many recent developments of related research in cryo-electron microscopy [15,28], angular synchronization of eigenvectors [11,27] and vector diffusion maps [29]. In many areas of machine learning, high-dimensional data points in general can be treated by various methods, such as the Principle Component Analysis [18], to reduce vectors into some low-dimensional space and then use the connection graph with rotations on edges to provide the additional information for proximity. In computer vision, there has been a great deal of recent work dealing with trillions

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 66–77, 2012.

of photos that are now available on the web [2]. Feature matching techniques [24] can be used to derive vectors associated with the images. Then information networks of photos can be built which are exactly connection graphs with rotations corresponding to the angles and positions of the cameras in use. The use of connection graphs can be further traced to earlier work in graph gauge theory for computing the vibrational spectra of molecules and examining the spins associated with vibrations [9].

Many information networks arising from massive data sets exhibit the small world phenomenon. Consequently the usual graph distance is no longer very useful. It is crucial to have the appropriate metric for expressing the proximity between two vertices. Previously, various notions of diffusion distances have been defined [29] and used for manifold learning and dimension reduction. Here we consider two basic notions, the *connection PageRank* and the *connection resistance*, (which are generalizations of the usual PageRank and effective resistance). Both the connection PageRank and connection resistance can then be used to define correlations between vertices in the connection graph. To illustrate the usage of both metrics, we derive edge ranking using the connection PageRank and the connection resistance. In the applications to cryo-electron microscopy, the edge ranking can help eliminate the superfluous or erroneous edges that appear because of various "noises".

The notion of PageRank was first introduced by Brin and Page [7] in 1998 for Google's Web search algorithms. Although the PageRank was originally designed for the Web graph, the concepts work well for any graph for quantifying the correlations of pairs of vertices (or pairs of subsets) in any given graph. There are very efficient and robust algorithms for computing and approximating PageRank [3,6,17]. In this paper, we further generalize the PageRank for connection graphs and give efficient and sharp approximation algorithms for computing the connection PageRank.

The effective resistance plays a major role in electrical network theory and can be traced back to the classical work of Kirchhoff [22]. Here we consider a generalized version of effective resistance for the connection graphs. To illustrate the usage of connection resistance, we examine a basic problem on graph sparsification. Graph sparsification was first introduced by Benczúr and Karger [5,19,20,21] for approximately solving various network design problems. The heart of the graph sparsification algorithms is the sampling technique for randomly selecting edges. The goal is to approximate a given graph G on n vertices by a sparse graph $\tilde{G}$, called a sparsifier, with fewer edges on the same set of vertices such that every cut in the sparsifier $\tilde{G}$ has its size within a factor $(1 \pm \epsilon)$ of the size of the corresponding cut in G for some constant ϵ. Spielman and Teng [30] constructed a spectral sparsifier with $O(n \log^c n)$ edges for some large constant c. In [33], Spielman and Srivastava gave a different sampling scheme using the effective resistances to construct an improved spectral sparsifier with only $O(n \log n)$ edges. In this paper, we will construct the connection sparsifer using the weighted connection resistance.

A Summary of the Results

Our results can be summarized as follows:

- We give definitions for the connection graph and the connection Laplacian in Section 2. In particular, we discuss the notion of "consistency" in a connection graph (which is considered to be the ideal situation for various applications). We give a characterization for a consistent connection graph by using the eigenvalues of the connection Laplacian.
- We introduce the connection PageRank in Section 3. We give two efficient approximation algorithms for computing the connection PageRank vectors. The two algorithms provide somewhat different approximation guarantees. For ranking edges, we require the approximation to be sharp with error bounds of order $\log(1/\epsilon)$ so that the bounds will still be effective when the ϵ is taken to be in the range of $\widetilde{O}\left(1/n^2\right)$, for example.
- We define the connection resistance in Section 4 and then examine various properties of the connection resistance.
- We use the connection resistance to give an edge ranking algorithm and a sparsification algorithm for connection graphs in Section 5.

2 Preliminaries

For positive integers m, n and d, we consider a family of matrices, denoted by $\mathcal{F}(m, n, d; \mathbf{R})$ consisting of all $md \times nd$ matrices with real-valued entries. A matrix in $\mathcal{F}(m, n, d; \mathbf{R})$ can also be viewed as a $m \times n$ matrix whose entries are represented by $d \times d$ blocks. A *rotation* is a matrix that is used to perform a rotation in Euclidean space. Namely, a rotation O is a square matrix, with real entries, satisfying $O^T = O^{-1}$ and $\det(O) = 1$. All rotation matrices of size $d \times d$ are known to form the special orthogonal group $\mathsf{SO}(d)$. It is easy to check that all eigenvalues of a rotation O are of norm 1. Furthermore, a rotation $O \in \mathsf{SO}(d)$ with d odd has an eigenvalue 1 (see [14]).

2.1 The Connection Laplacian

Suppose $G = (V, E, w)$ is an undirected graph with vertex set V, edge set E and edge weights $w_{uv} = w_{vu} > 0$ for edges (u, v) in E. Suppose each oriented edge (u, v) is associated with a rotation matrix $O_{uv} \in \mathsf{SO}(d)$ satisfying $O_{uv}O_{vu} = I_{d\times d}$. Let O denote the set of rotations associated with all oriented edges in G. The *connection graph*, denoted by $\mathbb{G} = (V, E, O, w)$, has G as the *underlying graph*. The *connection matrix* $\mathbb{A}$ of $\mathbb{G}$ is defined by:

$$\mathbb{A}(u, v) = \begin{cases} w_{uv}O_{uv} & \text{if}(u, v) \in E, \\ 0_{d\times d} & \text{if}(u, v) \notin E \end{cases}$$

where $0_{d\times d}$ is the zero matrix of size $d \times d$. In other words, for $|V| = n$, we view $\mathbb{A} \in \mathcal{F}(n, n, d; \mathbf{R})$ as a block matrix where each block is either a $d \times d$

rotation matrix O_{uv} multiplied by a scalar weight w_{uv}, or a $d \times d$ zero matrix. The matrix $\mathbb{A}$ is symmetric as $O_{uv}^T = O_{vu}$ and $w_{uv} = w_{vu}$. The diagonal matrix $\mathbb{D} \in \mathcal{F}(n, n, d; \mathbf{R})$ is defined by the diagonal blocks $\mathbb{D}(u, u) = d_u I_{d\times d}$ for $u \in V$. Here d_u is the weighted degree of u in G, i.e., $d_u = \sum_{(u,v)\in E} w_{uv}$.

The *connection Laplacian* $\mathbb{L} \in \mathcal{F}(n, n, d; \mathbf{R})$ of a graph $\mathbb{G}$ is the block matrix $\mathbb{L} = \mathbb{D} - \mathbb{A}$. Recall that for any orientation of edges of the underlying graph G on n vertices and m edges, the combinatorial Laplacian L can be written as $L = B^T W B$ where W is a $m \times m$ diagonal matrix with $W_{e,e} = w_e$, and B is the edge-vertex incident matrix of size $m \times n$ such that $B(e, v) = 1$ if v is e's head; $B(e, v) = -1$ if v is e's tail; and $B(e, v) = 0$ otherwise. A useful observation for the connection Laplacian is the fact that it can be written in a similar form. Let $\mathbb{B} \in \mathcal{F}(m, n, d; \mathbf{R})$ be the block matrix given by

$$\mathbb{B}(e, v) = \begin{cases} O_{uv} & v \text{ is } e\text{'s head}, \\ -I_{d\times d} & v \text{ is } e\text{'s tail}, \\ 0_{d\times d} & \text{otherwise}. \end{cases}$$

Also, let the block matrix $\mathbb{W} \in \mathcal{F}(m, m, d; \mathbf{R})$ denote a diagonal block matrix given by $\mathbb{W}(e, e) = w_e I_{d\times d}$. Then, we have the following useful lemma whose proof is omitted here.

Lemma 1. *(i) For any orientation of edges on graph $\mathbb{G}$, the connection Laplacian of $\mathbb{G}$ can be written as $\mathbb{L} = \mathbb{B}^T \mathbb{W} \mathbb{B}$. (ii) For any function $f : V \to \mathbf{R}^d$, we have*

$$f \mathbb{L} f^T = \sum_{(u,v)\in E} w_{uv} \left\| f(u) - f(v) O_{uv} \right\|_2^2 \tag{1}$$

where $f(v)$ here is regarded as a row vector of dimension d. (iii) $\mathbb{L}$ has a complete set of real eigenfunctions $\phi_1, \phi_2, \ldots, \phi_{nd}$ and corresponding real eigenvalues $0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{nd}$. Furthermore, $\lambda_i = 0$ if and only if $\phi_i(u) O_{vu} = \phi_i(v)$ for all $(u, v) \in E$.

2.2 The Consistency of a Connection Graph

For a connection graph $\mathbb{G} = (V, E, O, w)$, we say $\mathbb{G}$ is *consistent* if for any cycle $c = (v_1, v_2, \ldots, v_k, v_1)$ the product of rotations along the cycle is the identity matrix, i.e. $O_{v_k v_1} \prod_{i=1}^{k-1} O_{v_i v_{i+1}} = I_{d\times d}$. In other words, for any two vertices u and v, the products of rotations along different paths from u to v are the same. In the following theorem, we give a characterization for a consistent connection graph by using the eigenvalues of the connection Laplacian.

Theorem 1. *Suppose a connected graph G is the underlying graph of a connection graph $\mathbb{G}$. Then $\mathbb{G}$ is consistent if and only if the eigenvalues of $\mathbb{L}$ are d copies of eigenvalues of L where $\mathbb{L}$ is the connection Laplacian of $\mathbb{G}$, L is the combinatorial Laplacian of G and d is the dimension of rotations.*

Proof. ($\Longrightarrow$). For a fixed vertex $u \in V$ and an arbitrary d-dimensional vector $\widehat{x}$, we can define a function $\widehat{f} : V \to \mathbf{R}^d$, by defining $\widehat{f}(u) = \widehat{x}$ initially. Then we assign $\widehat{f}(v) = \widehat{f}(u)O_{vu}$ for all the neighbors v of u. Since G is connected and $\mathbb{G}$ is consistent, we can continue the assigning process to all neighboring vertices without any conflict until all vertices are assigned. The resulting function $\widehat{f} : V \to \mathbf{R}^d$ satisfies $\widehat{f}\mathbb{L}\widehat{f}^T = \sum_{(u,v)\in E} w_{uv} \left\| \widehat{f}(u) - \widehat{f}(v)O_{uv} \right\|_2^2 = 0$. Therefore $\mathbb{L}$ has an eigenspace of dimension d for the eigenvalue 0, and it has d orthogonal eigenfunctions $\widehat{f}_1, \ldots, \widehat{f}_d$ corresponding to eigenvalues 0.

Now, let us consider the underlying graph G. Let $f_i : V \to \mathbf{R}$ denote the eigenfunctions of L corresponding to the eigenvalue λ_i for $i \in [n]$ respectively. Our proof of this direction follows directly from the following claim whose proof is omitted here.

Claim. Functions $f_i \otimes \widehat{f}_k : V \to \mathbf{R}^d$ for $i \in [n], k \in [d]$ are the orthogonal eigenfunctions of $\mathbb{L}$ corresponding to eigenvalue λ_i where $f_i \otimes \widehat{f}_k(v) = f_i(v)\widehat{f}_k(v)$.

($\Longleftarrow$). Suppose $\widehat{f}_1, \ldots, \widehat{f}_d$ are d orthogonal eigenfunctions of $\mathbb{L}$ corresponding to the eigenvalue 0. Equation (1) implies that for any pair of vertices $u, v \in V$ joined by a path $P = (u = v_1, v_2, \ldots, v_k)$, we have, for all $j = 1, \ldots, d$, $\widehat{f}_j(u) \prod_{i=1}^{k-1} O_{v_i v_{i+1}} = \widehat{f}_j(v)$. Furthermore, for two adjacent vertices u and v, we have

$$\langle \widehat{f}_i(u), \widehat{f}_j(u) \rangle = \langle \widehat{f}_i(u)O_{uv}, , \widehat{f}_j(u)O_{uv} \rangle = \langle \widehat{f}_i(v), \widehat{f}_j(v) \rangle$$

Therefore, $\widehat{f}_1(v), \ldots, \widehat{f}_d(v)$ must form an orthogonal basis of $\mathbf{R}^d$ for all $v \in V$. Now, suppose that for two vertices u and v there are two different paths from u to v such that the products, denoted by Π_1 and Π_2, of rotations along the two paths are different. There must be a vector $g(u) \in \mathbf{R}^d$ such that $g(u)\Pi_1 \neq g(u)\Pi_2$. However, $\widehat{f}_1(v), \ldots, \widehat{f}_d(v)$ form a orthogonal basis of $\mathbf{R}^d$ and $\widehat{f}_i(u)\Pi_1 = \widehat{f}_i(v) = \widehat{f}_i(u)\Pi_2$ for $1 \leq i \leq d$. This is impossible. The theorem is proved. □

2.3 Random Walks on a Connection Graph

Consider the underlying graph G of a connection graph $\mathbb{G} = (V, E, O, w)$. A random walk on G is defined by the transition probability matrix P where $P_{uv} = w_{uv}/d_u$ denotes the probability of moving to a neighbor v at a vertex u. We can write $P = D^{-1}A$, where A is the weighted adjacency matrix of G and D is the diagonal matrix of weighted degree.

In a similar way, we can define a random walk on the connection graph $\mathbb{G}$ by setting the transition probability matrix $\mathbb{P} = \mathbb{D}^{-1}\mathbb{A}$. While P acts on the space of real-valued functions, $\mathbb{P}$ acts on the space of vector-valued functions $f : V \to \mathbf{R}^d$.

Theorem 2. *Suppose $\mathbb{G}$ is consistent. Then for any positive integer t, any vertex $u \in V$ and any function $\widehat{s} : V \to \mathbf{R}^d$ satisfying $\|\widehat{s}(v)\|_2 = 0$ for all $v \in V \backslash \{u\}$, we have $\|\widehat{s}(u)\|_2 = \sum_v \|\widehat{s}\ \mathbb{P}^t(v)\|_2$.*

Proof. The proof of this theorem is straightforward from the assumption that $\mathbb{G}$ is consistent. For $\widehat{p} = \widehat{s}\,\mathbb{P}^t$, note that $\widehat{p}(v)$ is the summation of all d dimensional vectors resulted from rotating $\widehat{s}(u)$ via rotations along all possible paths of length t from u to v. Since $\mathbb{G}$ is consistent, the rotated vectors arrive at v via different paths are positive multiples of the same vector. Also the rotations maintain the 2-norm of vectors. Thus, $\frac{\|\widehat{p}(v)\|_2}{\|\widehat{s}(u)\|_2}$ is simply the probability that a random walk in G arriving at v from u after t steps. The theorem follows. □

3 PageRank Vectors in a Connection Graph

The PageRank vector is based on random walks. Here we consider a lazy walk on G with the transition probability matrix $Z = \frac{I+P}{2}$. In [3], a PageRank vector $\mathrm{pr}_{\alpha,s}$ is defined by a recurrence relation involving a seed vector s (as a probability distribution) and a positive jumping constant $\alpha < 1$ (or transportation constant). Namely, $\mathrm{pr}_{\alpha,s} = \alpha s + \mathrm{pr}_{\alpha,s}(1-\alpha)Z$.

For the connection graph $\mathbb{G}$, the PageRank vector $\widehat{\mathrm{pr}}_{\alpha,\widehat{s}} : V \to \mathbf{R}^d$ is defined by the same recurrence relation involving a seed vector $\widehat{s} : V \to \mathbf{R}^d$ and a positive jumping constant $\alpha < 1$:

$$\widehat{\mathrm{pr}}_{\alpha,\widehat{s}} = \alpha\widehat{s} + (1-\alpha)\widehat{\mathrm{pr}}_{\alpha,\widehat{s}}\mathbb{Z}$$

where $\mathbb{Z} = \frac{1}{2}\left(I_{nd\times nd} + \mathbb{P}\right)$ is the transition probability matrix of a lazy random walk on $\mathbb{G}$. An alternative definition of the PageRank vector is the following geometric sum of random walks:

$$\widehat{\mathrm{pr}}_{\alpha,\widehat{s}} = \alpha\sum_{t=0}^{\infty}(1-\alpha)^t\,\widehat{s}\,\mathbb{Z}^t = \alpha\widehat{s} + (1-\alpha)\widehat{\mathrm{pr}}_{\alpha,\widehat{s}\mathbb{Z}}. \tag{2}$$

By Theorem 2 and Equation (2), we here state the following useful fact concerning PageRank vectors for a consistent connection graph.

Proposition 1. *Suppose that a connection graph $\mathbb{G}$ is consistent. Then for any $u \in V$, $\alpha \in (0,1)$ and any function $\widehat{s} : V \to \mathbf{R}^d$ satisfying $\|\widehat{s}(u)\|_2 = 1$ and $\widehat{s}(v) = 0$ for $v \neq u$, we have $\left\|\widehat{\mathrm{pr}}_{\alpha,\widehat{s}}(v)\right\|_2 = \mathrm{pr}_{\alpha,\chi_u}(v)$. In particular, $\sum_{v\in V}\left\|\widehat{\mathrm{pr}}_{\alpha,\widehat{s}}(v)\right\|_2 = \|\mathrm{pr}_{\alpha,\chi_u}\|_1 = 1$.*

We will call such a PageRank vector $\widehat{\mathrm{pr}}_{\alpha,\widehat{s}}$ a connection PageRank vector on u. To compute a connection PageRank, we need the following subroutine called Push and Lemma 2 for proving Theorem 3.

Lemma 2. *Let $\widehat{p}'$ and $\widehat{r}'$ denote the resulting vectors after performing operation Push(u) with $\widehat{p}$ and $\widehat{r}$. Then $\widehat{p}' + \widehat{\mathrm{pr}}_{\alpha,\widehat{r}'} = \widehat{p} + \widehat{\mathrm{pr}}_{\alpha,\widehat{r}}$.*

Theorem 3. *For a vector s with $\sum_{v\in V}\|\widehat{s}(v)\|_2 \leq 1$, and a constant $0 < \epsilon < 1$, the algorithm ApproximatePR$(\widehat{s},\alpha,\epsilon)$ computes an approximate PageRank vector $\widehat{p} = \widehat{\mathrm{pr}}_{\alpha,\widehat{s}-\widehat{r}}$ such that the residual vector $\widehat{r}$ satisfies $\frac{\|\widehat{r}(v)\|_2}{d_v} \leq \epsilon$, for all $v \in V$ and $\sum_{v:\|\widehat{p}(v)\|_2>0} d_v \leq \frac{1}{\epsilon\alpha}$. The running time for the algorithm is $O(\frac{d^2}{\epsilon\alpha})$.*

$\mathsf{Push}(u, \alpha)$:

Let $\widehat{p}' = \widehat{p}$ and $\widehat{r}' = \widehat{r}$, except for these changes:

1. Let $\widehat{p}'(u) = \widehat{p}(u) + \alpha\widehat{r}(u)$ and $\widehat{r}'(u) = \frac{1-\alpha}{2}\widehat{r}(u)$.
2. For each vertex v such that $(u, v) \in E$: $\widehat{r}'(v) = \widehat{r}(v) + \frac{(1-\alpha)w_{uv}}{2d_u}\widehat{r}(u)\ O_{uv}$.

$(\widehat{p}, \widehat{r}) = \mathsf{ApproximatePR}(\widehat{s}, \alpha, \epsilon)$

1. Let $\widehat{p}(v) = \mathbf{0}$ and $\widehat{r}(v) = \widehat{s}(v)$ for all $v \in V$.
2. While $\|\widehat{r}(u)\|_2 \geq \epsilon d_u$ for some vertex u:
   Pick any vertex u where $\|\widehat{r}(u)\|_2 \geq \epsilon d_u$ and apply operation $\mathsf{Push}(u, \alpha)$.
3. Return $\widehat{p}$ and $\widehat{r}$.

Theorem 4. *For constants $0 < \epsilon, \alpha < 1$, and a seed vector $\widehat{s}$ with $\sum_v \|\widehat{s}(v)\|_2 \leq 1$, the algorithm $\mathsf{SharpApproximatePR}(\widehat{s}, \alpha, \epsilon)$ computes approximate PageRank vector $\widehat{p} = \widehat{\mathrm{pr}}_{\alpha,\widehat{s}-\widehat{r}}$ such that the residual vector $\widehat{r}$ satisfies $\max_v \frac{\|\widehat{r}(v)\|_2}{d_v} \leq \epsilon$ and the running time is $O(\frac{md^2\log(1/\epsilon)}{\alpha})$.*

Proof. The algorithm returns an approximate PageRank vector $\widehat{p} = \widehat{\mathrm{pr}}_{\alpha,\widehat{s}-\widehat{r}}$ when the residual vector satisfies that $\max_v \frac{\|\widehat{r}(v)\|_2}{d_v} \leq \epsilon$.

To bound the running time, we examine one fixed round of while-loop in the second step. Let T denote the total number of Push operations performed by $\mathsf{ApproximatePR}$ and let d_i denote the degree of the vertex involved in the ith Push operation. When the ith Push operation was performed, the quantity $\sum_v \|\widehat{r}(v)\|_2$ decreases at least by the amount $\alpha\xi d_i$. Since at the beginning of each while-loop $\sum_v \|\widehat{r}(v)\|_2$ is at most $2\xi\sum_{v\in V} d(v) = 2m\xi$, we have $\xi\alpha\sum_{i=1}^{T} d_i \leq 2m\xi$, which implies that $\sum_{i=1}^{T} d_i \leq \frac{2m}{\alpha}$. Since there are at most $\log(\frac{1}{\epsilon})$ rounds in the second step, the total running time is bounded by $O(\frac{md^2\log(1/\epsilon)}{\alpha})$. □

$(\widehat{p}, \widehat{r}) = \mathsf{SharpApproximatePR}(\widehat{s}, \alpha, \epsilon)$

1. Let $\xi = 1$, $\widehat{r} = \widehat{s}$ and $\widehat{p} = \mathbf{0}$.
2. While $\xi > \epsilon$:
   a. Set $\xi = \frac{\xi}{2}$.
   b. Let $\widehat{p}'$ and $\widehat{r}'$ be the output of $\mathsf{ApproximatePR}(\widehat{r}, \alpha, \xi)$.
   c. Let $\widehat{p} = \widehat{p} + \widehat{p}'$ and $\widehat{r} = \widehat{r}'$.
3. Return $\widehat{p}$ and $\widehat{r}$.

4 The Connection Resistance

Motivated by the definition of effective resistance in electrical network theory, we consider the following block matrix $\Psi = \mathbb{B}\mathbb{L}_{\mathbb{G}}^{+}\mathbb{B}^{T} \in \mathcal{F}(m, m, d; \mathbf{R})$ where $\mathbb{L}^{+}$ is the pseudo-inverse of $\mathbb{L}$. Note that for a matrix M, the *pseudo-inverse* of M is defined as the unique matrix M^{+} satisfying the following four criteria [14,25]: *(i)* $MM^{+}M = M$; *(ii)* $M^{+}MM^{+} = M^{+}$; *(iii)* $(MM^{+})^{*} = (MM^{+})$; and *(iv)* $(M^{+}M)^{*} = M^{+}M$.

We define the connection resistance $\mathbb{R}_{\mathsf{eff}}(e)$ as $\mathbb{R}_{\mathsf{eff}}(v, u) = \|\Psi(e, e)\|_2$. Note that block $\Psi(e, e)$ is a $d \times d$ matrix. If $d = 1$ or all orthogonal transformations are identity transformation, i.e. $O_{uv} = I_{d\times d}$ for all $(u, v) \in E$, then it can be shown that $\mathbb{R}_{\mathsf{eff}}(u, v)$ is reduced to the usual effective resistance $R_{\mathsf{eff}}(u, v)$ of the underlying graph G. In general, the connection resistance between the endpoints of an edge $e = (u, v)$ is not necessarily equal to its effective resistance in the underlying graph G. We will investigate the relation between the effective resistance in the underlying graphs and connection resistance for some family of connection graphs.

We consider the connection graph whose underlying graph G is a tree. Our first observation is the following Lemma.

Lemma 3. *Suppose $\mathbb{G}$ is a connection graph whose underlying graph G is a tree. The rotation matrices between u and v are $O_{uv} \in \mathsf{O}(d)$ for $(u, v) \in E$. Let $\mathbb{L}$ be the connection Laplacian of $\mathbb{G}$ and L be the Laplacian of G respectively. Then for two vertices u and v joined by a path, denoted by $(v_1 = u, v_1, \ldots, v_k = v)$, we have*

$$\mathbb{L}^{+}(u, v) = \begin{cases} L^{+}(u, v) \ \prod_{i=1}^{k-1} O_{v_i, v_{i+1}} & u \neq v, \\ L^{+}(u, v) \ I_{d\times d} & u = v. \end{cases}$$

By using the above lemma, we examine the relation between the connection resistance and the effective resistance by the following theorem and give one of its application by Corollary 1. The proof will be included in the full paper.

Theorem 5. *Suppose $\mathbb{G}$ is a connection graph whose underlying graph is a tree. If all rotations $O_{uv} \in \mathsf{SO}(d)$ for some odd number d. Then for any edge (u, v) satisfying $L^{+}_{u,v} \leq 0$, we have $\mathbb{R}_{\mathsf{eff}}(u, v) = R_{\mathsf{eff}}(u, v)$.*

Corollary 1. *For any uniform weighted path on vertices $v_1, v_2, \ldots, v_n$ with rotation matrices $O_{v_i v_{i+1}} \in \mathsf{SO}(d)$ for $1 \leq i < n$ and some odd number d, then $\mathbb{R}_{\mathsf{eff}}(v_1, v_n) = R_{\mathsf{eff}}(v_1, v_n)$.*

5 Ranking Edges by Using the Connection Resistance

A central part of a graph sparsification algorithm is the sampling technique for selecting edges. It is crucial to choose the appropriate probabilistic distribution which can lead to a sparsifier preserving *every* cut in the original graph. The following algorithm Sample is a generic sampling algorithm for a graph sparsification problem. We will sample edges using the distribution proportional to the weighted connection resistances.

$(\widetilde{\mathbb{G}} = (V, \widetilde{E}, O, \widetilde{w})) = \mathsf{Sample}(\mathbb{G} = (V, E, O, w), p', q)$

1. For every edge $e \in E$, set p_e proportional to p'_e.
2. Choose a random edge e of $\mathbb{G}$ with probability p_e, and add e to $\widetilde{\mathbb{G}}$ with edge weight $\widetilde{w}_e = \frac{w_e}{qp_e}$. Take q samples independently with replacement, summing weights if an edge is chosen more than once.
3. Return $\widetilde{\mathbb{G}}$.

Theorem 6. *For a given connection graph $\mathbb{G}$ and some positive $\xi > 0$, we consider $\widetilde{\mathbb{G}} = \mathsf{Sample}(G, p', q)$, where $p'_e = w_e \mathbb{R}_{\mathsf{eff}}(e)$ and $q = \frac{4nd \log(nd) \log(1/\xi)}{\epsilon^2}$. Suppose $\mathbb{G}$ and $\widetilde{\mathbb{G}}$ have connection Laplacian $\mathbb{L}_{\mathbb{G}}$ and $\mathbb{L}_{\widetilde{\mathbb{G}}}$ respectively. Then with probability at least ξ, for any function $\forall f : V \to \mathbf{R}^d$, we have*

$$(1-\epsilon) f \mathbb{L}_{\mathbb{G}} f^T \leq f \mathbb{L}_{\widetilde{\mathbb{G}}} f^T \leq (1+\epsilon) f \mathbb{L}_{\mathbb{G}} f^T. \tag{3}$$

Before proving Theorem 6, we need the following two lemmas, in particular concerning the matrix $\Lambda = \mathbb{W}^{1/2} \mathbb{B} \mathbb{L}_{\mathbb{G}}^{+} \mathbb{B}^T \mathbb{W}^{1/2}$. We omit their proofs here.

Lemma 4. *(i) Λ is a projection matrix, i.e. $\Lambda^2 = \Lambda$. (ii) The eigenvalues of Λ are 1 with multiplicity at most nd and 0 otherwise. (iii) $\Lambda(e,e) = \Lambda(\cdot, e)^T \Lambda(\cdot, e)$.*

To show that $\widetilde{\mathbb{G}} = (V, \widetilde{E}, O, \widetilde{w})$ is a good sparsifier for $\mathbb{G}$ satisfying (3), we need to show that the quadratic forms $f \mathbb{L}_{\widetilde{\mathbb{G}}} f^T$ and $f \mathbb{L}_{\mathbb{G}} f^T$ are close. By applying similar methods as in [33], we reduce the problem of preserving $f \mathbb{L}_{\mathbb{G}} f^T$ to that of $g \Lambda g^T$ for some function g. We consider the diagonal matrix $\mathbb{S} \in \mathcal{F}(m, m, d; \mathbf{R})$, where the diagonal blocks are scalar matrices given by $\mathbb{S}(e,e) = \frac{\widetilde{w}_e}{w_e} I_{d \times d} = \frac{N_e}{qp_e} I_{d \times d}$ and N_e is the number of times an edge e is sampled.

Lemma 5. *Suppose $\mathbb{S}$ is a nonnegative diagonal matrix such that $\|\Lambda \mathbb{S} \Lambda - \Lambda \Lambda\|_2 \leq \epsilon$. Then, $\forall f : V \to \mathbf{R}^d, (1-\epsilon) f \mathbb{L}_{\mathbb{G}} f^T \leq f \mathbb{L}_{\widetilde{\mathbb{G}}} f^T \leq (1+\epsilon) f \mathbb{L}_{\mathbb{G}} f^T$.*

We also require the following concentration inequality in order to prove our main theorems. Previously, various matrix concentration inequalities have been derived by many authors including Achiloptas [1], Cristofies-Markström [10], Recht [26], and Tropp [34]. Here we will use the simple version that is proved in [8].

Theorem 7. *Let X_i be independent random symmetric $k \times k$ matrices, $X_i \geq 0$, $\|X_i\|_2 \leq M$ for all i a.s. Then for every $\epsilon \in (0,1)$ we have*

$$\Pr\left[\left\|\sum_i X_i - \mathbf{E}\left[\sum_i X_i\right]\right\|_2 > \epsilon \sum_i \|\mathbf{E}[X_i]\|_2\right] \leq k \exp\left(-\frac{\epsilon^2 \sum_i \|\mathbf{E}[X_i]\|_2}{4M}\right).$$

Proof (of Theorem 6). Our algorithm samples edges from $\mathbb{G}$ independently with replacements, with probabilities p_e proportional to $w_e \mathbb{R}_{\mathsf{eff}}(e)$. Note that sampling q edges from $\mathbb{G}$ corresponds to sampling q columns from Λ. So we can write

$$\Lambda \mathbb{S} \Lambda = \sum_e \Lambda(\cdot, e) \mathbb{S}(e,e) \Lambda(\cdot, e)^T = \sum_e \frac{N_e}{qp_e} \Lambda(\cdot, e) \Lambda(\cdot, e)^T = \frac{1}{q} \sum_{i=1}^{q} y_i y_i^T$$

for block matrices $y_1, \ldots, y_q \in R^{nd \times d}$ drawn independently with replacements from the distribution $y = \frac{1}{\sqrt{p_e}} \Lambda(\cdot, e)$ with probability p_e. Now, we can apply Theorem 7. The expectation of yy^T is given by $\mathbf{E}\left[yy^T\right] = \sum_e p_e \frac{1}{p_e} \Lambda(\cdot, e)\Lambda(\cdot, e)^T = \Lambda$ which implies that $\left\|\mathbf{E}\left[yy^T\right]\right\|_2 = \|\Lambda\|_2 = 1$. We also have a bound on the norm of $y_i y_i^T$: $\left\|y_i y_i^T\right\|_2 \leq \max_e \left(\frac{\left\|\Lambda(\cdot,e)^T \Lambda(\cdot,e)\right\|_2}{p_e}\right) = \max_e \left(\frac{w_e \mathbb{R}_{\mathsf{eff}}(e)}{p_e}\right)$. Since the probability p_e is proportional to $w_e \mathbb{R}_{\mathsf{eff}}(e)$, i.e. $p_e = \frac{w_e \mathbb{R}_{\mathsf{eff}}(e)}{\sum_e w_e \mathbb{R}_{\mathsf{eff}}(e)} = \frac{\|\Lambda(e,e)\|_2}{\sum_e \|\Lambda(e,e)\|_2}$, we have $\left\|y_i y_i^T\right\|_2 \leq \sum_e \|\Lambda(e,e)\|_2 \leq \sum_e \mathrm{Tr}\left(\Lambda(e,e)\right) = \mathrm{Tr}\left(\Lambda\right) \leq nd$. To complete the proof, by setting $q = \frac{4nd \log(nd) \log(1/\xi)}{\epsilon^2}$ and the fact that dimension of yy^T is nd, we have

$$\Pr\left[\left\|\frac{1}{q}\sum_{i=1}^{q} y_i y_i^T - \mathbf{E}\left[yy^T\right]\right\|_2 > \epsilon\right] \leq nd \exp\left(-\frac{\epsilon^2 \sum_{i=1}^{q} \left\|\mathbf{E}\left[y_i y_i^T\right]\right\|_2}{4nd}\right)$$
$$\leq nd \exp\left(-\frac{\epsilon^2 q}{4nd}\right) \leq \xi$$

for some constant $0 < \xi < 1$. Thus, the theorem follows. □

The oversampling Theorem in [23] can be modified and stated as follows.

Theorem 8 (Oversampling). *For a given connection graph $\mathbb{G}$ and some positive $\xi > 0$, we consider $\widetilde{\mathbb{G}} = \mathsf{Sample}(G, p', q)$, where $p'_e = w_e \mathbb{R}_{\mathsf{eff}}(e), t = \sum_{e \in E} p'_e$ and $q = \frac{4t \log(t) \log(1/\xi)}{\epsilon^2}$. Suppose $\mathbb{G}$ and $\widetilde{\mathbb{G}}$ have connection Laplacian $\mathbb{L}_{\mathbb{G}}$ and $\mathbb{L}_{\widetilde{\mathbb{G}}}$ respectively. Then with probability at least ξ, for all $f : V \to \mathbf{R}^d$, we have $(1-\epsilon) f \mathbb{L}_{\mathbb{G}} f^T \leq f \mathbb{L}_{\widetilde{\mathbb{G}}} f^T \leq (1+\epsilon) f \mathbb{L}_{\mathbb{G}} f^T$.*

Now let us consider a variation of the connection resistance denoted by $\overline{\mathbb{R}_{\mathsf{eff}}}(e) = \mathrm{Tr}\left(\Psi(e,e)\right)$. Clearly, we have $\overline{\mathbb{R}_{\mathsf{eff}}}(e) = \mathrm{Tr}\left(\Psi(e,e)\right) \geq \|\Psi(e,e)\|_2 = \mathbb{R}_{\mathsf{eff}}(e)$ and $\sum_e w_e \overline{\mathbb{R}_{\mathsf{eff}}}(e) = \sum_e \mathrm{Tr}\left(\Lambda(e,e)\right) = \mathrm{Tr}\left(\Lambda\right) \leq nd$. Using Theorem 8, we have the following.

Corollary 2. *For a given connection graph $\mathbb{G}$ and some positive $\xi > 0$, we consider $\widetilde{\mathbb{G}} = \mathsf{Sample}(G, p', q)$, where $p'_e = w_e \overline{\mathbb{R}_{\mathsf{eff}}}(e)$ and $q = \frac{4nd \log(nd) \log(1/\xi)}{\epsilon^2}$. Suppose $\mathbb{G}$ and $\widetilde{\mathbb{G}} = \mathsf{Sample}(G, p', q)$ have connection Laplacian $\mathbb{L}_{\mathbb{G}}$ and $\mathbb{L}_{\widetilde{\mathbb{G}}}$ respectively. Then with probability at least ξ, for all $f : V \to \mathbf{R}^d$, we have $(1-\epsilon) f \mathbb{L}_{\mathbb{G}} f^T \leq f \mathbb{L}_{\widetilde{\mathbb{G}}} f^T \leq (1+\epsilon) f \mathbb{L}_{\mathbb{G}} f^T$.*

References

1. Achlioptas, D.: Database-friendly random projections. In: Proceedings of the 20th ACM Symposium on Principles of Database Systems, pp. 274–281 (2001)
2. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building Rome in a Day. In: Proceedings of the 12th IEEE International Conference on Computer Vision, pp. 72–79 (2009)

3. Andersen, R., Chung, F., Lang, K.: Local graph partitioning using pagerank vectors. In: Proceedings of the 47th IEEE Symposium on Founation of Computer Science, pp. 475–486 (2006)
4. Anderson, G.W., Guionnet, A., Zeitouni, O.: An introduction to random matrices. Cambridge University Press (2010)
5. Benczúr, A.A., Karger, D.R.: Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In: Proceedings of the 28th ACM Symposium on Theory of Computing, pp. 47–55 (1996)
6. Berkhin, P.: Bookmark-coloring approach to personalized pagerank computing. Internet Mathematics 3, 41–62 (2006)
7. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems 30(1-7), 107–117 (1998)
8. Chung, F., Radcliffe, M.: On the spectra of general random graphs. Electronic Journal of Combinatorics 18(1), 215–229 (2011)
9. Chung, F., Sternberg, S.: Laplacian and vibrational spectra for homogeneous graphs. J. Graph Theory 16, 605–627 (1992)
10. Cristofides, D., Markström, K.: Expansion properties of random Cayley graphs and vertex transitive graphs via matrix martingales. Random Structures Algorithms 32(8), 88–100 (2008)
11. Cucuringu, M., Lipman, Y., Singer, A.: Sensor network localization by eigenvector synchronization over the Euclidean group. ACM Transactions on Sensor Networks (in press)
12. Firat, A., Chatterjee, S., Yilmaz, M.: Genetic clustering of social networks using random walks. Computational Statistics and Data Analysis 51(12), 6285–6294 (2007)
13. Fouss, F., Pirotte, A., Renders, J.-M., Saerens, M.: Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. Knowledge and Data Engineering 19(3), 355–369 (2007)
14. Golub, G.H., Van Loan, C.F.: Matrix computations, 3rd edn., pp. 257–258. Johns Hopkins, Baltimore (1996)
15. Hadani, R., Singer, A.: Representation theoretic patterns in three dimensional cryo-electron microscopy I - the intrinsic reconstitution algorithm. Annals of Mathematics 174(2), 1219–1241 (2011)
16. Herbster, M., Pontil, M., Rojas, S.: Fast Prediction on a Tree. In: Proceedings of the Neural Information Processing Systems Foundation, pp. 657–664 (2008)
17. Jeh, G., Widom, J.: Scaling personalized web search. In: Proceedings of the 12th World Wide Web Conference WWW, pp. 271–279 (2003)
18. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer Series in Statistics (2002)
19. Karger, D.R.: Random sampling in cut, flow, and network design problems. Mathematics of Operations Research 24(2), 383–413 (1999)
20. Karger, D.R.: Using randomized sparsification to approximate minimum cuts. In: Proceedings of the 15th ACM Symposium on Discrete Algorithms, pp. 424–432 (1994)
21. Karger, D.R.: Minimum cuts in near-linear time. Journal of the ACM 47(1), 46–76 (2000)
22. Kirchhoff, F.: Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme geführt wird. Ann. Phys. Chem. 72, 497–508 (1847)

23. Koutis, I., Miller, G.L., Peng, R.: Approaching Optimality for Solving SDD Linear Systems. In: Proceedings of 51st IEEE Symposium on Foundations of Computer Science, pp. 235–244 (2010)
24. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE International Conference on Computer Vision, pp. 1150–1157 (1999)
25. Penrose, R.: A generalized inverse for matrices. Cambridge Philosophical Society 51, 406–413 (1955)
26. Recht, B.: Simpler approach to matrix completion. Journal of Machine Learning Research (to appear)
27. Singer, A.: Angular synchronization by eigenvectors and semidefinite programming. Applied and Computational Harmonic Analysis 30(1), 20–36 (2011)
28. Singer, A., Zhao, Z., Shkolnisky, Y., Hadani, R.: Viewing angle classification of cryo-electron microscopy images using eigenvectors. SIAM Journal on Imaging Sciences 4(2), 723–759 (2011)
29. Singer, A., Wu, H.-T.: Vector Diffusion Maps and the Connection Laplacian. Communications on Pure and Applied Mathematics (to appear)
30. Spielman, D.A., Teng, S.-H.: Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In: Proceedings of the 36th ACM Symposium on Theory of Computing, pp. 81–90 (2004)
31. Spielman, D.A., Teng, S.-H.: Nearly-linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems (2006), http://www.arxiv.org/abs/cs.NA/0607105
32. Spielman, D.A., Teng, S.-H.: Spectral Sparsification of Graphs (2010), http://arxiv.org/abs/0808.4134
33. Spielman, D.A., Srivastava, N.: Graph sparsification by effective resistances. In: Proceedings of 40th ACM Symposium on Theory of Computing, pp. 563–568 (2008)
34. Tropp, J.: User-Friendly Tail Bounds for Sums of Random Matrices, http://arxiv.org/abs/1004.4389
35. Vu, V.: Spectral norm of random matrices. Combinatorica 27(6), 721–736 (2007)
36. Wigderson, A., Xiao, D.: Derandomizing the Ahlswede-Winter matrix-valued Chernoff bound using pessimistic estimators, and applications. Theory of Computing 4(1), 53–76 (2008)

A Game-Theoretic Model of Attention in Social Networks

Ashish Goel* and Farnaz Ronaghi**

Department of Management Science and Engineering
Stanford University, Stanford, CA
{ashishg,farnaaz}@stanford.edu

Abstract. We model the economics of producing content in online social networks such as Facebook and Twitter. We propose a game-theoretic model within which we quantify inefficiencies from contributions by strategic users in online environments. Attention and information are assumed to be the main motivation for user contributions. We treat attention as a mechanism for sharing the profit from consuming information and introduce a general framework for analyzing dynamics of contributions in online environments. We analyze the proposed model and identify conditions for existence and efficient computation of pure-strategy Nash equilibrium.

We prove a bicriteria bound on the price of anarchy; in particular we show that the social welfare from central control over level of contribution by users is no larger than the social welfare from strategic agents with twice as large consumption utilities. We then construct and analyze a family of production games that have an arbitrarily large price of anarchy. We also prove non-robustness of the price of anarchy for a particular instance of the introduced family, establishing a distinction between the games studied here and network congestion games.

1 Introduction

Social networking websites allow users to sign up and keep in touch with others by friending them. Users are allowed to post short status updates, photos, videos and links depending on the social network. Despite the differences between these networks, one feature is common to many of them: Users see a linear news feed reflecting a chronologically sorted ordering of posts from friends whenever she logs into each of these sites [1]. Given that the main advantage of these sites is the convenience of getting a quick update, the balance of information from friends in the feed becomes important in determining the value the user will gain from the update [1].

* This research was supported in part by grant 0904325. Part of the research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053.

** This research was supported by a Stanford Engineering Fellowship.

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 78–92, 2012.

A user's news feed can easily get flooded by contributions from an active friend who attracts most of the attention in a social cluster. It has been observed that there is a strong correlation between lack of attention and a user's decision to stop contributing; also, the more a user contributes the more attention she tends to receive. Getting attention paid to one's contributions is a form of value [2] and users are willing to forsake financial gain for it [5]. Attention was also shown to spur further contributions in video sharing [6] and blogging [9]; moreover it was introduced as the main ingredient in successful peer production websites [16]. Taking attention and information as the foremost motivation for user activity in social networks, high participation from a subset of friends of user u makes u's attention a scarce resource for other friends of u, which in turn can make these other friends less likely to contribute and hence get even less attention. We model this dynamics of user contributions in online social networks in a game-theoretic setting and quantify the inefficiencies from strategic user participation.

Our Contributions. Our main contribution is proposing and analyzing a game-theoretic model of user participation in online social networks. We refer to the proposed model by "the general production game" throughout the paper. The main elements of our model are users that choose the level/quality of their contributions. Updates from a user are viewed as a bundle in her friends' news feed, ignoring the order in which information arrives. We assume that the network structure is fixed. Users are strategic in selecting their level of contribution but they are not strategic in selecting what information or how much information they consume. They are assumed to be utility-maximizing agents who derive utility from attention and information simultaneously. Moreover they incur a non-negative cost for producing content. Producing more information needs more effort and there is no cost for inaction. Users prefer more information to less but they have limited attention capacity so their utility from consumption has diminishing returns. We formalize these assumptions in section 2 and refer to them as "general assumptions" throughout the paper.

While users consume content, they pay attention to those who generated it. In other words, users share the profit from consuming information with those who produced it in the first place. We introduce a general utility scheme and analyze three instances of it: Proportional, Incremental, and Shapley utility models. The proportional utility model splits user utility from consumption among friends proportional to their contribution. The incremental utility model regards the marginal consumption utility from a user's contribution as the attention she receives. The Shapley utility is similar to the incremental utility; however the incremental utility from a user's information is computed with respect to only the information from "earlier" users in a random permutation [14].

We observe that under general assumptions, a unique pure-strategy Nash equilibrium exists for our proposed game and is characterized via the Karush Kuhn Tucker(KKT) conditions. Our main result states necessary conditions such that the social welfare from central control over user production, obligating users to act according to social interest, is no greater than the social welfare from

strategic production when users have twice as large consumption utilities. We have proved that under general assumption both Shapley and incremental utility models satisfy this necessary conditions. Our results suggest that improvements in user experience, such as ease of information discovery and spam detection, can compensate for inefficiencies from strategic behavior in online environments. We also analyze the proposed game for simple interesting cases in section 4.2 and show that the price of anarchy can be arbitrarily large regardless of the number of agents and network structure.

Our main result is similar to the bicriteria bound for network congestion games presented in [13], and superficially, it might seem that earlier proof techniques should directly imply our results. However, both atomic and non-atomic network congestion games are proved to have a "robust pure Nash price of anarchy" in the sense of [12]. We show (in section 5) that one particular instance of our proposed production game does not admit a "robust price of anarchy", establishing a distinction between the proposed game and network congestion games.

Related Work. Attention affects the propagation of information in social networks, determining the effectiveness of advertising and viral marketing. Many different approaches have been taken to study attention, including empirical studies of dynamics of attention [16,15], empirical and game-theoretic analysis of the impact of attention in marketing [1,8], and studies on the impact of attention(exposure) in high-quality user-generated content [4,7].

Wu et al. [15] study dynamics of collective attention for a piece of story on digg.com, and they propose a stochastic model that predicts the amount of attention a story gets by incorporating novelty of the story as a decaying factor. In a follow-up paper [16], the authors study feedback loops of attention in peer production websites such as youtube and digg.com. They empirically show a strong correlation between lack of attention and users' decision to stop contributing.

Borgs et al. [1] address the asymmetry of online relationships in social networks such as Facebook and Twitter. They model the social network as a complete bi-partite graph where users are either producers or consumers of information, and edges of the graph have non-negative weights that represent the quality of updates from a particular producer (in their setting, an advertiser) to a particular consumer. Users receive update at a rate chosen by advertisers and can adopt two types of behavior in response to an excessively high rate: unfollowing and disengagement. They study the set of ties that are realized and stabilized over time in the followership and engagement models.

Ghosh et al. [4] study the problem of high-quality user generated content in online crowd-sourcing websites. They propose a game-theoretic model that incorporates the quality of content and study mechanisms of splitting attention (exposure time) to incentivize high-quality content and maximize user participation. They independently propose a proportional model for splitting the exposure time among contributions, and show that the proportional mechanism elicits both high quality and high participation in equilibrium.

Unlike previous work, we directly model attention as a scarce resource, as well as the cost of creating new and useful pieces of information. Attention is shown to be the foremost motivator in peer production websites such as digg.com and YouTube [16]; it is also one of the main motivators in online social networks such as Facebook and Twitter. Information and attention are duals in social networks hence, instead of modeling information and attention as two separate entities, we model attention as a mechanism for sharing the profit from consuming information. We propose a model in which all agents are strategic and derive utility simultaneously from consuming and producing information. Agents are connected to one another in either a symmetric or asymmetric network, i.e. we don't make any assumptions on network structure. Also we can re-interpret the only decision variable in our model from level of contributions to quality of contributions or rate of updates and our results remain valid.

2 Model

Every user as a member of the social network has friendship relations with at least one other user. User a produces x_a units of information which appear on her friends' feed. She perceives $y_a := \sum_{b \sim a} q_{ba} x_b$ units of information from her feed where $b \sim a$ means a is a friend of b and q_{ba} represents a's interest in b's updates. User a pays attention to user b when she consumes the information produced by her. We model such exchange of attention and information between users of a social network in a utilitarian framework where every user a incurs an increasing cost $c_a(x_a)$ for producing information and derives increasing utility $f_a(y_a)$ from consuming it. Users also derive positive utility from receiving attention. We denote the amount of attention user a receives from her friend b by $t_{a,b}(\boldsymbol{x})$. So an arbitrary agent a derives

$$u_a(\boldsymbol{x}) = f_a(y_a) - c_a(x_a) + t_a(\boldsymbol{x}) \tag{1}$$

utility from her network of friends where $t_a(\boldsymbol{x}) = \sum_{b \sim a} t_{a,b}(\boldsymbol{x})$ and $\boldsymbol{x}$ represents the strategy vector.

We analyze the proposed utility scheme and state conditions under which our bicriteria bound holds. We make three general assumptions throughout the paper. First, we assume that consumption utility $f_a(y_a)$ is a differentiable, concave, and increasing function for every agent and $f_a(0) = 0$. Second, we assume that production cost $c_a(x_a)$ is a differentiable, increasing, and strictly convex function for all agents and $c_a(0) = 0$. Third, we assume $t_{a,b}(\boldsymbol{x})$ is increasing in x_a and $t_{a,b}(0, \boldsymbol{x}_{-a}) = 0$ where $\boldsymbol{x}_{-a}$ is the strategy vector including production level for all users except for user a. We also study three particular instances of our general model denoted by *incremental* utility, *Shapley* utility, and *proportional* utility. Each instance corresponds to a different method of splitting user attention among friends in online environments.

Incremental utility models the amount of attention a user receives by the sum of consumption utility margins she imposes on her friends in the social network. Formally, the incremental utility models the attention user a receives by

$$t_a(\boldsymbol{x}) = \sum_{b \sim a} f_b(y_b) - f_b(y_b - q_{ab}x_a). \tag{2}$$

Shapely utility implements the Shapley cost sharing scheme as a profit sharing mechanism to split the profit from consuming information among those who originally created it. Shapley mechanism assigns

$$t_a(\boldsymbol{x}) = \sum_{b \sim a} \sum_{\sigma \in \mathbb{S}_{N(b)}} \frac{1}{d_b!} f_b\Big(\sum_{i=1}^{\sigma^{-1}(a)} q_{\sigma(i)b}x_{\sigma(i)}\Big) - f_b\Big(\sum_{i=1}^{\sigma^{-1}(a)-1} q_{\sigma(i)b}x_{\sigma(i)}\Big) \tag{3}$$

units of utility from attention to every user a, where $\mathbb{S}_{N(b)}$ denotes the set of all permutations of b's friends; $N(b)$ is the set of b's friends and $d_b = |N(b)|$. Marginal profit terms inside the second sum are known as *ordered marginals*. The Shapley value is defined as the expectation of *ordered marginals* over a uniform distribution on all arrival permutations [14]. Shapley utility arises as a natural attention sharing mechanism when incoming updates are shown at random order in the feed provided to users.

Proportional utility is an alternate way of splitting user attention among friends in a social network. The amount of attention a user receives is modeled as the weighted sum of friends' consumption utilities where the weights are equal to the proportion of the user's contribution. Formally, the amount of attention a user receives in the network is

$$t_a(\boldsymbol{x}) = \sum_{b \sim a} \frac{q_{ab}x_a}{y_b} f_b(y_b). \tag{4}$$

Friends with more high quality updates receive more attention in the proportional mechanism. Similar to the Shapley utility, proportional mechanism arises as a natural profit sharing scheme when updates are viewed in a random order. Position bios is a well-established phenomenon in online social networks; items shown higher in user's update feed have a higher probability of receiving actions. We ignore the impacts of position bios on the distribution of attention throughout the paper.

3 Existence and Computability of Nash Equilibrium

We determine sufficient conditions such that our general utility model admits pure-strategy Nash equilibrium. Moreover we identify exact potential functions for incremental and Shapley utility models; existence of exact potential functions implies convergence of the natural Nash dynamics to a pure-strategy equilibrium.

Strategy vector $\boldsymbol{x}$ is a pure Nash equilibrium if every player a chooses her strategy x_a to maximize $u_a(x_a, \boldsymbol{x}_{-a})$. Rosen's theorem [10] for concave n-player non-cooperative games establishes existence of a unique pure Nash equilibrium and KKT conditions characterize it.

Proposition 3.1. *Our proposed general production game admits a unique pure-strategy Nash equilibrium if* $|\frac{\partial^2 t_a(\boldsymbol{x})}{\partial x_a^2} - \frac{\partial^2 c_a(x_a)}{\partial x_a^2}| > \epsilon$ *for constant* $\epsilon > 0$*, and general assumptions hold. Strategy vector* $\boldsymbol{x}$ *is a Nash equilibrium strategy if and only if for every player* a*,*

$$x_a\left(\frac{\partial t_a(\boldsymbol{x})}{\partial x_a} - \frac{\partial c_a(x_a)}{\partial x_a}\right) = 0. \tag{5}$$

Proof. General assumptions guarantee strict concavity of the utility function for very player a. The strategy space can be reduced to a convex and compact set; since $|\frac{\partial^2 t_a(\boldsymbol{x})}{\partial x_a^2} - \frac{\partial^2 c_a(x_a)}{\partial x_a^2}| > \epsilon$ we can define upper bounds on values of x_a. We can apply Rosen's theorem for concave n-player noncooperative games with convex and compact strategy space [10] to conclude existence of Nash equilibria for proposed utility game.

At Nash equilibrium every player solves the following optimization problem:

$$\begin{aligned} \text{Maximize} \quad & f_a(y_a) - c_a(x_a) + t_a(\boldsymbol{x}) \\ \text{Subject to:} \quad & x_a \geq 0. \end{aligned}$$

KKT conditions, stated in (5), determine necessary and sufficient conditions for optimality. □

Although proposition 3.1 proves existence of Nash equilibrium for the proposed production game, it fails to establish Nash equilibrium as naturally arising from user behavior in online social networks. Proposition 3.2 identifies exact potential functions for incremental and Shapley utility models. Existence of exact potential functions alludes that the natural Nash Dynamics, in which players iteratively play best response; converges to a pure Nash Equilibrium for the game although convergence might take exponential time [11].

Proposition 3.2. *The Incremental utility and Shapley utility games admit an exact potential function defined correspondingly as*

$$\Phi^I(\boldsymbol{x}) = \sum_a \left\{ f_a(y_a) - c_a(x_a) \right\}, \tag{6}$$

and

$$\Phi^S(\boldsymbol{x}) = \sum_a -c_a(x_a) + \sum_{S \subseteq N(a), s=|S|} \frac{1}{s\binom{d_a}{s}} f_a\Big(\sum_{c \in s} q_{ca} x_c\Big). \tag{7}$$

Proof. Similar to the proof statement we use a superscript of I to denote the incremental model and a superscript of S to denote the Shapley utility model. It is easy to observe that

$$u_a^I(x'_a, \boldsymbol{x}_{-a}) - u_a^I(\boldsymbol{x}) = \Phi^I(x'_a, \boldsymbol{x}_{-a}) - \Phi^I(\boldsymbol{x}),$$

so by definition $\Phi^I(\boldsymbol{x})$ is an exact potential function for the incremental game.

User Shapley utility $u_a^S(\boldsymbol{x}_a)$ can be rewritten as

$$u_a^S(\boldsymbol{x}) = f_a(y_a) - c_a(x_a) + \sum_{b \sim a} \sum_{S \subseteq N(b), s=|S|} \frac{1}{s\binom{d_b}{s}} \{ f_b(\sum_{c \in S} q_{cb} x_c) - f_b(\sum_{c \in S, c \neq a} q_{cb} x_c) \}. \tag{8}$$

Consider a function of the form

$$\Phi^S(x) = \sum_a -c_a(x_a) + \sum_{S \subseteq N(a), s=|S|} \kappa_{a,s} f_a(\sum_{c \in S} q_{ca} x_c)$$

with $\kappa_{a,s} = \frac{1}{s\binom{d_a}{s}}$. This is an exact potential function for the Shapley utility game; if user a switches strategies from x_a to x'_a, then for all of a's friends the consumption utility $f_b(\sum_{c \in S} q_{cb} x_c)$ changes in all subsets containing a and the change in potential function is equal to the change in agent a's utility given in (8). □

While we have not proved convergence bounds with best-response dynamics observe that under general assumptions proposed potential functions are strictly concave; hence the unique Nash equilibrium of the incremental and Shapley utility games can be computed in polynomial time.

We prove a bicriteria bound on the price of anarchy in the rest of the paper. We state conditions under which such bounds hold for the general production games, and apply our bounds to the Shapley and incremental utility models.

4 Analysis of the Price of Anarchy

Price of anarchy quantifies the degradation in the efficiency of a game due to strategic behavior of participating players [13]. The pure Nash price of anarchy is defined as the ratio of the welfare for the worst pure Nash equilibrium and the optimum welfare where the welfare function $W(\boldsymbol{x})$ is defined as the total utility of all agents. The optimum welfare refers to a setting where a central authority obligates users to behave according to the socially-optimal strategy vector $\boldsymbol{x^*} = \arg\max_{\boldsymbol{x}} W(x)$. We derive a bicriteria bound on the price of anarchy that compares $W(x^*)$ with equilibrium welfare in an augmented social network. We prove that inefficiencies from players' strategic behavior can be compensated by "doubling" the happiness function. In section 4.2, we analyze the price of anarchy for the simplest class of production games without augmentation. We show that the price of anarchy can be arbitararily big even theough these production games have linear happiness and polynomial cost functions.

4.1 A Bicriteria Bound on the Price of Anarchy

We derive a bicriteria bound on the price of anarchy for the proposed general production game. We compare the optimal social welfare in an online environment against the equilibrium welfare in an augmented environment, where the

happiness function is twice as large. We show that the equilibrium welfare in this augmented version is at least as large as the optimal welfare for the original social network. Our result relies on the existence of an exact potential function; it also requires correctness of certain inequalities. We show that both these inequalities hold for Shapley and incremental utility models.

This section requires a more detailed notation since we work with two utility models simultaneously. We introduce the notation first and then state our result formally. Define $g_a(x) = 2f_a(x)$ for all agents a, $W_g(\boldsymbol{x})$ denotes the social welfare for the general production game with consumption utilities $g_a(x)$; similarly $W(\boldsymbol{x})$ denotes the social welfare for the general production game with consumption utilities $f_a(x)$. We differentiate between the incremental and the Shapley utility models with a superscript of I for the former and superscript of S for the latter. We distinguish the social optimum from the Nash equilibrium by a superscript of $*$ for social optimum and a superscript of e for the Nash equilibrium. For example, $\boldsymbol{x}^{I,e}$ represents the equilibrium strategy vector for the incremental game where consumption utility for all agents a is equal to $f_a(x)$ and $\boldsymbol{x}_g^{I,*}$ denotes the socially optimal strategy for the same game where consumption utility for all agents a is $g_a(x)$.

We first state our main result in theorem 4.1; our result compares equilibrium social welfare in an augmented social network against the social optimum in the original one. We consider improved user experience as the source of augmentation in the social network and model it by defining a new happiness function $g_a(x) = 2f_a(x)$. We next show that our result holds for the Shapley and incremental utility models in propositions 4.3 and 4.2.

Theorem 4.1. *Let $\boldsymbol{x}_g^e$ denote the equilibrium of a general production game with happiness function $g_a(x) = 2f_a(x)$ then $W_g(\boldsymbol{x}_g^e) \geq W(\boldsymbol{x}^*)$ if:*

1. *the general assumptions hold,*
2. *$\forall a$, $|\frac{\partial^2 t_a(\boldsymbol{x})}{\partial x_a^2} - \frac{\partial^2 c_a(x_a)}{\partial x_a^2}| > \epsilon$ for constant $\epsilon > 0$,*
3. *the game admits an exact potential function $\phi(\boldsymbol{x})$, and*
4. *for all valid strategy vectors $\boldsymbol{x}$, $W_g(\boldsymbol{x}) \geq \phi_g(\boldsymbol{x})$ and $\phi_g(\boldsymbol{x}) \geq W(\boldsymbol{x})$.*

Proof. We would like to show that $W_g(\boldsymbol{x}_g^e) \geq W(x^*)$. Assumptions one and two guarantee existence of a unique pure-strategy equilibrium $\boldsymbol{x}_g^e$. We instantiate assumption four with $\boldsymbol{x} = \boldsymbol{x}_g^e$ so $W_g(\boldsymbol{x}_g^e) \geq \phi_g(\boldsymbol{x}_g)$. The equilibrium strategy $\boldsymbol{x}_g$ maximizes $\phi_g(\boldsymbol{x})$ because $\phi_g(\boldsymbol{x})$ is an exact potential function for the general production game, so $\phi_g(\boldsymbol{x}_g^e) \geq \phi_g(\boldsymbol{x}^*)$. We can instantiate assumption four once more with $\boldsymbol{x} = \boldsymbol{x}^*$, obtaining $\phi_g(\boldsymbol{x}^*) \geq W(\boldsymbol{x}^*)$ which concludes the proof. □

Our bicriteria bound suggests that any inefficiency from user strategic behavior in an online environment can be compensated by improvements in user experience. Every different choice of $t_a(\boldsymbol{x})$ corresponds to a different attention sharing mechanism. Not all attention sharing mechanisms admit an exact potential function, e.g. the proportional production game. We identified exact potential functions for the incremental and Shapley utility models in proposition 3.2; we only need to prove that assumption four from theorem 4.1 holds.

Proposition 4.2. *Under general assumptions,* $W_g^I(\boldsymbol{x}) \geq \phi_g^I(\boldsymbol{x})$ *and* $\phi_g^I(\boldsymbol{x}) \geq W^I(\boldsymbol{x})$ *for all valid strategy vectors* $\boldsymbol{x}$*.*

Proof. The potential function for the incremental utility game is given according to (6) and

$$W^I(\boldsymbol{x}) = \sum_a \left\{ f_a(y_a) - c_a(x_a) + \sum_{b \sim a} f_b(y_b) - f_b(y_b - q_{ab}x_a) \right\}. \tag{9}$$

Since $f_a(x)$ is increasing, $\sum_a \sum_{b\sim a} \{f_b(y_b) - f_b(y_b - q_{ab}x_a)\} \geq 0$, thus $W_g^I(\boldsymbol{x}) \geq \phi_g^I(\boldsymbol{x})$.

Two observations prove the second part of the proposition. First, $W^I(\boldsymbol{x})$ can be expanded and rewritten as:

$$W(\boldsymbol{x}) = \sum_a \left\{ f_a(y_a) - c_a(x_a) + \sum_{b \sim a} f_a(y_a) - f_a(y_a - q_{ba}x_b) \right\}$$

Second, since $f_a(x)$ is concave and increasing and $y_a = \sum_{b\sim a} q_{ba}x_b$;

$$f_a(y_a) \geq \sum_{b \sim a} f_a(y_a) - f_a(y_a - q_{ba}x_b).$$

So $\phi_g^I(\boldsymbol{x}) \geq W^I(\boldsymbol{x})$. □

Similarly we use concavity of user happiness function to show that assumption four also holds for the Shapley utility game.

Proposition 4.3. *Under general assumption,* $W_g^S(\boldsymbol{x}) \geq \phi_g^S(\boldsymbol{x})$ *and* $\phi_g^S(\boldsymbol{x}) \geq W^S(\boldsymbol{x})$ *for all valid strategy vectors* $\boldsymbol{x}$*.*

Proof. The social welfare function for Shapley utility game with consumption utility $g_a(x)$ is

$$W_g^S(\boldsymbol{x}) = \sum_a g_a(y_a) - c_a(x_a) + \sum_{b\sim a} \sum_{S \subseteq N(b), s=|S|} \frac{1}{s\binom{d_b}{s}} \{g_b(\sum_{c\in S} q_{cb}x_c) - g_b(\sum_{c\in S, c\neq a} q_{cb}x_c)\} \tag{10}$$

and

$$\Phi_g^S(\boldsymbol{x}) = \sum_a -c_a(x_a) + \sum_{S\subseteq N(a), s=|S|} \frac{1}{s\binom{d_a}{s}} g_a(\sum_{c\in S} q_{ca}x_c).$$

Note that $g_a(\sum_{c\in s} q_{ca}x_c) \leq g_a(y_a)$ since $g_a(x)$ is an increasing function so

$$\Phi_g^S(\boldsymbol{x}) \leq \sum_a -c_a(x_a) + \sum_{S\subseteq N(a), s=|S|} \frac{1}{s\binom{d_a}{s}} g_a(y_a). \tag{11}$$

The second term in (11) is equal to $g_a(y_a)$ and the last term in (10) is positive, thus $W_g^S(\boldsymbol{x}) \geq \Phi_g^S(\boldsymbol{x})$.

We next show $\Phi_g^S(\boldsymbol{x}) \geq W^S(\boldsymbol{x})$. We first expand and rewrite $W^S(\boldsymbol{x})$ as

$$W^S(\boldsymbol{x}) = \sum_a f_a(y_a) - c_a(x_a) + \sum_{S \subseteq N(a), s=|S|} \frac{1}{s\binom{d_a}{s}} \sum_{b \in S} \{f_a(\sum_{c \in S} q_{ca} x_c) - f_a(\sum_{c \in S, c \neq b} q_{ca} x_c)\}. \tag{12}$$

Since $f_a(x)$ is concave and increasing

$$\sum_{b \in S} \{f_a(\sum_{c \in S} q_{ca} x_c) - f_a(\sum_{c \in S, c \neq b} q_{ca} x_c)\} \leq f_a(\sum_{c \in S} q_{ca} x_c).$$

This is sufficient to show that

$$f_a(y_a) \leq \sum_{S \subseteq N(a), s=|S|} \frac{1}{s\binom{d_a}{s}} f_a(\sum_{c \in S} q_{ca} x_c). \tag{13}$$

One can expand the right hand side (RHS) summation in (13) and rewrite it using the size of subsets as the summation variable as follows

$$RHS = \sum_{s=1}^{d_a} \sum_{S \subseteq N(a), |S|=s} \frac{1}{s\binom{d_a}{s}} f_a(\sum_{c \in S} q_{ca} x_c).$$

We prove (13) using a simple procedure.

1. Among all subsets with original size s, choose the set S^* with the smallest value of $f_a(\sum_{c \in S} q_{ca} x_c)$,
2. Choose an arbitrary element x_m from S^*,
3. Remove x_m from S^*,
4. Choose an arbitrary subset S' such that $x_m \notin S'$,
5. Add x_m to S',
6. Repeat the procedure until all sets have the same value of $f_a(\sum_{c \in S} q_{ca} x_c)$.

The total sum in RHS decreases throughout the procedure so

$$\frac{1}{d_a} f_a(y_a) \leq \sum_{S \subseteq N(a), |S|=s} \frac{1}{s\binom{d_a}{s}} f_a(\sum_{c \in S} q_{ca} x_c). \tag{14}$$

All remaining subsets evaluate to $f_a(y_a)$ and there are $\binom{d_a-1}{s-1}$ such subsets after all iterations are over. Summing (14) over all players a shows that (13) holds and $\Phi_g^S(\boldsymbol{x}) \geq W^S(\boldsymbol{x})$. □

We showed that the social welfare induced by central control for incremental and Shapley utility models is no larger than the social welfare under strategic contribution when users have twice as large consumption utilities. Although our result does not attribute inefficiencies from user strategic behavior to lack of attention or excessive information, it indicates that improvements in user experience such as spam reduction, easy exploration and improved information discovery compensate for either of the existing inefficiencies from strategic behavior in online environments.

4.2 Simple Games with Unbounded Price of Anarchy

Our bicriteria bound does not exactly quantify the price of anarchy. Although the social gain from central control can be compensated by improvements in user experience, the degradation from strategic behavior can be still unbounded. We introduce a family of general production games that have an arbitrarily large price of anarchy. We consider agents with linear consumption utilities and convex polynomial cost functions. It is worthwhile to note that the incremental, proportional and, Shapley utility functions are equal.

We investigate a family of production games parameterized by γ and a set $\{\alpha_a\}$ of marginal consumption utilities. Agents have linear consumption utility $f_a(y) = \alpha_a y, \alpha_a > 0$. Moreover, they have polynomial cost functions $c_a(x) = \frac{1}{\gamma}x^\gamma, \gamma > 1$. Regardless of network structure the optimal and equilibrium strategy vectors can be characterized via FOC, so we can exactly quantify the price of anarchy.

Theorem 4.4. *Regardless of the network structure, the price of anarchy for the family of production games defined by cost function $c_a(x) = \frac{1}{\gamma}x^\gamma$ and utility functions $f_a(y) = \alpha_a y$, is equal to $(\frac{2\gamma-1}{\gamma-1})2^{\frac{\gamma}{1-\gamma}}$ when $\gamma > 1$ and $\alpha_a > 0$.*

Proof. The utility function for every player a is $u_a(\boldsymbol{x}) = \alpha_a y_a - c_a(x_a) + \beta_a x_a$ where $\beta_a = \sum_{b\sim a} q_{ba}\alpha_b$ and the social welfare function is $W(\boldsymbol{x}) = \sum_a \{2\beta_a x_a - c_a(x_a)\}$. It is easy to observe that the social welfare and agent utility functions are strictly concave so the first-order optimality conditions characterize the pure Nash equilibrium strategy, $\boldsymbol{x}$, and the optimum strategy, $\boldsymbol{x^*}$, as

$$x_a = \beta_a^{\frac{1}{\gamma-1}},$$
$$x_a^* = (2\beta_a)^{\frac{1}{\gamma-1}}.$$

Therefore the social welfare at pure Nash equilibrium $W(\boldsymbol{x}) = \frac{2\gamma-1}{\gamma}\sum_a \beta_a^{\frac{\gamma}{\gamma-1}}$ and the optimal social welfare $W(\boldsymbol{x^*}) = \frac{\gamma-1}{\gamma}2^{\frac{\gamma}{\gamma-1}}\sum_a \beta_a^{\frac{\gamma}{\gamma-1}}$. The pure-Nash price of anarchy is defined as the ratio of the pure Nash equilibrium social welfare divided by the optimum social welfare and is equal to $(\frac{2\gamma-1}{\gamma-1})2^{\frac{\gamma}{1-\gamma}}$. □

Exact analysis of the price of anarchy provides us with more predictive power over the existing inefficiencies due to strategic behavior. We now distinguish games from the introduced family with an arbitrarily large and an arbitrarily close-to-one price of anarchy.

Corollary 4.5. *Regardless of network structure, the price of anarchy for the family of production games defined by cost function $c_a(x) = \frac{1}{\gamma}x^\gamma$ and utility functions $f_a(y) = \alpha_a y, \alpha_a > 0$ can be arbitrarily large when γ gets arbitrarily close to one and the price of anarchy can be arbitrarily close to one when γ gets arbitrarily large.*

Corollary 4.5 identifies utility games with almost linear cost functions and linear consumption utilities as instances with unbounded price of anarchy. It also identifies utility games, that have polynomial cost with large coefficient and linear consumption utilities, as instances with almost no inefficiencies from strategic behavior. Non-existence of inefficiencies from strategic behavior in the latter example is mainly due to infinitesimal production at the pure Nash equilibrium and the social optimum for all users.

5 Robust Analysis of the Price of Anarchy

Smooth analysis of games with sum objectives identifies a sufficient condition for an upper bound on the price of anarchy of pure Nash equilibria and encodes a canonical proof template for deriving such bounds [12]. Canonical bounds extend automatically to more general notions of equilibria such as mixed Nash equilibrium, correlated equilibrium and no-regret sequences. A utility maximization game is (λ, μ)-smooth if for every two outcomes $\boldsymbol{x}$ and $\boldsymbol{x}^*$,

$$\sum_a u_a(x_a^*, \boldsymbol{x}_{-a}) \geq \lambda W(\boldsymbol{x}^*) - \mu W(\boldsymbol{x}). \tag{15}$$

Roughgarden [12] defines robust price of anarchy as the best lower bound on the price of anarchy that is provable using a smoothness argument. The robust price of anarchy for a utility maximization game is

$$\sup \left\{ \frac{\lambda}{1+\mu} \ : \ (\lambda, \mu) \text{ s.t. game is } (\lambda, \mu)\text{-smooth} \right\}. \tag{16}$$

Congestion games with cost functions restricted to a fixed set are proved to be tight; meaning that the canonical price of anarchy is also robust. In particular network routing games are (λ, μ)-smooth with robust price of anarchy of $\frac{4}{3}$ for non-atomic flows and a price of anarchy of $\frac{5}{2}$ for atomic flows. We focus on production games with linear consumption utility and quadratic cost functions($\gamma = 2$). Theorem 4.4 quantifies price of anarchy of $\frac{3}{4}$ for this class of games. On the other hand, we show that the robust price of anarchy for the same class of games is at most 0.098, meaning that the robust price of anarchy is not tight which is a strong distinction between production games and network congestion games.

Theorem 5.1. *Robust price of anarchy for a production game with consumption utilities $f_a(y) = \alpha_a y, \alpha > 0$ and production cost $c_a(x) = 0.5x^2$ is at most 0.098.*

Proof. Agent utility for the proposed family of games is

$$u_a(\boldsymbol{x}) = \alpha_a y_a - \frac{x_a^2}{2} + \beta_a x_a, \tag{17}$$

where $\beta_a = \sum_{b \sim a} q_{ba} \alpha_b$. Also the social welfare function can be summarized into

$$W(\boldsymbol{x}) = \sum_a \left\{ 2\beta_a x_a - \frac{x_a^2}{2} \right\}. \tag{18}$$

Consider two strategy vectors $\boldsymbol{x}$ and $\boldsymbol{x}^*$ where for all users a, $\boldsymbol{x_a} = d\beta_a$ and $\boldsymbol{x_a^*} = c\beta_a$ for constants c and d ($c \neq d$). Smoothness conditions in (15) can be written for $\boldsymbol{x}$ and $\boldsymbol{x}^*$ as

$$\sum_a (d + c - \frac{c^2}{2})\beta_a^2 \geq \sum_a (2\lambda c - \lambda\frac{c^2}{2} - 2\mu d + \mu\frac{d^2}{2})\beta_a^2.$$

So (λ, μ)-smoothness requires

$$d + c - \frac{c^2}{2} \geq 2\lambda c - \lambda\frac{c^2}{2} - 2\mu d + \mu\frac{d^2}{2}. \tag{19}$$

Robust price of anarchy is defined as the supremum of $\frac{\lambda}{1+\mu}$ over all pairs of (λ, μ) for which the smoothness conditions hold. Equation (19) requires $\mu \geq \frac{2\lambda c - \lambda\frac{c^2}{2} - d - c + \frac{c^2}{2}}{2d - \frac{d^2}{2}}$ where $2d - \frac{d^2}{2} > 0$. Supremum of $\frac{\lambda}{1+\mu}$ takes place at the smallest value of μ, so we can set

$$\mu = \frac{2\lambda c - \lambda\frac{c^2}{2} - d - c + \frac{c^2}{2}}{2d - \frac{d^2}{2}}. \tag{20}$$

After substituting μ from (20), the canonical price of anarchy bound from (15) will be equal to

$$\frac{(2d - \frac{d^2}{2})\lambda}{(2c - \frac{c^2}{2})\lambda + 2d - \frac{d^2}{2} - d - c - \frac{c^2}{2}}. \tag{21}$$

Equation (21) is a hyperbolic function in λ and its supremum is equal to

$$\frac{4d - d^2}{4c - c^2} \tag{22}$$

when $2c - \frac{c^2}{2} > 0$ and $2d - \frac{d^2}{2} - d - c - \frac{c^2}{2} > 0$. Our choice of values for c and d that satisfy above inequalities determines an upper bound on the robust price of anarchy. Values $c = 2.2$ and $d = 0.1$ generate an upper bound of 0.098 for robust price of anarchy using (22) and satisfy above inequalities with valid values for λ and μ. □

Theorem 5.1 shows that the robust price of anarchy is not equal to the pure-Nash price of anarchy so any bound from the canonical analysis of the game is not tight. This is in contrast with Roughgarden's result about network routing games where canonical bounds are tight even for the smallest class of equilibria, i.e. pure Nash equilibria [12]. Theorem 5.1 does not prove smoothness of the production game but in a sense it shows that canonical analysis and robust price of anarchy are not the right tool for analyzing proposed production games. Although our bicriteria bound is very similar to that of Tardos et al. in [13], but theorem 5.1 proves a strong distinction between routing games and the games studies in this paper.

6 Discussion

We introduced a game-theoretic framework to analyze inefficiencies from strategic behavior in online social networks. We proved that the degradation in efficiency of the proposed game resulting from strategic user participation can be compensated by improvements in the online environment.

Although we are unable to give a closed-form solution for the pure-equilibrium strategy in the general production game, we can characterize a closed-form solution in d-regular graphs. We are also able to find the pure-strategy Nash equilibrium numerically in general networks. Equilibrium utility is higher for larger values of d in a d-regular graph. Our numerical analysis on several random networks shows that the equilibrium utility, the utility from attention, and the utility from information are significantly correlated with the number of friends a user has in the social network. There are a number of interesting directions for future work within the proposed framework; we conclude this paper by explaining some of these directions.

A natural direction to explore is to study the proposed general production game as a network formation game where strategic players can add and drop links in the social network. Because the proposed general production game does not include any cost for information overload, dense network structures are more likely to form. Empirical evidence indicates that users appreciate new information less as they consume more information. We implicitly incorporate cost of information overload by modeling user happiness as a concave function. An interesting direction is to model an explicit cost for information overload. Our preliminary results show that unfortunately our bicriteria bound on the price of anarchy does not hold in this setting.

Different mechanisms of viewing information on user's news feed in online social networks can be modeled as a different attention sharing mechanism within our general production framework. An interesting future direction is to compare available attention sharing schemes e.g. chronological sorting, collaborative filtering, and etc in the proposed game-theoretic framework. This is very similar to the mechanism-design approach that Ghosh et al have taken [4] [3].

Acknowledgements. We are thankful to Amin Saberi and anonymous reviewers for helpful comments.

References

1. Borgs, C., Chayes, J., Karrer, B., Meeder, B., Ravi, R., Reagans, R., Sayedi, A.: Game-Theoretic Models of Information Overload in Social Networks. In: Kumar, R., Sivakumar, D. (eds.) WAW 2010. LNCS, vol. 6516, pp. 146–161. Springer, Heidelberg (2010)
2. Franck, G.: Essays on Science and Society: Scientific Communication–A Vanity Fair? Science 286(5437), 53–55 (1999)
3. Ghosh, A., Hummel, P.: A game-theoretic analysis of rank-order mechanisms for user-generated content. In: Proceedings of the 12th ACM Conference on Electronic Commerce, EC 2011, pp. 189–198. ACM, New York (2011)

4. Ghosh, A., McAfee, P.: Incentivizing high-quality user-generated content. In: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, pp. 137–146. ACM, New York (2011)
5. Huberman, B.A., Loch, C.H., ÖNçüler, A.: Status As a Valued Resource. Social Psychology Quarterly 67(1), 103–114 (2004)
6. Huberman, B.A., Romero, D.M., Wu, F.: Crowdsourcing, attention and productivity. J. Inf. Sci. 35, 758–765 (2009)
7. Jain, S., Chen, Y., Parkes, D.C.: Designing incentives for online question and answer forums. In: Proceedings of the 10th ACM Conference on Electronic Commerce, EC 2009, pp. 129–138. ACM, New York (2009)
8. Leskovec, J., Adamic, L., Huberman, B.: The dynamics of viral marketing. In: Proceedings of the 7th ACM Conference on Electronic Commerce, EC 2006, pp. 228–237. ACM Press (2005)
9. Miura, A., Yamashita, K.: Psychological and Social Influences on Blog Writing: An Online Survey of Blog Authors in Japan. Journal of Computer-Mediated Communication 12(4), 1452–1471 (2007)
10. Rosen, J.B.: Existence and Uniqueness of Equilibrium Points for Concave N-Person Games. Econometrica 33(3), 520–534 (1965)
11. Rosenthal, R.W.: A class of games possessing pure-strategy Nash equilibria. International Journal of Game Theory 2(1), 65–67 (1973)
12. Roughgarden, T.: Intrinsic robustness of the price of anarchy. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 513–522. ACM, New York (2009)
13. Roughgarden, T., Tardos, E.: How bad is selfish routing? In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, pp. 93–102. IEEE Computer Society, Washington, DC (2000)
14. Shapley, L.S.: A value for n-person games. In: Kuhn, H.W., Tucker, A.W. (eds.) Contributions to the Theory of Games, Volume II. Annals of Mathematics Studies, vol. 28, pp. 307–317. Princeton University Press, Princeton (1953)
15. Wu, F., Huberman, B.A.: Novelty and collective attention. Technical report, Proceedings of National Academy of Sciences (2007)
16. Wu, F., Wilkinson, D.M., Huberman, B.A.: Feedback loops of attention in peer production. In: Proceedings of the 2009 International Conference on Computational Science and Engineering, vol. 4, pp. 409–415. IEEE Computer Society, Washington, DC (2009)

On Certain Properties of Random Apollonian Networks

Alan Frieze and Charalampos E. Tsourakakis

Department of Mathematical Sciences, Carnegie Mellon University, USA
af1p@random.math.cmu.edu, ctsourak@math.cmu.edu

Abstract. In this work we analyze fundamental properties of Random Apollonian Networks [34, 35], a popular random graph model which generates planar graphs with power law properties. Specifically, we analyze (a) the degree distribution, (b) the k largest degrees, (c) the k largest eigenvalues and (d) the diameter, where k is a constant.

1 Introduction

Due to the surge of interest in social networks, the Web graph, the Internet, biological networks and many other types of networks, a large amount of research has focused on modeling real-world networks in recent years. Existing well-known models include the preferential attachment model [7], Kronecker graphs [28], the Cooper-Frieze model [16], the Aiello-Chung-Lu model [1], protean graphs [31] and the Fabrikant-Koutsoupias-Papadimitriou model [21]. In this work we focus on Random Apollonian Networks (RANs), a popular random graph model for generating planar graphs with power law properties [35]. Before we state our main results we briefly describe the model.

Model: An example of a RAN is shown in Figure 1. At time $t = 1$ the RAN is shown in Figure 1a. At each step $t \geq 2$ a face F is chosen uniformly at random among the faces of G_t. Let i, j, k be the vertices of F. We add a new vertex inside F and we connect it to i, j, k. Higher dimensional RANs also exist where instead of triangles we have k-simplexes $k \geq 3$, see [34]. It is easy to see that the number of vertices n_t, edges m_t and faces F_t at time $t \geq 1$ in a RAN G_t satisfy:

$$n_t = t + 3, \;\; m_t = 3t + 3, \;\; F_t = 2t + 1.$$

Note that a RAN is a maximal planar graph since for any planar graph $m_t \leq 3n_t - 6 \leq 3t + 3$.

Surprisingly, despite the popularity of the model various important properties have been analyzed experimentally and heuristically with lack of rigor. In this work, we prove the following theorems using existing techniques [3, 22, 29].

Theorem 1 (Degree Sequence)
Let $Z_k(t)$ denote the number of vertices of degree k at time t, $k \geq 3$. For t sufficiently large and for any $k \geq 3$ there exists a constant b_k depending on k such that

$$|\mathbb{E}\left[Z_k(t)\right] - b_k t| \leq K, \;\; \textit{where} \;\; K = 3.6.$$

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 93–112, 2012.

Furthermore, for any $\lambda > 0$

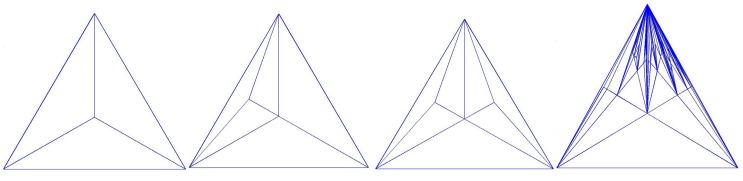
$$\mathbf{Pr}\left[|Z_k(t) - \mathbb{E}\left[Z_k(t)\right]| \geq \lambda\right] \leq e^{-\frac{\lambda^2}{72t}}. \tag{1}$$

For previous weaker results on the degree sequence see [33,35]. An immediate corollary which proves strong concentration of $Z_k(t)$ around its expectation is obtained from Theorem 1 and a union bound by setting $\lambda = 10\sqrt{t \log t}$. Specifically:

Corollary 1. *For all possible degrees* k

$$\mathbf{Pr}\left[|Z_k(t) - \mathbb{E}\left[Z_k(t)\right]| \geq 10\sqrt{t \log t}\right] = o(1).$$

The next theorem provides insight into the asymptotic growth of the highest degrees of RANs and is crucial in proving Theorem 3.



(a) $t = 1$ (b) $t = 2$ (c) $t = 3$ (d) $t = 100$

Fig. 1. Snapshots of a Random Apollonian Network (RAN) at: (a) $t = 1$ (b) $t = 2$ (c) $t = 3$ (d) $t = 100$

Theorem 2 (Highest Degrees). *Let* $\Delta_1 \geq \Delta_2 \geq \ldots \geq \Delta_k$ *be the* k *highest degrees of the RAN* G_t *at time* t *where* k *is a fixed positive integer. Also, let* $f(t)$ *be a function such that* $f(t) \to +\infty$ *as* $t \to +\infty$. *Then whp*[1]

$$\frac{t^{1/2}}{f(t)} \leq \Delta_1 \leq t^{1/2} f(t)$$

and for $i = 2, \ldots, k$

$$\frac{t^{1/2}}{f(t)} \leq \Delta_i \leq \Delta_{i-1} - \frac{t^{1/2}}{f(t)}.$$

The growing function $f(t)$ cannot be removed, see [22]. Using Theorem 2 and the technique of Mihail and Papadimitriou [29] we show how the top eigenvalues of the adjacency matrix representation of a RAN grow asymptotically as $t \to +\infty$ *whp*.

Theorem 3 (Largest Eigenvalues). *Let* k *be a fixed positive integer. Also, let* $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k$ *be the largest* k *eigenvalues of the adjacency matrix of* G_t. *Then whp* $\lambda_i = (1 \pm o(1))\sqrt{\Delta_i}$.

[1] An event A_t holds with high probability (*whp*) if $\lim_{t \to +\infty} \mathbf{Pr}\left[A_t\right] = 1$.

Also, we show the following refined upper bound for the asymptotic growth of the diameter.

Theorem 4 (Diameter). *The diameter $d(G_t)$ of G_t satisfies in probability $d(G_t) \leq \rho \log t$ where $\frac{1}{\rho} = \eta$ is the unique solution greater than 1 of the equation $\eta - 1 - \log \eta = \log 3$.*

The outline of the paper is as follows: in Section 2 we present briefly related work and technical preliminaries needed for our analysis. We prove Theorems 1, 2, 3 and 4 in Sections 3, 4, 5 and 6 respectively. Unavoidably due to the space constraint we have included for completeness reasons the proofs of certain lemmas which are omitted from the main part of our paper in the Appendix 8. In Section 7 we investigate another property of the model. Finally, in Section 8 we conclude by suggesting few open problems.

2 Related Work

Apollonius of Perga was a Greek geometer and astronomer noted for his writings on conic sections. He introduced the problem of space filling packing of spheres whose classical solution, the so-called Apollonian packing [25], exhibits a power law behavior. Specifically, the circle size distribution follows a power law with exponent around 1.3 [11]. Apollonian Networks (ANs) were introduced in [4] and independently in [20]. Zhou et al. [35] introduced Random Apollonian Networks (RANs). Their degree sequence was analyzed inaccurately in [35] (see comment in [33]) and subsequently using physicist's methodology in [33]. Eigenvalues of RANs have been studied only experimentally [5]. Concerning the diameter of RNAs it has been shown to grow logarithmically [35] using heuristic arguments (see for instance equation B6, Appendix B in [35]). RANs are planar 3-trees, a special case of random k-trees [27]. Cooper and Uehara [17] and Gao [23] analyzed the degree distribution of random k-trees, a closely related model to RANs. In RANs –in contrast to random k-trees– the random k clique chosen at each step has never previously been selected. For example, in the two dimensional RAN any chosen face is being subdivided into three new faces by connecting the incoming vertex to the vertices of the boundary. Random k-trees due to their power law properties have been proposed as a model for complex networks, see, e.g., [17,24] and references therein. Recently, a variant of k-trees, namely ordered increasing k-trees has been proposed and analyzed in [30]. Closely related to RANs but not the same are random Apollonian network structures which have been analyzed by Darrasse, Soria et al. [8,18,19].

Bollobás, Riordan, Spencer and Tusnády [10] proved rigorously the power law distribution of the Barabási-Albert model [7]. Chung, Lu, Vu [15] Flaxman, Frieze, Fenner [22] and Mihail, Papadimitriou [29] have proved rigorous results for eigenvalue related properties of real-world graphs using various random graph models. In Section 3 we invoke the following useful lemma.

Lemma 1 (Lemma 3.1, [14]). *Suppose that a sequence $\{a_t\}$ satisfies the recurrence*

$$a_{t+1} = (1 - \frac{b_t}{t + t_1})a_t + c_t$$

for $t \geq t_0$. Furthermore suppose $\lim_{t\to+\infty} b_t = b > 0$ and $\lim_{t\to+\infty} c_t = c$. Then $\lim_{t\to+\infty} \frac{a_t}{t}$ exists and

$$\lim_{t\to+\infty} \frac{a_t}{t} = \frac{c}{1+b}.$$

In Section 3 we also use the Azuma-Hoeffding inequality [6,26].

Lemma 2 (Azuma-Hoeffding inequality). *Let $\lambda > 0$. Also, let $(X_t)_{t=0}^n$ be a martingale with $|X_{t+1} - X_t| \leq c$ for $t = 0, \ldots, n-1$. Then:*

$$\mathbf{Pr}\left[|X_n - X_0| \geq \lambda\right] \leq \exp\Big(-\frac{\lambda^2}{2c^2 n}\Big).$$

3 Proof of Theorem 1

We decompose our proof in a sequence of Lemmas. For brevity let $N_k(t) = \mathbb{E}\left[Z_k(t)\right]$, $k \geq 3$. Also, let $d_v(t)$ be the degree of vertex v at time t and $\mathbf{1}(d_v(t) = k)$ be an indicator variable which equals 1 if $d_v(t) = k$, otherwise 0. Then, for any $k \geq 3$ we can express the expected number $N_k(t)$ of vertices of degree k as a sum of expectations of indicator variables:

$$N_k(t) = \sum_v \mathbb{E}\left[\mathbf{1}(d_v(t) = k)\right]. \tag{2}$$

We distinguish two cases in the following.

• CASE 1: $k = 3$:

Observe that a vertex of degree 3 is created only by an insertion of a new vertex. The expectation $N_3(t)$ satisfies the following recurrence[2]

$$N_3(t+1) = N_3(t) + 1 - \frac{3N_3(t)}{2t+1}. \tag{3}$$

The basis for Recurrence (3) is $N_3(1) = 4$. We prove the following lemma which shows that $\lim_{t\to+\infty} \frac{N_3(t)}{t} = \frac{2}{5}$.

Lemma 3. *$N_3(t)$ satisfies the following inequality:*

$$|N_3(t) - \frac{2}{5}t| \leq K, \quad \textit{where} \quad K = 3.6 \tag{4}$$

[2] The three initial vertices participate in one less face than their degree. However, this leaves our results unchanged.

Proof. We use induction. Assume that $N_3(t) = \frac{2}{5}t + e_3(t)$, where $e_3(t)$ stands for the error term. We wish to prove that for all t, $|e_3(t)| \leq K$. The result trivially holds for $t = 1$. We also see that for $t = 1$ inequality (4) is tight. Assume the result holds for some t. We show it holds for $t + 1$.

$$N_3(t+1) = N_3(t) + 1 - \frac{3N_3(t)}{2t+1} \Rightarrow$$
$$e_3(t+1) = e_3(t) + \frac{3}{5} - \frac{6t + 15e_3(t)}{10t+5} = e_3(t)\Big(1 - \frac{3}{2t+1}\Big) + \frac{3}{5(2t+1)} \Rightarrow$$
$$|e_3(t+1)| \leq K(1 - \frac{3}{2t+1}) + \frac{3}{5(2t+1)} \leq K$$

Therefore inductively Inequality (4) holds for all $t \geq 1$. □

• CASE 2: $k \geq 4$:

For $k \geq 4$ the following holds:

$$\mathbb{E}\left[\mathbf{1}(d_v(t+1) = k)\right] = \mathbb{E}\left[\mathbf{1}(d_v(t) = k)\right](1 - \frac{k}{2t+1}) + \mathbb{E}\left[\mathbf{1}(d_v(t) = k-1)\right]\frac{k-1}{2t+1} \quad (5)$$

Therefore, we can rewrite Equation (2) for $k \geq 4$ as follows:

$$N_k(t+1) = N_k(t)(1 - \frac{k}{2t+1}) + N_{k-1}(t)\frac{k-1}{2t+1} \quad (6)$$

Lemma 4. *For any $k \geq 3$, the limit $\lim_{t\to+\infty} \frac{N_k(t)}{t}$ exists. Specifically, let $b_k = \lim_{t\to+\infty} \frac{N_k(t)}{t}$. Then, $b_3 = \frac{2}{5}, b_4 = \frac{1}{5}, b_5 = \frac{4}{35}$ and for $k \geq 6$ $b_k = \frac{24}{k(k+1)(k+2)}$. Furthermore, for all $k \geq 3$*

$$|N_k(t) - b_k t| \leq K, \quad \textit{where} \quad K = 3.6. \quad (7)$$

Proof. For $k = 3$ the result holds by Lemma 3 and specifically $b_3 = \frac{2}{5}$. Assume the result holds for some k. We show that it holds for $k + 1$ too. Rewrite Recursion (6) as: $N_k(t+1) = (1 - \frac{b_t}{t+t_1})N_k(t) + c_t$ where $b_t = k/2$, $t_1 = 1/2$, $c_t = N_{k-1}(t)\frac{k-1}{2t+1}$. Clearly $\lim_{t\to+\infty} b_t = k/2 > 0$ and $\lim_{t\to+\infty} c_t = \lim_{t\to+\infty} b_{k-1}t\frac{k-1}{2t+1} = b_{k-1}(k-1)/2$. Hence by Lemma 1:

$$\lim_{t\to+\infty} \frac{N_k(t)}{t} = \frac{(k-1)b_{k-1}/2}{1 + k/2} = b_{k-1}\frac{k-1}{k+2}.$$

Since $b_3 = \frac{2}{5}$ we obtain that $b_4 = \frac{1}{5}$, $b_5 = \frac{4}{35}$ for any $k \geq 6$, $b_k = \frac{24}{k(k+1)(k+2)}$. This shows that the degree sequence of RANs follows a power law distribution with exponent 3.

Now we prove Inequality (7). The case $k = 3$ was proved in Lemma 3. Let $e_k(t) = N_k(t) - b_k t$. Assume the result holds for some $k \geq 3$, i.e., $|e_k(t)| \leq K$ where $K = 3.6$. We show it holds for $k+1$ too. Substituting in Recurrence (2) and using the fact that $b_{k-1}(k-1) = b_k(k+2)$ we obtain the following:

$$e_k(t+1) = e_k(t) + \frac{k-1}{2t+1} e_{k-1}(t) - \frac{k}{2t+1} e_k(t) \Rightarrow$$
$$|e_k(t+1)| \leq |(1 - \frac{k}{2t+1}) e_k(t)| + |\frac{k-1}{2t+1} e_{k-1}(t)| \leq K(1 - \frac{1}{2t+1}) \leq K$$

Hence by induction, Inequality (7) holds for all $k \geq 3$. □

Using integration and a first moment argument, it can be seen that Lemma 4 agrees with Theorem 2 where it is shown that the maximum degree is $\approx t^{1/2}$. (While $b_k = O(k^{-3})$ suggests a maximum degree of order $t^{1/3}$, summing b_k over $k \geq K$ suggests a maximum degree of order $t^{1/2}$).

Finally, the next Lemma proves the concentration of $Z_k(t)$ around its expected value for $k \geq 3$. This lemma applies Lemma 2 and completes the proof of Theorem 1.

Lemma 5. *Let $\lambda > 0$. For $k \geq 3$*

$$\mathbf{Pr}\left[|Z_k(t) - \mathbb{E}\left[Z_k(t)\right]| \geq \lambda\right] \leq e^{-\frac{\lambda^2}{72t}}. \tag{8}$$

Proof. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be the probability space induced by the construction of a RAN after t insertions. Fix k, where $k \geq 3$, and let $(X_i)_{i \in \{0,1,\ldots,t\}}$ be the martingale sequence defined by $X_i = \mathbb{E}\left[Z_k(t)|\mathcal{F}_i\right]$, where $\mathcal{F}_0 = \{\emptyset, \Omega\}$ and $\mathcal{F}_i$ is the σ-algebra generated by the RAN process after i steps. Notice $X_0 = \mathbb{E}\left[Z_k(t)|\{\emptyset, \Omega\}\right] = N_k(t)$, $X_t = Z_k(t)$. We show that $|X_{i+1} - X_i| \leq 6$ for $i = 0, \ldots, t-1$. Let $P_j = (Y_1, \ldots, Y_{j-1}, Y_j)$, $P'_j = (Y_1, \ldots, Y_{j-1}, Y'_j)$ be two sequences of face choices differing only at time j. Also, let $\bar{P}, \bar{P}'$ continue from P_j, P'_j until t. We call the faces Y_j, Y'_j special with respect to $\bar{P}, \bar{P}'$. We define a measure preserving map $\bar{P} \mapsto \bar{P}'$ in the following way: for every choice of a non-special face in process $\bar{P}$ at time l we make the same face choice in $\bar{P}'$ at time l. For every choice of a face inside the special face Y_j in process $\bar{P}$ we make an isomorphic (w.r.t., e.g., clockwise order and depth) choice of a face inside the special face Y'_j in process $\bar{P}'$. Since the number of vertices of degree k can change by at most 6, i.e., the (at most) 6 vertices involved in the two faces Y_j, Y'_j the following holds:

$$|\mathbb{E}\left[Z_k(t)|P\right] - \mathbb{E}\left[Z_k(t)|P'\right]| \leq 6.$$

Furthermore, this holds for any P_j, P'_j. We deduce that X_{i-1} is a weighted mean of values, whose pairwise differences are all at most 6. Thus, the distance of the mean X_{i-1} is at most 6 from each of these values. Hence, for any one step refinement $|X_{i+1} - X_i| \leq 6 \; \forall i \in \{0, \ldots, t-1\}$. By applying the Azuma-Hoeffding inequality as stated in Lemma 2 we obtain

$$\mathbf{Pr}\left[|Z_k(t) - \mathbb{E}\left[Z_k(t)\right]| \geq \lambda\right] \leq 2e^{-\frac{\lambda^2}{72t}}. \tag{9}$$

□

4 Proof of Theorem 2

We decompose the proof of Theorem 2 into several lemmas which we prove in the following. Specifically, the proof follows directly from Lemmas 7, 8, 9, 10, 11. We partition the vertices into three sets: those added before t_0, between t_0 and t_1 and after t_1 where $t_0 = \log\log\log(f(t))$ and $t_1 = \log\log(f(t))$. Recall that $f(t)$ is a function such that $\lim_{t\to+\infty} f(t) = +\infty$. We define a supernode to be a collection of vertices and the degree of the supernode the sum of the degrees of its vertices.

Lemma 6. *Let $d_t(s)$ denote the degree of vertex s at time t. and let $a^{(k)} = a(a+1)\dots(a+k-1)$ denote the rising factorial function. Then, for any positive integer k*

$$\mathbb{E}\left[d_t(s)^{(k)}\right] \leq \frac{(k+2)!}{2}\left(\frac{2t}{s}\right)^{\frac{k}{2}}. \tag{10}$$

Proof. See Appendix. □

Lemma 7. *The degree X_t of the supernode V_{t_0} of vertices added before time t_0 is at least $t_0^{1/4}\sqrt{t}$ whp.*

Proof. We consider a modified process $\mathcal{Y}$ coupled with the RAN process, see also Figure 2. Specifically, let Y_t be the modified degree of the supernode in the modified process $\mathcal{Y}$ which is defined as follows: for any type of insertion in the original RAN process –note there exist three types of insertions with respect to how the degree X_t of the supernode (black circle) gets affected, see also Figure 2– Y_t increases by 1. We also define $X_{t_0} = Y_{t_0}$. Note that $X_t \geq Y_t$ for all $t \geq t_0$. Let $d_0 = X_{t_0} = Y_{t_0} = 6t_0 + 6$ and $p^* = \mathbf{Pr}\left[Y_t = d_0 + r | Y_{t_0} = d_0\right]$.
The following technical claim is proved in an appendix.

Claim (1)

$$p^* \leq \binom{d_0+r-1}{d_0-1}\left(\frac{2t_0+3}{2t+1}\right)^{d_0/2} e^{\frac{3}{2}+t_0-\frac{d_0}{2}+\frac{2r}{3\sqrt{t}}}$$

Let $\mathcal{A}_1$ denote the event that the supernode consisting of the first t_0 vertices has degree Y_t in the modified process $\mathcal{Y}$ less than $t_0^{1/4}\sqrt{t}$. Note that since $\{X_t \leq t_0^{1/4}\sqrt{t}\} \subseteq \{Y_t \leq t_0^{1/4}\sqrt{t}\}$ it suffices to prove that $\mathbf{Pr}\left[Y_t \leq t_0^{1/4}\sqrt{t}\right] = o(1)$. Using Claim (1) we obtain

$$\begin{aligned}
\mathbf{Pr}\left[\mathcal{A}_1\right] &\leq \sum_{r=0}^{t_0^{1/4}\sqrt{t}-(6t_0+6)} \binom{r+6t_0+5}{6t_0+5}\left(\frac{2t_0+3}{2t+1}\right)^{3t_0+3} e^{-\frac{3}{2}-2t_0+\frac{2t_0^{1/4}}{3}} \\
&\leq t_0^{1/4}t^{1/2}\frac{\left(t_0^{1/4}t^{1/2}\right)^{6t_0+5}}{(6t_0+5)!}\left(\frac{2t_0+3}{2t+1}\right)^{3t_0+3} e^{-\frac{3}{2}-2t_0+\frac{2t_0^{1/4}}{3}} \\
&\leq \left(\frac{t}{2t+1}\right)^{3t_0+3}\frac{t_0^{3t_0/2+3/2}(2t_0+3)^{3t_0+3}}{(6t_0+5)^{6t_0+5}} e^{4t_0+7/2+2/3t_0^{1/4}} \\
&\leq 2^{-(3t_0+3)}\frac{e^{4t_0+7/2+2/3t_0^{1/4}}}{(6t_0+5)^{\frac{3}{2}t_0+\frac{1}{2}}} = o(1).
\end{aligned}$$

□

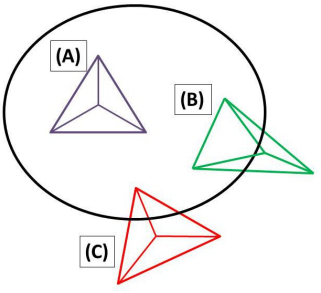


Fig. 2. Coupling used in Lemma 7

Lemma 8. *No vertex added after t_1 has degree exceeding $t_0^{-2}t^{1/2}$ whp.*

Proof. Let $\mathcal{A}_2$ denote the event that some vertex added after t_1 has degree exceeding $t_0^{-2}t^{1/2}$. We use a union bound, a third moment argument and Lemma 6 to prove that $\mathbf{Pr}[\mathcal{A}_2] = o(1)$. Specifically

$$\mathbf{Pr}[\mathcal{A}_2] \leq \sum_{s=t_1}^{t} \mathbf{Pr}\left[d_t(s) \geq t_0^{-2}t^{1/2}\right] = \sum_{s=t_1}^{t} \mathbf{Pr}\left[d_t(s)^{(3)} \geq (t_0^{-2}t^{1/2})^{(3)}\right]$$
$$\leq t_0^6 t^{-3/2} \sum_{s=t_1}^{t} \mathbb{E}\left[d_t(s)^{(3)}\right] \leq 5!\sqrt{2}t_0^6 \sum_{s=t_1}^{t} s^{-3/2} \leq 5!2\sqrt{2}t_0^6 t_1^{-1/2} = o(1).$$

□

Lemma 9. *No vertex added before t_1 has degree exceeding $t_0^{1/6}t^{1/2}$ whp.*

Proof. Let $\mathcal{A}_3$ denote the event that some vertex added before t_1 has degree exceeding $t_0^{1/6}t^{1/2}$. We use again a third moment argument and Lemma 6 to prove that $\mathbf{Pr}[\mathcal{A}_3] = o(1)$.

$$\mathbf{Pr}[\mathcal{A}_3] \leq \sum_{s=1}^{t_1} \mathbf{Pr}\left[d_t(s) \geq t_0^{1/6}t^{1/2}\right] = \sum_{s=1}^{t_1} \mathbf{Pr}\left[d_t(s)^{(3)} \geq (t_0^{1/6}t^{1/2})^{(3)}\right]$$
$$\leq t_0^{-1/2}t^{-3/2} \sum_{s=1}^{t_1} \mathbb{E}\left[d_t(s)^{(3)}\right] \leq t_0^{-1/2}t^{-3/2} \sum_{s=1}^{t_1} 5!\sqrt{2}\frac{t^{3/2}}{s^{3/2}}$$
$$\leq 5!\sqrt{2}\zeta(3/2)t_0^{-1/2} = o(1)$$

where $\zeta(3/2) = \sum_{s=1}^{+\infty} s^{-3/2} \approx 2.612$. □

Lemma 10. *The k highest degrees are added before t_1 and have degree Δ_i bounded by $t_0^{-1}t^{1/2} \leq \Delta_i \leq t_0^{1/6}t^{1/2}$ whp.*

Proof. For the upper bound it suffices to show that $\Delta_1 \leq t_0^{1/6}t^{1/2}$. This follows immediately by Lemmas 8 and 9. The lower bound follows directly from Lemmas 7, 8 and 9. Assume that at most $k-1$ vertices added before t_1 have degree exceeding the lower bound $t_0^{-1}t^{1/2}$. Then the total degree of the supernode formed by the first t_0 vertices is $O(t_0^{1/6}\sqrt{t})$. This contradicts Lemma 7. Finally, since each vertex $s \geq t_1$ has degree at most $t_0^{-2}\sqrt{t} \ll t_0^{-1}t^{1/2}$ the k highest degree vertices are added before t_1 *whp*. □

The proof of Theorem 2 is completed with the following lemma whose proof is included in Appendix 8

Lemma 11. *The k highest degrees satisfy $\Delta_i \leq \Delta_{i-1} - \frac{\sqrt{t}}{f(t)}$ whp.*

5 Proof of Theorem 3

Having computed the highest degrees of a RAN in Section 4, eigenvalues are computed by adapting existing techniques [15, 22, 29]. We decompose the proof of Theorem 3 in Lemmas 12, 13, 14, 15. Specifically, in Lemmas 12, 13 we bound the degrees and co-degrees respectively. Having these bounds, we decompose the graph into a star forest and show in Lemmas 14 and 15 that its largest eigenvalues, which are $(1 \pm o(1))\sqrt{\Delta_i}$, dominate the eigenvalues of the remaining graph. This technique was pioneered by Mihail and Papadimitriou [29].

We partition the vertices into three set S_1, S_2, S_3. Specifically, let S_i be the set of vertices added after time t_{i-1} and at or before time t_i where

$$t_0 = 0, t_1 = t^{1/8}, t_2 = t^{9/16}, t_3 = t.$$

In the following we use the recursive variational characterization of eigenvalues [13]. Specifically, let A_G denote the adjacency matrix of a simple, undirected graph G and let $\lambda_i(G)$ denote the i-th largest eigenvalue of A_G. Then

$$\lambda_i(G) = \min_S \max_{x \in S, x \neq 0} \frac{x^T A_G x}{x^T x}$$

where S ranges over all $(n-i+1)$ dimensional subspaces of $\mathbb{R}^n$.

Lemma 12. *For any $\epsilon > 0$ and any $f(t)$ with $f(t) \to +\infty$ as $t \to +\infty$ the following holds whp: for all s with $f(t) \leq s \leq t$, for all vertices $r \leq s$, then $d_s(r) \leq s^{\frac{1}{2}+\epsilon}r^{-\frac{1}{2}}$.*

Proof. Set $q = \lceil \frac{4}{\epsilon} \rceil$. We use Lemma 6, a union bound and Markov's inequality to obtain:

$$\begin{aligned}
\mathbf{Pr}\left[\bigcup_{s=f(t)}^{t}\bigcup_{r=1}^{s}\{d_s(r)\geq s^{1/2+\epsilon}r^{-1/2}\}\right] &\leq \sum_{s=f(t)}^{t}\sum_{r=1}^{s}\mathbf{Pr}\left[d_s(r)^{(q)}\geq (s^{1/2+\epsilon}r^{-1/2})^{(q)}\right]\\
&\leq \sum_{s=f(t)}^{t}\sum_{r=1}^{s}\mathbf{Pr}\left[d_s(r)^{(q)}\geq (s^{-(q/2+q\epsilon)}r^{q/2})\right]\\
&\leq \sum_{s=f(t)}^{t}\sum_{r=1}^{s}\frac{(q+2)!}{2}\Big(\frac{2s}{r}\Big)^{q/2}s^{-q/2}s^{-q\epsilon}r^{q/2}\\
&= \frac{(q+2)!}{2}2^{q/2}\sum_{s=f(t)}^{t}s^{1-q\epsilon}\\
&\leq \frac{(q+2)!}{2}2^{q/2}\int_{f(t)-1}^{t}x^{1-q\epsilon}\,\mathrm{d}x\\
&\leq \frac{(q+2)!}{2(q\epsilon-2)}2^{q/2}(f(t)-1)^{2-q\epsilon}=o(1).
\end{aligned}$$

□

Lemma 13. *Let S_3' be the set of vertices in S_3 which are adjacent to more than one vertex of S_1. Then $|S_3'|\leq t^{1/6}$ whp.*

Proof. First, observe that when vertex s is inserted it becomes adjacent to more than one vertex of S_1 if the face chosen by s has at least two vertices in S_1. We call the latter property $\mathcal{A}$ and we write $s\in\mathcal{A}$ when s satisfies it. At time t_1 there exist $2t_1+1$ faces total, which consist of faces whose three vertices are all from S_1. At time $s\geq t_2$ there can be at most $6t_1+3$ faces with at least two vertices in S_1 since each of the original $2t_1+1$ faces can give rise to at most 3 new faces with at least two vertices in s_1. Consider a vertex $s\in S_3$, i.e., $s\geq t_2$. By the above argument, $\mathbf{Pr}\left[|N(s)\cap S_1|\geq 2\right]\leq\frac{6t_1+3}{2t+1}$. Writing $|S_3'|$ as a sum of indicator variables, i.e., $|S_3'|=\sum_{s=t_2}^{t}I(s\in\mathcal{A})$ and taking the expectation we obtain

$$\begin{aligned}
\mathbb{E}\left[|S_3'|\right]&\leq\sum_{s=t_2}^{t}\frac{6t_1+3}{2t+1}\leq(6t_1+3)\int_{t_2}^{t}(2x+1)^{-1}\,\mathrm{d}x\\
&\leq(3t^{\frac{1}{8}}+\tfrac{3}{2})\ln\frac{2t+1}{2t_2+1}=o(t^{1/7})
\end{aligned}$$

By Markov's inequality:

$$\mathbf{Pr}\left[|S_3'|\geq t^{1/6}\right]\leq\frac{\mathbb{E}\left[|S_3'|\right]}{t^{1/6}}=o(1).$$

Therefore, we conclude that $|S_3'|\leq t^{1/6}$ *whp*. □

Lemma 14. *Let $F\subseteq G$ be the star forest consisting of edges between S_1 and S_3-S_3'. Let $\Delta_1\geq\Delta_2\geq\ldots\geq\Delta_k$ denote the k highest degrees of G. Then $\lambda_i(F)=(1-o(1))\sqrt{\Delta_i}$ whp.*

Proof. It suffices to show that $\Delta_i(F) = (1 - o(1))\Delta_i(G)$ for $i = 1, \ldots, k$. Note that since the k highest vertices are inserted before t_1 *whp*, the edges they lose are the edges between S_1 and the ones incident to S_3' and S_2 and we know how to bound the cardinalities of all these sets. Specifically by Lemma 13 $|S_3'| \leq t^{1/6}$ *whp* and by Theorem 2 the maximum degree in G_{t_1}, G_{t_2} is less than $t_1^{1/2+\epsilon_1} = t^{1/8}$, $t_2^{1/2+\epsilon_2} = t^{5/16}$ for $\epsilon_1 = 1/16, \epsilon_2 = 1/32$ respectively *whp*. Also by Theorem 2, $\Delta_i(G) \geq \frac{\sqrt{t}}{\log t}$. Hence, we obtain

$$\Delta_i(F) \geq \Delta_i(G) - t^{1/8} - t^{5/16} - t^{1/6} = (1 - o(1))\Delta_i(G).$$

□

To complete the proof of Theorem 3 it suffices to prove that $\lambda_1(H)$ is $o(\lambda_k(F))$ where $H = G - F$. We prove this in the following lemma. The proof is based on bounding maximum degree of appropriately defined subgraphs using Lemma 12 and standard inequalities from spectral graph theory [13].

Lemma 15. $\lambda_1(H) = o(t^{1/4})$ *whp.*

Proof. From Gershgorin's theorem [32] the maximum eigenvalue of any graph is bounded by the maximum degree. We bound the eigenvalues of H by bounding the maximum eigenvalues of six different induced subgraphs. Specifically, let $H_i = H[S_i]$, $H_{ij} = H(S_i, S_j)$ where $H[S]$ is the subgraph induced by the vertex set S and $H(S, T)$ is the subgraph containing only edges with one vertex is S and other in T. We use Lemma 14 to bound $\lambda_1(H(S_1, S_3))$ and Lemma 13 for the other eigenvalues. We set $\epsilon = 1/64$.

$$\lambda_1(H_1) \leq \Delta_1(H_1) \leq t_1^{1/2+\epsilon} = t^{33/512}.$$
$$\lambda_1(H_2) \leq \Delta_1(H_2) \leq t_2^{1/2+\epsilon} t_1^{-1/2} = t^{233/1024}.$$
$$\lambda_1(H_3) \leq \Delta_1(H_3) \leq t_3^{1/2+\epsilon} t_2^{-1/2} = t^{15/64}.$$
$$\lambda_1(H_{12}) \leq \Delta_1(H_{12}) \leq t_2^{1/2+\epsilon} = t^{297/1024}.$$
$$\lambda_1(H_{23}) \leq \Delta_1(H_{23}) \leq t_3^{1/2+\epsilon} t_1^{-1/2} = t^{29/64}.$$
$$\lambda_1(H_{13}) \leq \Delta_1(H_{13}) \leq t^{1/6}.$$

Therefore *whp* we obtain

$$\lambda_1(H) \leq \sum_{i=1}^{3} \lambda_1(H_i) + \sum_{i<j} \lambda_1(H_{i,j}) = o(t^{1/4}).$$

□

6 Proof of Theorem 4

Before we give the proof of Theorem 4, we give a simple proof that the diameter of a RAN is $O(\log t)$ *whp*.

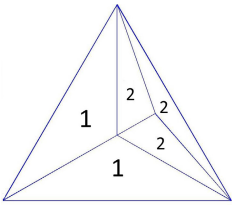


Fig. 3. An instance of the process for $t = 2$. Each face is labeled with its depth.

We begin with a necessary definition for the proof of Claim (2). We define the *depth of a face* recursively. Initially, we have three faces, see Figure 1a, whose depth equals 1. For each new face β created by picking a face γ, we have $depth(\beta) = depth(\gamma) + 1$. An example is shown in Figure 3, where each face is labeled with its corresponding depth.

Claim (2). The diameter $d(G_t)$ satisfies $d(G_t) = O(\log t)$ *whp*.

Proof. A simple but key observation is that if k^* is the maximum depth of a face then $d(G_t) = O(k^*)$. Hence, we need to upper bound the depth of a given face after t rounds. Let $F_t(k)$ be the number of faces of depth k at time t, then:

$$\mathbb{E}\left[F_t(k)\right] = \sum_{1 \le t_1 < t_2 < \ldots < t_k \le t} \prod_{j=1}^{k} \frac{1}{2t_j + 1} \le \frac{1}{k!}\left(\sum_{j=1}^{t} \frac{1}{2j+1}\right)^k \le \frac{1}{k!}\left(\frac{1}{2}\log t\right)^k \le \left(\frac{e \log t}{2k}\right)^{k+1}$$

By the first moment method we obtain $k^* = O(\log t)$ *whp* and by our observation $d(G_t) = O(\log t)$ *whp*. □

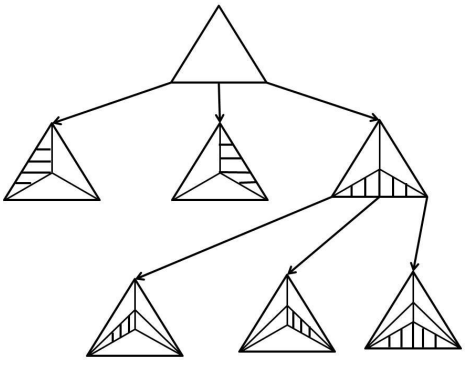


Fig. 4. RANs as random ternary trees

The depth of a face can be formalized via a bijection between random ternary trees and RANs. Using this bijection we prove Theorem 4 which gives a refined upper bound on the asymptotic growth of the diameter.

Proof. Consider the random process which starts with a single vertex tree and at every step picks a random leaf and adds three children to it. Let T be the resulting tree after t steps. There exists a natural bijection between the RAN process and this process, see [18] and also Figure 4. The depth of T in probability is $\frac{\rho}{2}\log t$ where $\frac{1}{\rho} = \eta$ is the unique solution greater than 1 of the equation $\eta - 1 - \log\eta = \log 3$, see Broutin and Devroye [12], pp. 284-285. Note that the diameter $d(G_t)$ is at most twice the height of the tree and hence the result follows. □

The above observation, i.e., the bijection between RANs and random ternary trees cannot be used to lower bound the diameter. A counterexample is shown in Figure 5 where the height of the random ternary tree can be made arbitrarily large but the diameter is 2. Albenque and Marckert proved in [2] that if v, u are two i.i.d. uniformly random internal vertices, i.e., $v, u \geq 4$, then the distance $d(u, v)$ tends to $\frac{6}{11}\log n$ with probability 1 as the number of vertices n of the RAN grows to infinity. However, an exact expression of the asymptotic growth of the diameter to the best of our knowledge remains an open problem. Finally, it is worth mentioning that the diameter of the RAN grows faster asymptotically than the diameter of the classic preferential attachment model [7] which *whp* grows as $\frac{\log t}{\log\log t}$, see Bollobás and Riordan [9].

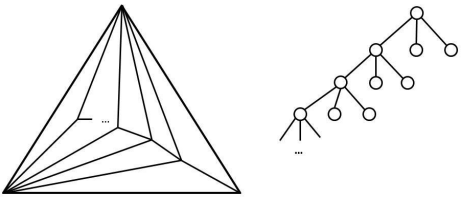


Fig. 5. The height of the random ternary tree cannot be used to lower bound the diameter. The height of the random ternary tree can be arbitrarily large but the diameter is 2.

7 Waiting Times

Consider the three initial faces of Figure 1a. Let's call the face which receives the first vertex A and the other two faces B, C. Let X equal the number of steps until a new vertex picks face B or C. Clearly, $X \in \{1, 2, \ldots\}$. What is the expectation $\mathbb{E}[X]$? For any $t \geq 1$

$$\mathbf{Pr}[X > t] = \prod_{j=1}^{t} \frac{3 + 2(j-1)}{5 + 2(j-1)} = \frac{3}{2t+3}.$$

Using now the fact that $\mathbb{E}[X] = \sum_{t=1}^{+\infty} \mathbf{Pr}[X \geq t] = 1 + \sum_{t=1}^{+\infty} \mathbf{Pr}[X > t]$ we obtain that $\mathbb{E}[X] = +\infty$.

8 Open Problems

We propose three open problems for future work. The first concerns the diameter. Specifically, as we mentioned also earlier, an interesting problem is to find an exact asymptotic expression for the diameter of RAN.

Conductance: We conjecture that the conductance of a RAN is $\Theta\left(\frac{1}{\sqrt{t}}\right)$ *whp*. Figure 6 shows that $\Phi(G_t) \leq \frac{1}{\sqrt{t}}$.

Hamiltonicity and Longest Path: We conjecture that *whp* a RAN is not Hamiltonian but the length of the longest path is $\Omega(n)$.

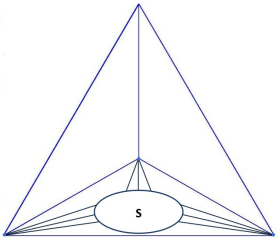


Fig. 6. By the pigeonhole principle, one of the three initial faces receives $\Theta(t)$ vertices. Using Theorem 2 it is not hard to see that the encircled set of vertices S has conductance $\phi(S) \approx \frac{\sqrt{t}}{t} = \frac{1}{\sqrt{t}}$ *whp*.

Acknowledgements. Research supported by NSF Grant No. CCF-1013110. We would like to thank Luc Devroye and Alexis Darrasse for pointing out references [12] and [2,30] respectively.

References

1. Aiello, W., Chung, F., Lu, L.: A random graph model for power law graphs. Experimental Mathemathics 10(1), 53–66 (2001)
2. Albenque, M., Marckert, J.F.: Some families of increasing planar maps. Electronic Journal of Probability 13, 1624–1671
3. Alon, N., Spencer, J.: The Probabilistic Method. Wiley-Interscience (2008)
4. Andrade, J.S., Herrmann, H.J., Andrade, R.F.S., da Silva, L.R.: Apollonian networks: simultaneously scale-free, small world Euclidean, space filling, and with matching graphs. Phys. Rev. Lett. 94, 018702 (2005)
5. Andrade, R.F.S., Miranda, J.G.V.: Spectral Properties of the Apollonian Network. Physica A 356 (2005)
6. Azuma, K.: Weighted sums of certain dependent variables. Tohoku Math. J 3, 357–367 (1967)
7. Barabási, A., Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509–512 (1999)
8. Bodini, O., Darrasse, A., Soria, M.: Distances in random Apollonian network structures Arxiv, http://arxiv.org/abs/0712.2129
9. Bollobás, B., Riordan, O.: The Diameter of a Scale-Free Random Graph. Combinatorica (2002)

10. Bollobás, B., Riordan, O., Spencer, J., Tusnády, G.: The Degree Sequence of a Scale Free Random Graph Process. Random Struct. Algorithms 18(3), 279–290 (2001)
11. Boyd, D.W.: The Sequence of Radii of the Apollonian Packing. Mathematics of Computation 19, 249–254 (1982)
12. Broutin, N., Devroye, L.: Large Deviations for the Weighted Height of an Extended Class of Trees. Algorithmica 46, 271–297 (2006)
13. Chung Graham, F.: Spectral Graph Theory. American Mathematical Society (1997)
14. Chung Graham, F., Lu, L.: Complex Graphs and Networks, (107). American Mathematical Society (2006)
15. Chung, F., Lu, L., Vu, V.H.: Spectra of random graphs with given expected degrees. Proceedings of the National Academy of Sciences of the United States of America 100, 6313–6318
16. Cooper, C., Frieze, A.: A general model of web graphs. Random Structures & Algorithms 22(3), 311–335 (2003)
17. Cooper, C., Uehara, R.: Scale Free Properties of random k-trees. Mathematics in Computer Science 3(4), 489–496 (2010)
18. Darrasse, A., Soria, M.: Degree distribution of random Apollonian network structures and Boltzmann sampling. In: 2007 Conference on Analysis of Algorithms, AofA 2007, DMTCS Proceedings (2007)
19. Darrasse, A., Hwang, H.-K., Bodini, O., Soria, M.: The connectivity-profile of random increasing k-trees, Arxiv, http://arxiv.org/abs/0910.3639
20. Doye, J.P.K., Massen, C.P.: Self-similar disk packings as model spatial scale-free networks. Phys. Rev. E 71, 016128 (2005)
21. Fabrikant, A., Koutsoupias, E., Papadimitriou, C.: Heuristically Optimized Trade-Offs: A New Paradigm for Power Laws in the Internet. In: Widmayer, P., Triguero, F., Morales, R., Hennessy, M., Eidenbenz, S., Conejo, R. (eds.) ICALP 2002. LNCS, vol. 2380, pp. 110–122. Springer, Heidelberg (2002)
22. Flaxman, A., Frieze, A., Fenner, T.: High Degree Vertices and Eigenvalues in the Preferential Attachment Graph. Internet Mathematics 2(1) (2005)
23. Gao, Y.: The degree distribution of random k-trees. Theoretical Computer Science 410(8-10) (2009)
24. Gao, Y., Hobson, C.: Random k-tree as a model for complex networks. In: Workshop on Algorithms and Models for the Web-Graph, WAW (2006)
25. Graham, R.L., Lagarias, J.C., Mallows, C.L., Wilks, A.R., Yan, C.H.: Apollonian Circle Packings: Number Theory. J. Number Theory 100(1), 1–45, MR1971245
26. Hoeffding, W.: Probability inequalities for sumes of bounded random variables. J. Amer. Statist. Assoc. 58, 13–30 (1963)
27. Kloks, T.: Treewidth: Computations and Approximations. Springer (1994)
28. Leskovec, J., Faloutsos, C.: Scalable modeling of real graphs using Kronecker multiplication. In: Machine Learning Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvalis, Oregon, USA, June 20-24 (2007)
29. Mihail, M., Papadimitriou, C.: On the Eigenvalue Power Law. In: Rolim, J.D.P., Vadhan, S.P. (eds.) RANDOM 2002. LNCS, vol. 2483, pp. 254–262. Springer, Heidelberg (2002)
30. Panholzer, A., Seitz, G.: Ordered increasing k-trees: Introduction and analysis of a preferential attachment network model. In: DMTCS Proc., AofA 2010, pp. 549–564 (2010)
31. Pralat, P., Wormald, N.: Growing Protean Graphs. Internet Mathematics 4(1), 1–16 (2007)
32. Strang, G.: Linear Algebra and Its Applications. Brooks Cole (2005)
33. Wu, Z.-X., Xu, X.-J., Wang, Y.-H.: Comment on "Maximal planar networks with large clustering coefficient and power-law degree distribution". Physical Review, E 73, 058101 (2006)
34. Zhang, Z.Z., Comellas, F., Fertin, G., Rong, L.L.: High dimensional Apollonian networks, ArXiv, http://arxiv.org/abs/cond-mat/0503316
35. Zhou, T., Yan, G., Wang, B.H.: Maximal planar networks with large clustering coefficient and power-law degree distribution. Phys. Rev. E 71, 046141 (2005)

Appendix

Proof of Lemma 6

Proof. As we mentioned in the proof of Theorem 1 the three initial vertices $1, 2, 3$ have one less face than their degree whereas all other vertices have degree equal to the number of faces surrounding them. In this proof we treat both cases but we omit it in all other proofs.

• CASE 1: $s \geq 4$

Note that $d_s(s) = 3$. By conditioning successively we obtain

$$\begin{aligned}
\mathbb{E}\left[d_t(s)^{(k)}\right] &= \mathbb{E}\left[\mathbb{E}\left[d_t(s)^{(k)}|d_{t-1}(s)\right]\right] \\
&= \mathbb{E}\left[(d_{t-1}(s))^{(k)}\big(1-\frac{d_{t-1}(s)}{2t-1}\big)+(d_{t-1}(s)+1)^{(k)}\frac{d_{t-1}(s)}{2t-1}\right] \\
&= \mathbb{E}\left[(d_{t-1}(s))^{(k)}\big(1-\frac{d_{t-1}(s)}{2t-1}\big)+(d_{t-1}(s))^{(k)}\frac{d_{t-1}(s)+k}{d_{t-1}(s)}\frac{d_{t-1}(s)}{2t-1}\right] \\
&= \mathbb{E}\left[(d_{t-1}(s))^{(k)}\right]\big(1+\frac{k}{2t-1}\big) = \ldots = 3^{(k)}\prod_{t'=s+1}^{t}\big(1+\frac{k}{2t'-1}\big) \\
&\leq 3^{(k)}\exp\Big(\sum_{t'=s+1}^{t}\frac{k}{2t'-1}\Big) \leq 3^{(k)}\exp\Big(k\int_{s}^{t}\frac{\mathrm{d}x}{2x-1}\Big) \\
&\leq \frac{(k+2)!}{2}\exp\Big(\tfrac{k}{2}\log\frac{t-1/2}{s-1/2}\Big) \leq \frac{(k+2)!}{2}\Big(\frac{2t}{s}\Big)^{\frac{k}{2}}.
\end{aligned}$$

• CASE 2: $s \in \{1, 2, 3\}$

Note that initially the degree of any such vertex is 2. For any $k \geq 0$

$$\begin{aligned}
\mathbb{E}\left[d_t(s)^{(k)}\right] &= \mathbb{E}\left[\mathbb{E}\left[d_t(s)^{(k)}|d_{t-1}(s)\right]\right] \\
&= \mathbb{E}\left[(d_{t-1}(s))^{(k)}\big(1-\frac{d_{t-1}(s)-1}{2t-1}\big)+(d_{t-1}(s)+1)^{(k)}\frac{d_{t-1}(s)-1}{2t-1}\right] \\
&= \mathbb{E}\left[(d_{t-1}(s))^{(k)}\big(1+\frac{k}{2t-1}\big)-(d_{t-1}(s))^{(k)}\frac{k}{(2t-1)d_{t-1}(s)}\right] \\
&\leq \mathbb{E}\left[(d_{t-1}(s))^{(k)}\right]\big(1+\frac{k}{2t-1}\big) \leq \ldots \leq \frac{(k+2)!}{2}\big(\frac{2t}{s}\big)^{\frac{k}{2}}.
\end{aligned}$$

□

Proof of Claim (1)

Proof. Let $\tau = (t_0 \equiv \tau_0, \underbrace{\tau_1, \ldots, \tau_r}_{\text{insertion times}}, \tau_{r+1} \equiv t)$ be a vector denoting that Y_t increases by 1 at τ_i for $i = 1, \ldots, r$. We upper bound the probability p_τ of this event in the

following.Note that we consider the case where the vertices have same degree as the number of faces around them. As we mentioned earlier, the other case is analyzed in exactly the same way, modulo a negligible error term.

$$\begin{aligned} p_\tau &= \left[\prod_{k=1}^{r} \frac{d_0+k-1}{2\tau_k+1}\right]\left[\prod_{k=0}^{r}\prod_{j=\tau_k+1}^{\tau_{k+1}-1}\left(1-\frac{d_0+k}{2j+1}\right)\right] \\ &= d_0(d_0+1)\ldots(d_0+r-1)\left[\prod_{k=1}^{r}\frac{1}{2\tau_k+1}\right]\exp\left(\sum_{k=0}^{r}\sum_{j=\tau_k+1}^{\tau_{k+1}-1}\log\left(1-\frac{d_0+k}{2j+1}\right)\right) \\ &= \frac{(d_0+r-1)!}{(d_0-1)!}\left[\prod_{k=1}^{r}\frac{1}{2\tau_k+1}\right]\exp\left(\sum_{k=0}^{r}\sum_{j=\tau_k+1}^{\tau_{k+1}-1}\log\left(1-\frac{d_0+k}{2j+1}\right)\right) \end{aligned}$$

Consider now the inner sum which we upper bound using an integral:

$$\begin{aligned} \sum_{j=\tau_k+1}^{\tau_{k+1}-1}\log\left(1-\frac{d_0+k}{2j+1}\right) &\le \int_{\tau_k+1}^{\tau_{k+1}}\log\left(1-\frac{d_0+k}{2x+1}\right)\mathrm{d}x \\ &\le -\left(\tau_{k+1}+\tfrac{1}{2}\right)\log\left(2\tau_{k+1}+1\right)+ \\ &\frac{2\tau_{k+1}+1-(d_0+k)}{2}\log\left(2\tau_{k+1}+1-(d_0+k)\right)+ \\ &\left(\tau_k+\tfrac{3}{2}\right)\log\left(2\tau_k+3\right)-\frac{2\tau_k+3-(d_0+k)}{2}\log\left(2\tau_k+3-(d_0+k)\right) \end{aligned}$$

since

$$\int\log\left(1-\frac{d_0+k}{2x+1}\right) = -\left(x+\tfrac{1}{2}\right)\log\left(2x+1\right)+\frac{2x+1-(d_0+k)}{2}\log\left(2x+1-(d_0+k)\right)$$

Hence we obtain $\sum_{k=0}^{r}\sum_{j=\tau_k+1}^{\tau_{k+1}-1}\log\left(1-\frac{d_0+k}{2j+1}\right) \le A+\sum_{k=1}^{r}B_k$ where

$$\begin{aligned} A &= \left(\tau_0+\tfrac{3}{2}\right)\log\left(2\tau_0+3\right)-\frac{2\tau_0+3-d_0}{2}\log\left(2\tau_0+3-d_0\right) \\ &-\left(\tau_{r+1}+\tfrac{1}{2}\right)\log\left(2\tau_{r+1}+1\right)+\frac{2\tau_{r+1}+1-(d_0+r)}{2}\log\left(2\tau_{r+1}+1-(d_0+r)\right) \end{aligned}$$

and

$$\begin{aligned} B_k &= \left(\tau_k+\tfrac{3}{2}\right)\log\left(2\tau_k+3\right)-\frac{2\tau_k+3-(d_0+k)}{2}\log\left(2\tau_k+3-(d_0+k)\right) \\ &-\left(\tau_k+\tfrac{1}{2}\right)\log\left(2\tau_k+1\right)+\frac{2\tau_k+1-(d_0+k-1)}{2}\log\left(2\tau_k+1-(d_0+k-1)\right). \end{aligned}$$

We first upper bound the quantities B_k for $k=1,\ldots,r$. By rearranging terms and using the identity $\log(1+x)\le x$ we obtain

$$B_k = \left(\tau_k + \tfrac{1}{2}\right)\log\left(1 + \frac{1}{\tau_k + \frac{1}{2}}\right) + \log\left(2\tau_k + 3\right)$$
$$- \frac{1}{2}\log\left(2\tau_k + 3 - (d_0 + k)\right) - \frac{2\tau_k + 2 - (d_0 + k)}{2}\log\left(1 + \frac{1}{2\tau_k + 2 - (d_0 + k)}\right).$$
$$\leq \frac{1}{2} + \frac{1}{2}\log\left(2\tau_k + 3\right) - \frac{1}{2}\log\left(1 - \frac{d_0 + k}{2\tau_k + 3}\right)$$

First we rearrange terms and then we bound the term e^A by using the inequality $e^{-x-x^2/2} \geq 1 - x$ which is valid for $0 < x < 1$:

$$A = -\left(\tau_0 + \tfrac{3}{2}\right)\log\left(1 - \frac{d_0}{2\tau_0 + 3}\right) + \left(\tau_{r+1} + \tfrac{1}{2}\right)\log\left(1 - \frac{d_0 + r}{2\tau_{r+1} + 1}\right) + \frac{d_0}{2}\log\left(2\tau_0 + 3 - d_0\right)$$
$$- \frac{d_0 + r}{2}\log\left(2\tau_{r+1} + 1 - (d_0 + r)\right). \Rightarrow$$
$$e^A = \left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2})}\left(1 - \frac{d_0 + r}{2\tau_{r+1} + 1}\right)^{\tau_{r+1} + \frac{1}{2}}\left(2\tau_0 + 3 - d_0\right)^{\frac{d_0}{2}}\left(2\tau_{r+1} + 1 - (d_0 + r)\right)^{-\frac{d_0 + r}{2}}$$
$$= \left(\frac{2\tau_0 + 3}{2\tau_{r+1} + 1}\right)^{d_0/2}(2\tau_{r+1} + 1)^{-r/2}\left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2}) + \frac{d_0}{2}}\left(1 - \frac{d_0 + r}{2\tau_{r+1} + 1}\right)^{\tau_{r+1} + \frac{1}{2} - \frac{d_0 + r}{2}}$$
$$\leq \left(\frac{2t_0 + 3}{2t + 1}\right)^{d_0/2}(2t + 1)^{-r/2}\left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2}) + \frac{d_0}{2}} e^{\left(-\frac{d_0 + r}{2t+1} - \left(-\frac{d_0 + r}{2t+1}\right)^2/2\right)(t + 1/2 - \frac{d_0 + r}{2})}$$
$$= \left(\frac{2t_0 + 3}{2t + 1}\right)^{d_0/2}(2t + 1)^{-r/2}\left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2}) + \frac{d_0}{2}} e^{-\frac{d_0 + r}{2} + \frac{(d_0 + r)^2}{8t + 4} + \frac{(d_0 + r)^3}{4(2t+1)^2}}$$

Now we upper bound the term $\exp\left(A + \sum_{k=1}^r B_k\right)$ using the above upper bounds:

$$e^{A + \sum_{k=1}^r B_k} \leq e^A e^{r/2}\prod_{i=1}^r \sqrt{\frac{2\tau_k + 3}{1 - \frac{d_0 + k}{2\tau_k + 3}}}$$
$$\leq \left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2}) + \frac{d_0}{2}} e^{-\frac{d_0}{2} + \frac{(d_0 + r)^2}{8t + 4} + \frac{(d_0 + r)^3}{4(2t+1)^2}}\left(\frac{2t_0 + 3}{2t + 1}\right)^{d_0/2} \times$$
$$(2t + 1)^{-r/2}\prod_{i=1}^r \sqrt{\frac{2\tau_k + 3}{1 - \frac{d_0 + k}{2\tau_k + 3}}}$$

Using the above upper bound we get that

$$p_\tau \leq C(r, d_0, t_0, t)\prod_{k=1}^r \left[\left(2\tau_k + 3 - (d_0 + k)\right)^{-1/2}\left(1 + \frac{1}{\tau_k + 1/2}\right)\right]$$

where

$$C(r, d_0, t_0, t) = \frac{(d_0 + r - 1)!}{(d_0 - 1)!}\left(1 - \frac{d_0}{2\tau_0 + 3}\right)^{-(\tau_0 + \frac{3}{2}) + \frac{d_0}{2}} e^{-\frac{d_0}{2} + \frac{(d_0 + r)^2}{8t + 4} + \frac{(d_0 + r)^3}{4(2t+1)^2}}\left(\frac{2t_0 + 3}{2t + 1}\right)^{d_0/2}(2t + 1)^{-r/2}$$

We need to sum over all possible insertion times to bound the probability of interest p^*. We set $\tau'_k \leftarrow \tau_k - \lceil \frac{d_0 + k}{2} \rceil$ for $k = 1, \ldots, r$. For $d = o(\sqrt{t})$ and $r = o(t^{2/3})$ we obtain:

$$
\begin{aligned}
p^* &\le C(r,d_0,t_0,t) \sum_{t_0+1\le\tau_1<..<\tau_r\le t} \prod_{k=1}^{r} \Big[(2\tau_k+3-(d_0+k))^{-1/2}\big(1+\frac{1}{\tau_k+1/2}\big)\Big] \\
&\le C(r,d_0,t_0,t) \sum_{t_0-\lceil\frac{d_0}{2}\rceil+1\le\tau_1'\le..\le\tau_r'\le t-\lceil\frac{d_0+r}{2}\rceil} \prod_{k=1}^{r} \Big[(2\tau_k'+3)^{-1/2}\big(1+\frac{1}{\tau_k'+\frac{d_0+k}{2}+1/2}\big)\Big] \\
&\le \frac{C(r,d_0,t_0,t)}{r!}\left(\sum_{t_0-\lceil\frac{d_0}{2}\rceil}^{t-\lceil\frac{d_0+r}{2}\rceil}(2\tau_k'+3)^{-1/2}+\tfrac{1}{\sqrt{2}}(\tau_k'+3/2)^{-3/2}\right)^r \\
&\le \frac{C(r,d_0,t_0,t)}{r!}\left(\int_0^{t-\frac{d+r}{2}}\Big[(2x+3)^{-1/2}+\tfrac{1}{\sqrt{2}}(x+3/2)^{-3/2}\Big]\,\mathrm{d}x\right)^r \\
&\le \frac{C(r,d_0,t_0,t)}{r!}\Big(\sqrt{2t+3-(d_0+r)}+2/3\Big)^r \\
&\le \frac{C(r,d_0,t_0,t)}{r!}(2t)^{r/2}e^{-\frac{r}{2}\frac{d_0+r-3}{2t}}e^{\frac{2r}{3\sqrt{2t-(d_0+r)+3}}} \\
&\le \binom{d_0+r-1}{d_0-1}\big(\frac{2t_0+3}{2t+1}\big)^{d_0/2}\Big[\big(1-\frac{d_0}{2t_0+3}\big)^{-(1-\frac{d_0}{2t_0+3})}\Big]^{t_0+3/2}\times \\
&\big(\frac{2t}{2t+1}\big)^{r/2}\exp\Big(-\frac{d_0}{2}+\frac{(d_0+r)^2}{8t+4}+\frac{(d_0+r)^3}{4(2t+1)^2}-\frac{r(d_0+r-3)}{4t}+\frac{2r}{3\sqrt{2t+3-(d_0+r)}}\Big)
\end{aligned}
$$

By removing the $o(1)$ terms in the exponential and using the fact that $x^{-x} \le e$ we obtain the following bound on the probability p^*.

$$p^* \le \binom{d_0+r-1}{d_0-1}\Big(\frac{2t_0+3}{2t+1}\Big)^{d_0/2}e^{\frac{3}{2}+t_0-\frac{d_0}{2}+\frac{2r}{3\sqrt{t}}}.$$

□

Lemma 11

Proof. Let $\mathcal{A}_4$ denote the event that there are two vertices among the first t_1 with degree $t_0^{-1}t^{1/2}$ and within $\frac{\sqrt{t}}{f(t)}$ of each other. By the definition of conditional probability and Lemma 8

$$\mathbf{Pr}\left[\mathcal{A}_4\right] = \mathbf{Pr}\left[\mathcal{A}_4|\bar{\mathcal{A}}_3\right]\mathbf{Pr}\left[\bar{\mathcal{A}}_3\right] + \mathbf{Pr}\left[\mathcal{A}_4|\mathcal{A}_3\right]\mathbf{Pr}\left[\mathcal{A}_3\right] \le \mathbf{Pr}\left[\mathcal{A}_4|\bar{\mathcal{A}}_3\right] + o(1)$$

it suffices to show that $\mathbf{Pr}\left[\mathcal{A}_4|\bar{\mathcal{A}}_3\right] = o(1)$. Note that by a simple union bound

$$\mathbf{Pr}\left[\mathcal{A}_4\right] \le \sum_{1\le s_1<s_2\le t_1}\sum_{l=-\frac{\sqrt{t}}{f(t)}}^{\frac{\sqrt{t}}{f(t)}} p_{l,s_1,s_2} = O\Big(t_1^2\frac{\sqrt{t}}{f(t)}\max p_{l,s_1,s_2}\Big)$$

where $p_{l,s_1,s_2} = \mathbf{Pr}\left[d_t(s_1)-d_t(s_2)=l|\bar{\mathcal{A}}_3\right]$.

We consider two cases and we show that in both cases $\max p_{l,s_1,s_2} = o(\frac{f(t)}{t_1^2\sqrt{t}})$.

• CASE 1 $(s_1, s_2) \notin E(G_t)$:

Note that at time t_1 there exist $m_{t_1} = 3t_1 + 3 < 4t_1$ edges in G_{t_1}.

$$p_{l,s_1,s_2} \leq \sum_{r=t_0^{-1}t^{1/2}}^{t_0^{1/6}t^{1/2}} \sum_{d_1,d_2=3}^{4t_1} \mathbf{Pr}\left[d_t(s_1) = r \wedge d_t(s_2) = r - l | d_{t_1}(s_1) = d_1, d_{t_1}(s_2) = d_2\right] \tag{2}$$

$$\leq t_0^{1/6}t^{1/2} \sum_{d_1,d_2=3}^{4t_1} \binom{2t_0^{1/6}t^{1/2}}{d_1 - 1} \binom{2t_0^{1/6}t^{1/2}}{d_2 - 1} \Big(\frac{2t_0+3}{2t+1}\Big)^{(d_1+d_2)/2} e^{\frac{3}{2}+t_1+\frac{2t_0^{1/6}}{3}} \tag{3}$$

$$\leq t_0^{1/6}t^{1/2} \sum_{d_1,d_2=3}^{4t_1} (2t_0^{1/6}t^{1/2})^{d_1+d_2-2} \Big(\frac{2t_0+3}{2t+1}\Big)^{(d_1+d_2)/2} e^{2t_1}$$

$$\leq t_0^{1/6}t^{1/2} e^{2t_1} t_1^2 (2t_0^{1/6}t^{1/2})^{8t_1-2} \Big(\frac{2t_0+3}{2t+1}\Big)^{4t_1}$$

$$= t_0^{4t_1/3+1/6} t^{-1/2} e^{2t_1} t_1^2 2^{8t_1} (2t_0+3)^{4t_1} \Big(\frac{t}{2t+1}\Big)^{4t_1}$$

$$= o\Big(\frac{f(t)}{t_1^2\sqrt{t}}\Big)$$

Note that we omitted the tedious calculation justifying the transition from (2) to (3) since calculating the upper bound of the joint probability distribution is very similar to the calculation of Lemma 7.

• CASE 2 $(s_1, s_2) \in E(G_t)$:

Notice that in any case (s_1, s_2) share at most two faces (which may change over time). Note that the two connected vertices s_1, s_2 share a common face only if $s_1, s_2 \in \{1, 2, 3\}$[3]. Consider the following modified process $\mathcal{Y}'$: whenever an incoming vertex "picks" one of the two common faces we don't insert it. We choose two other faces which are not common to s_1, s_2 and add one vertex in each of those. Notice that the number of faces increases by 1 for both s_1, s_2 as in the original process and the difference of the degrees remains the same. An algebraic manipulation similar to Case 1 gives the desired result. □

[3] We analyze the case where $s_1, s_2 \geq 4$. The other case is treated in the same manner.

Mutual or Unrequited Love: Identifying Stable Clusters in Social Networks with Uni- and Bi-directional Links

Yanhua Li*, Zhi-Li Zhang, and Jie Bao

University of Minnesota, Twin Cities
{yanhua,zhzhang,baojie}@cs.umn.edu

Abstract. Many social networks, e.g., Slashdot and Twitter, can be represented as directed graphs (*digraphs*) with two types of links between entities: mutual (bi-directional) and one-way (uni-directional) connections. Social science theories reveal that mutual connections are more stable than one-way connections, and one-way connections exhibit various tendencies to become mutual connections. It is therefore important to take such tendencies into account when performing clustering of social networks with both mutual and one-way connections.

In this paper, we utilize the *dyadic* methods to analyze social networks, and develop a generalized mutuality tendency theory to capture the tendencies of those node pairs which tend to establish mutual connections more frequently than those occur by chance. Using these results, we develop a *mutuality-tendency-aware* spectral clustering algorithm to identify more stable clusters by maximizing the *within-cluster* mutuality tendency and minimizing the *cross-cluster* mutuality tendency. Extensive simulation results on synthetic datasets as well as real online social network datasets such as Slashdot, demonstrate that our proposed mutuality-tendency-aware spectral clustering algorithm extracts more stable social community structures than traditional spectral clustering methods.

1 Introduction

Graph models are widely utilized to represent relations among entities in social networks. Especially, many online social networks, e.g., Slashdot and Twitter, where the users' social relationships are represented as directed edges in directed graphs (or in short, *digraphs*). Entity connections in a digraph can be categorized into two types, namely, bi-directional links (mutual connections) and uni-directional links (*one-way* connections). Social theories [28] and online social network analysis [2, 6, 28] have revealed that various types of connections exhibit different stabilities, where mutual connections are more stable than one-way connections. In other words, mutual connections are the source of social cohesion [3, 4] that, if two individuals mutually attend to one another, then the bond is reinforced in each direction.

Studying the social network structure and properties of social ties have been an active area of research. Clustering and identifying social structures in social networks is an especially important problem [8, 17, 24] that has wide applications, for instance, community detection and friend recommendation in social networks. Existing clustering

* The work is supported in part by the NSF grants CNS-0905037, CNS-1017647 and the DTRA Grant HDTRA1-09-1-0050.

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 113–125, 2012.

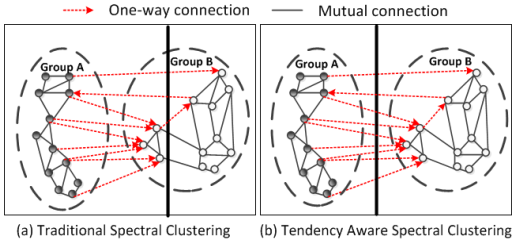


Fig. 1. An example network

methods [21,29] are originally developed for *undirected* graphs, based on the classical *spectral clustering theory*. Several recent studies (see, e.g., [10,21,27,29]) extend the spectral clustering method to digraphs, by first converting the underlying digraphs to undirected graphs via some form of *symmetrization*, and then apply spectral clustering to the resulting symmetrized (undirected) graphs. However, all these methods have two common drawbacks, which prevent them from obtaining *stable* clusters with *more mutual connections*. First, these methods do not explicitly distinguish between *mutual* and *one-way* connections commonly occurring in many social networks, treating them essentially as the same and therefore ignoring the different social relations and interpretations these two types of connections represent (see Section 2 for more details). Second, by simply minimizing the total cross-cluster links (that are symmetrized in some fashion), these methods do not explicitly account for the potential tendencies of node pairs to become mutually connected. As a simple example, Fig. 1 shows two groups of people in a network, where people in the same group tend to have more mutual (stable) connections, and people across two groups have more one-way (unstable) connections. When using the traditional spectral clustering method, as shown in Fig. 1(a), group B will be partitioned into two clusters, due to its strict rule of minimizing the total number of across cluster edges. On the other hand, the correct partition should be done as shown in Fig. 1(b), where the majority of mutual (stable) connections are placed within clusters, and one-way (unstable) connections are placed across clusters.

In this paper, we propose and develop a stable social cluster detection algorithm that takes into account the tendencies of node pairs whether to form mutual (thus stable) connections or not, which can result in more *stable* cluster structures. To tackle this clustering problem, we need to answer the following questions: 1) how to track and evaluate the tendencies of node pairs to become mutual (stable) relations? and 2) how to cluster the entities in social networks by accounting for their mutuality tendencies so as to extract more stable clustering structures?

To address these questions, we utilize dyadic methods to analyze social networks, and develop a generalized mutuality tendency theory which better captures the tendencies of node pairs that tend to establish mutual connections more frequently than those occur by chance. Using these results, we develop a *mutuality-tendency-aware* spectral clustering algorithm to detect more stable clusters by maximizing the *within-cluster* mutuality tendency and minimizing the *cross-cluster* mutuality tendency. Our contributions are summarized as follows.

◇ Motivated by the social science mutuality tendency theory, we establish a new *cluster-based* mutuality tendency theory. It yields a symmetrized mutuality tendency for each node pair, that measures the strength of social ties in (or across) clusters (Sec 3).
◇ Based on our theory, we develop a *mutuality-tendency-aware* spectral clustering algorithm that partitions the social graphs into stable clusters, by maximizing within-cluster mutuality tendencies and minimizing across-cluster mutuality tendencies (Sec 4).
◇ The experimental results – based on both social network structures of synthetical and real social network datasets – confirm that our clustering algorithm generates more stable clusters than the traditional spectral clustering algorithms (Sec 5).

2 Preliminaries, Related Work and Problem Definition

In this section, we first introduce the existing dyadic analysis methods in the social theory literature for analyzing and characterizing social network mutual connections and one-way connections. We then present the classic spectral clustering theory which was developed for *undirected* graphs, and briefly survey some related works which apply this theory to *digraphs* through *symmetrization*. We end the section with the problem definition, namely, how to identify *stable* clusters in social networks by taking into account mutuality tendencies of mutual and one-way connections.

2.1 Dyadic Analysis and Mutuality Tendency

Given a social network with both uni- and bi-directional links, such a network can be represented as a (simple) digraph $G = (V, E)$ with $|V| = n$ nodes. Let A be the standard adjacency matrix of the digraph, where $A_{ij} = 1$ if the directed edge $i \to j$ is present, and $A_{ij} = 0$ otherwise. Social scientists commonly view the social network G as a collection of dyads [28], where *a dyad is an* unordered *pair of nodes and* directed *edges between two nodes in the pair.* Denote a dyad as $Dy_{ij} = (A_{ij}, A_{ji})$, for $i < j$. Since dyad is an unordered notion, we have in total $N_d = n(n-1)/2$ dyads in G. Hence, there are only three possible isomorphism dyads. The first type of dyads is *mutual* relationship, where both directional edges $i \to j$ and $j \to i$ are present. The second type of dyads is *one-way* relationship, where either $i \to j$ or $j \to i$ is present, but not both. The last type of dyads is *null* relationship, where no edges show up between i and j. Let m, b, and u denote the number of mutual, one-way, and null dyads in the network. Clearly, $m + b + u = n(n-1)/2$.

Interpretations of Dyads. Social scientists have observed that mutual social relations and one-way relations in social networks typically exhibit different stabilities, namely, mutual relations are more stable than one-way relations [28]. Hence in the social science literature, one prevalent interpretation of dyadic relations in social networks are the following: mutual dyads are considered as stable connections between two nodes and null relation dyads represent no relations; the one-way dyads [1,5,16,18,20] are viewed as an *intermediate* state of relations, which are in transition to more stable equilibrium states of reciprocity (mutual or no relation). Several recent empirical studies [6,9] of online social networks have further revealed and confirmed that mutual social relations are more stable relations than one-way connections.

Measuring Mutuality Tendency. The notion of mutuality tendency has been introduced in the social science literature (see, e.g., [7,28]) to measure the tendency for a node pair to establish mutual connections. For any dyad between i and j in a digraph G, if i places a link to j, ρ_{ij} represents the tendency that j will reciprocate to i more frequently than would occur by chance. Let $\mathbf{X}_{ij}$ denote the random variable that represents whether or not node i places a directed edge to node j. There are only two possible events (i.e., $\mathbf{X}_{ij}$ takes two possible values): $\mathbf{X}_{ij} = 1$, representing the edge is present; or $\mathbf{X}_{ij} = 0$, the edge is not present. Let X_{ij} (resp. $\bar{X}_{ij}$) denote the event $\{\mathbf{X}_{ij} = 1\}$ (resp. $\{\mathbf{X}_{ij} = 0\}$). Then the probability of the event X_{ij} occurring is $P(X_{ij})$. The probability that i places a directed edge to j and j reciprocates back (i.e., node i and node j are mutually connected) is thus given by $P(X_{ij}, X_{ji}) = P(X_{ij})P(X_{ji}|X_{ij})$. Wofle [28] introduces the following measure of mutuality tendency in terms of the conditional probability $P(X_{ji}|X_{ij})$ as follows:

$$P(X_{ji}|X_{ij}) = P(X_{ji}) + \rho_{ij}P(\bar{X}_{ji}) = \frac{P(X_{ij}, X_{ji})}{P(X_{ij})}, \tag{1}$$

where $-\infty < \rho_{ij} \leq 1$ ensures $0 \leq P(X_{ji}) + \rho P(\bar{X}_{ji}) \leq 1$ to hold. Like many indices used in statistics, $-\infty < \rho \leq 1$ is dimensionless and easy to interpret, since it uses 0 and 1 as benchmarks, representing no tendency and maximum tendency for reciprocation. From eq.(1), the joint distribution $P(X_{ij}, X_{ji})$ in eq.(1) can be measured by the observed graph, namely, either $P(X_{ij}, X_{ji}) = P^{(\omega)}(X_{ij}, X_{ji}) = 1$, when i and j have mutual connection, or $P(X_{ij}, X_{ji}) = P^{(\omega)}(X_{ij}, X_{ji}) = 0$, otherwise, where the superscript ω indicates that the probability is obtained from the observed graph. On the other hand, the distribution for each individual edge is measured by $P(X_{ij}) = P^{(\mu)}(X_{ij}) = \frac{d_i}{|V|-1}$, where d_i is the out-going degree of node i. $P^{(\mu)}(X_{ij})$ represents the probability of edge $i \to j$ being generated under a random graph model, denoted by the superscript μ, with edges randomly generated while preserving the out-degrees. Hence, the tendency ρ is obtained by implicitly comparing the observed graph with a reference random digraph model.

Limitations of Wolfe's Mutuality Tendency Measure for Stable Social Structure Clustering. Although the node pair in a dyad is unordered (i.e., the two nodes are treated "symmetrically" in terms of dyadic relations), Wolfe's measure of mutual tendency is in fact *asymmetric*. This can be easily seen through the following derivation. By definition,

$$\frac{\rho_{ji}}{\rho_{ij}} = \frac{P(X_{ji})P(\bar{X}_{ij})}{P(X_{ij})P(\bar{X}_{ji})} = \frac{P(X_{ji}) - P(X_{ij})P(X_{ji})}{P(X_{ij}) - P(X_{ij})P(X_{ji})}$$

We see that $\rho_{ij} = \rho_{ji}$ if and only if $P(X_{ij}) = P(X_{ji})$ holds. Hence, given an arbitrary dyad in a social network Wolfe's measure of mutuality tendency of the node pair is asymmetric – in a sense that it is a *node-specific* measure of mutuality tendency. It does not provide a measure of mutuality tendency of the (unordered) *node pair* viewed together. In Section 3, we will introduce a new measure of mutuality tendency that is *symmetric* and captures the tendency of a node pair in a dyadic relation to establish mutual connection. This measure of mutuality tendency can be applied to clusters and

a whole network in a straightforward fashion, and leads us to develop a *mutuality-tendency-aware* spectral clustering algorithm.

2.2 Spectral Clustering Theory and Extensions to Digraphs via Symmetrization

Spectral clustering methods (see, e.g., [15, 22, 26, 27, 29]) are originally developed for clustering data with symmetric relations, namely, data that can be represented as *undirected* graphs, where each relation (edge) between two entities, $A_{ij} = A_{ji}$, represents their similarity. The goal is to partition the graph such that entities within each cluster are more similar to each other than those across clusters. This is done by minimizing the total weight of cross-cluster edges. Especially, [12] provides a systematic study on comparing a wide range of undirected graph based clustering algorithms using real large datasets, which gives a nice guideline of how to select clustering algorithms based on the underlying networks and the targeting objectives.

When relations between entities are *asymmetric*, or the underlying graph is *directed*, spectral clustering cannot be directly applied, as the notion of (semi-)definiteness is only defined for *symmetric* matrices. Several recent studies (see, e.g., [10, 21, 27, 29]) all attempt to circumvent this difficulty by first converting the underlying digraphs to undirected graphs via some form of *symmetrization*, and then apply spectral clustering to the resulting symmetrized (undirected) graphs. For example, the authors in [21] discuss several symmetrization methods, including the symmetrized adjacency matrix $\bar{A} = (A + A^T)/2$, the bibliographic coupling matrix AA^T and the co-citation strength matrix $A^T A$, and so forth. Symmetrization can also be done through a random walk on the underlying graph, where $P = D^{-1}A$ is the probability transition matrix and $D = diag[d_i^{out}]$ is a diagonal matrix of node out-degrees. For example, taking the objective function as the random walk flow circulation matrix $F_\pi = \Pi P$, where Π is the diagonal stationary distribution matrix, we have the symmetrized Laplacian of the circulation matrix as $\bar{\mathcal{L}} = (\tilde{\mathcal{L}} + \tilde{\mathcal{L}}^T)/2$, where $\tilde{\mathcal{L}}$ is the (asymmetric) digraph Laplacian matrix [13]. Then the classical spectral clustering algorithm can then be applied using $\bar{\mathcal{L}}$ which is symmetric and semi-definite. Zhou and et al [27, 29] use this type of symmetrization to perform clustering on digraphs. Moreover, Leicht and Newman [10] propose the digraph modularity matrix $Q = [Q_{ij}]$, which captures the difference between the observed digraph and the hypothetical random graph with edges randomly generated by preserving the in- and out-degrees of nodes, namely, $Q_{ij} = A_{ij} - d_i^{out} d_j^{in}/m$. Then, if the sum of edge modularities in a cluster S is large, nodes in S are well connected, since the edges in S tend to appear with higher probabilities than occur by chance. However, Q by definition is asymmetric, where [10] uses the symmetrized $\bar{Q} = (Q+Q^T)/2$ as objective to perform spectral clustering method. Essentially, the edge modularity captures how an individual edge appears more frequently than that happens by chance, thus the modularity based clustering method tends to group those nodes with more connections than expected together, which like all other clustering methods presented above completely ignores the distinction between mutual and one-way connections.

Problem Definition. In this paper we want to solve the following clustering problem in social networks with bi- and uni-directional links: Given a directed (social) graph where mutual connections represent more stable relations and one-way connections represent intermediate transferring states, *how can we account for mutual tendencies of dyadic*

relations and cluster the entities in such a way that nodes within each cluster have maximized mutuality tendencies to establish mutual connections, while across clusters, nodes have minimized tendencies to establish mutual connections? The clusters (representing social structures or communities) identified and extracted thereof will hence likely be more stable.

3 Cluster-Based Mutuality Tendency Theory

Inspired by Wolfe's study in [28], we propose a new measure of mutuality tendency for dyads that can be generalized to groups of nodes (clusters), and develop a *mutuality tendency theory* for characterizing the strength of social ties within a cluster (network structure) as well as across clusters in an asymmetric social graph. This theory lays the theoretical foundation for the network structure classification and community detection algorithms we will develop in section 4.

Let $\mathbf{X}_{ij}$ denote the random variable that represents whether or not node i places a directed edge to node j. There are only two possible events (i.e., $\mathbf{X}_{ij}$ takes two possible values): $\mathbf{X}_{ij} = 1$, representing the edge is present; or $\mathbf{X}_{ij} = 0$, the edge is not present. Let X_{ij} (resp. $\bar{X}_{ij}$) denote the event $\{\mathbf{X}_{ij} = 1\}$ (resp. $\{\mathbf{X}_{ij} = 0\}$). Given an *observed* (asymmetric) social graph G, to capture the *mutuality tendency* of dyads in this graph, we compare it with a *hypothetical*, *random* (social) graph, denoted as $G^{(\mu)}$, where links (dyadic relations) are generated randomly (i.e., by chance) in such a manner that the (out-)degree d_i of each node i in $G^{(\mu)}$ is the same as that in the observed social graph G. Under this random social graph model, the probability of the event X_{ij} occurring is $P^{(\mu)}(X_{ij}) = \frac{d_i}{|V|-1}$; namely, i places a (directed) link to node j randomly or by chance (the superscript μ indicates the probability distribution of link generations under the random social graph model). The probability that i places a directed edge to j and j reciprocates back (i.e., node i and node j are mutually connected) is thus given by $P^{(\mu)}(X_{ij}, X_{ji}) = P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji}|X_{ij}) = P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji})$, since $\mathbf{X}_{ij}$ and $\mathbf{X}_{ji}$ are independent under the random social graph model. On the observed social graph, denote $P^{(\omega)}(X_{ij}, X_{ji})$ to represent the event whether there is a mutual connection (symmetric link) between node i and node j, i.e., $P^{(\omega)}(X_{ij}, X_{ji}) = 1$, if the dyad Dy_{ij} is a mutual dyad in the *observed* social graph, and $P^{(\omega)}(X_{ij}, X_{ji}) = 0$, otherwise. We define the *mutuality tendency* of dyad Dy_{ij} as follows:

$$\theta_{ij} := P^{(\omega)}(X_{ij}, X_{ji}) - P^{(\mu)}(X_{ij}, X_{ji}) = P^{(\omega)}(X_{ij}, X_{ji}) - P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji}), \tag{2}$$

which captures how the node pair i and j establish a mutual dyad more frequently than would occur by chance.

This definition of mutuality tendency is a symmetric measure for dyad Dy_{ij}, i.e., $\theta_{ij} = \theta_{ji}$. In addition, it is shown that $\theta_{ij} \in [-1, 1]$. We remark that $\theta_{ij} = 0$ indicates that if node i places a directed link to node j, the tendency that node j will reciprocate back to node i is no more likely than would occur by chance; the same holds true if node j places a directed link to node i instead. On the other hand, $\theta_{ij} > 0$ indicates that if node i (resp. node j) places a directed link to node j (resp. node i), node j (resp. node i) will more likely than by chance to reciprocate. In particular, with $\theta_{ij} = 1$, node

j (resp. node i) will almost surely reciprocate. In contrast, $\theta_{ij} < 0$ indicates that if node i (resp. node j) places a directed link to node j (resp. node i), node j (resp. node i) will tend not to reciprocate back to node i (resp. node j). In particular, with $\theta_{ij} = -1$, node j (resp. node i) will almost surely not reciprocate back. Hence θ_{ij} provides a measure of strength of social ties between node i and j: $\theta_{ij} > 0$ suggests that the dyadic relation between node i and j is stronger, having a higher tendency (than by chance) to become mutual; whereas $\theta_{ij} < 0$ suggests that node i and j have weaker social ties, and their dyadic relation is likely to remain asymmetric or eventually disappear.

Mutuality Tendency of Clusters. The mutuality tendency measure for dyads defined in eq.(2) can be easily generalized for an arbitrary cluster (a subgraph) in an observed social graph, $S \subseteq G$. We define the mutuality tendency of a cluster S, Θ_S, as follows:

$$\Theta_S := \sum_{i\sim j; i,j\in S} P^{(\omega)}(X_{ij}, X_{ji}) - \sum_{i\sim j; i,j\in S} P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji}), \tag{3}$$

where the subscript $i \sim j : i, j \in S$ means that the summation accounts for all (unordered) dyads, and i, j are both in S. Denote the second term in eq.(3) as $m_S^{(\mu)}$, and the (out-degree) volume of the cluster S as $d_S := \sum_{i\in S} d_i$. As $P^{(\mu)}(X_{ij}) = d_i/(|V| - 1)$ and $P^{(\mu)}(X_{ji}) = d_j/(|V| - 1)$,

$$m_S^{(\mu)} = \sum_{i\sim j; i,j\in S} \frac{d_i d_j}{(|V| - 1)^2} = \frac{d_S^2 - \sum_{i\in S} d_i^2}{2(|V| - 1)^2}, \tag{4}$$

which represents the expected number of mutual connections among nodes in S *under the random social graph model*. Given the cluster S in the observed social graph G, define $m_S^{(\omega)} := \sum_{i\sim j; i,j\in S} P^{(\omega)}(X_{ij}, X_{ji})$, namely, $m_S^{(\omega)}$ represents the number of (observed) mutual connections among nodes in the cluster S in the observed social graph G. The mutual tendency of cluster S defined in eq.(3) is therefore exactly $\Theta_S = m_S^{(\omega)} - m_S^{(\mu)}$.

Hence Θ_S provides a measure of strength of (likely mutual) social ties among nodes in a cluster: $\Theta_S > 0$ suggests that there are more mutual connections among nodes in S than would occur by chance; whereas $\Theta_S < 0$ suggests that there are fewer mutual connections among nodes in S than would occur by chance. Using Θ_S, we can therefore quantify and detect clusters of nodes (network structures or communities) that have strong social ties. In particular, when $S = G$, Θ_G characterizes the mutuality tendency for the entire digraph G, i.e., $\Theta_G = m_G^{(\omega)} - m_G^{(\mu)} = \sum_{i\sim j} \theta_{ij}$, where $m_G^{(\omega)} := \sum_{i\sim j} P^{(\omega)}(X_{ij}, X_{ji})$ represents the number of (observed) mutual dyads among nodes in the observed social graph G, and

$$m_G^{(\mu)} = \sum_{i\sim j} \frac{d_i d_j}{(|V| - 1)^2} = \frac{d^2 - \sum_{i\in V} d_i^2}{2(|V| - 1)^2}, \tag{5}$$

represents the expected number of mutual dyads among nodes in G *under the random social graph model*. Likewise, given a bipartition $(S, \bar{S})$ of G, we define the cross-cluster mutuality tendency as

$$\Theta_{\partial S} := \sum_{i \in S \sim j \in \bar{S}} (P^{(\omega)}(X_{ij}X_{ji}) - P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji})) \tag{6}$$

Denote the second quantity in eq.(6) as $m_S^{(\mu)}$,

$$m_{\partial S}^{(\mu)} = \sum_{i \in S \sim j \in \bar{S}} \frac{d_i d_j}{(|V|-1)^2} = \frac{d_S d_{\bar{S}}}{(|V|-1)^2} \tag{7}$$

which represents the expected number of mutual connections among nodes across S and $\bar{S}$ under the random social graph model. Define $m_{\partial S}^{(\omega)} := \sum_{i \in S \sim j \in \bar{S}} P^{(\omega)}(X_{ij}, X_{ji})$ representing the number of (observed) mutual connections among nodes across clusters S and $\bar{S}$ in the observed social graph G. The mutuality tendency across cluster S and $\bar{S}$ defined in eq.(6) is therefore exactly $\Theta_{\partial S} = m_{\partial S}^{(\omega)} - m_{\partial S}^{(\mu)}$.

The mutuality tendency theory outlined above accounts for different interpretations and roles mutual and one-way connections represent and play in asymmetric social graphs, with the emphasis in particular on the importance of mutual connections in forming and developing stable social structures/communities with strong social ties. In the next section, we will show how we can apply this mutuality tendency theory for detecting and clustering stable network structures and communities in asymmetric social graphs.

4 Mutuality-Tendency-Aware Spectral Clustering Algorithm

In this section, we establish the basic theory and algorithm for solving the mutuality-tendency-aware clustering problem. Due to the space limitation, some proofs are delegated to the technical report [14].

Without loss of generality, we consider only simple (unweighted) digraphs $G = (V, E)$ (i.e., the adjacency matrix A is a 0-1 matrix). Define the mutual connection matrix $M := \min(A, A^T)$, which expresses all the mutual connections with unit weight 1. In other words, if node i and node j are mutually connected (with bidirectional links), $M_{ij} = M_{ji} = 1$, otherwise, $M_{ij} = M_{ji} = 0$. Hence, we have $M_{ij} = P^{(\omega)}(X_{ij}, X_{ji})$, representing the event whether there is a mutual connection (symmetric link) between node i and node j, i.e., in the dyad Dy_{ij} in the observed social graph. In addition, let δ_{ij} be the Kronecker delta symbol, i.e., $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ otherwise. Then, we define matrix

$$\bar{M} = \frac{dd^T - diag[d^2]}{(|V|-1)^2}$$

with d as the out-going degree vector, where each entry

$$\bar{M}_{ij} = \frac{d_i d_j - \delta_{ij} d_i^2}{(|V|-1)^2} = \begin{cases} \frac{d_i d_j}{(|V|-1)^2} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{8}$$

represents the probability that two nodes i and j independently place two unidirectional links to each other to form a mutual dyad. Hence, $\bar{M}_{ij} = P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji})$ represents the probability of node pair i and j to establish a mutual connection under random

graph model with edges randomly generated by preserving the node out-degrees. We denote $T = M - \bar{M}$ as the mutuality tendency matrix, with each entry

$$T_{ij} = P^{(\omega)}(X_{ij}, X_{ji}) - P^{(\mu)}(X_{ij})P^{(\mu)}(X_{ji}) = \theta_{ij} \tag{9}$$

as the individual dyad mutuality tendency.

Mutuality Tendency Lapacian. T is symmetric and those entries associated with non-mutual dyads are negative, representing less mutuality tendencies to establish mutual connections than those occur by chance. Define the mutuality tendency Laplacian matrix as

$$L_T = D_T - T \tag{10}$$

where $D_T = diag[d_T(i)]$ is the diagonal degree matrix of T, with $d_T(i) = \sum_j T_{ij}$. We have the following theorem presenting several properties of L_T.

Theorem 1. *The mutuality tendency Laplacian matrix L_T as defined in eq.(10) has the following properties*

- *Given a column vector $x \in \mathbb{R}^{|V|}$, the bilinear form $x^T L_T x$ satisfies*

$$x^T L_T x = \sum_{i \sim j} T_{ij}(x_i - x_j)^2. \tag{11}$$

- *L_T is symmetric and in general indefinite. In addition, L_T has one eigenvalue equal to 0, with corresponding eigenvector as $\mathbf{1} = [1, \cdots, 1]^T$.*

Mutuality Tendency Ratio Cut Function. For a digraph $G = (V, E)$, and a partition $V = (S, \bar{S})$ on G, we define the *mutuality tendency ratio cut function* as follows.

$$TRCut(S, \bar{S}) = \Theta_{\partial S} \left(\frac{1}{|S|} + \frac{1}{|\bar{S}|} \right), \tag{12}$$

which represents the overall mutuality tendency across clusters balanced by the "sizes" of the clusters. Then, the clustering problem is formulated as a minimization problem with $K = 2$ clusters. (More general cases with $|V| \geq K > 2$ will be discussed in the next subsection.)

$$\min_S TRCut(S, \bar{S}) \tag{13}$$

Since $\Theta_{\partial S} = \Theta_G - (\Theta_S + \Theta_{\bar{S}})$ holds true, we have

$$TRCut(S, \bar{S}) = (\Theta_G - (\Theta_S + \Theta_{\bar{S}})) \left(\frac{1}{|S|} + \frac{1}{|\bar{S}|} \right).$$

For a given graph G, the graph mutuality tendency Θ_G is a constant, the minimization problem in eq.(13) is equivalent to the following maximization problem:

$$\max_S \left\{ (\Theta_S + \Theta_{\bar{S}} - \Theta_G) \left(\frac{1}{|S|} + \frac{1}{|\bar{S}|} \right) \right\}. \tag{14}$$

Hence, minimizing the cross-cluster mutuality tendency is equivalent to maximize the within-cluster mutuality tendency. Using the results presented in Theorem 1, we prove the following theorem which provides the solution to the above mutuality tendency optimization problem.

Theorem 2. *Given the tendency Laplacian matrix $L_T = D_T - T$, the signs of the eigenvector of L_T corresponding to the smallest non-zero eigenvalue indicate the optimal solution $(S, \bar{S})$ to the optimization problem eq.(13).*

Moreover, the mutuality-tendency-aware spectral clustering can be easily generalized for the case of $K > 2$ (See more details in [14]).

Choice of K. We choose K, i.e., the total number of clusters, using the eigengap heuristic [25]. Theorem 1 shows that L_T has all real eigenvalues. Denote the eigenvalues of L_T in an increasing order, i.e., $\lambda_1 \leq \cdots \leq \lambda_n$, The index of the largest eigengap, namely, $K := \text{argmax}_{2 \leq K \leq n}(g(K))$, where $g(K) = \lambda_K - \lambda_{K-1}$, $K = 2, \cdots, n$, indicates how many clusters there are in the network.

5 Evaluations

In this section, we evaluate the performance of the *mutuality-tendency-aware* spectral clustering method by comparing it with various symmetrization methods based digraph spectral clustering algorithms. We only present the comparison results for the adjacency matrix symmetrization method, with objective matrix as $\bar{A} = (A + A^T)/2$. For other settings, we obtained similar results and omit them here, due to the space limitation. We will 1) first test the performances using synthetic datasets, and then 2) apply our method to real online network datasets, e.g., Slashdot social network, and discover stable clusters with respect to mutual and one-way connections.

Synthetic Datasets. We first consider synthetic datasets designed specifically to test the performance of our mutuality-tendency-aware spectral clustering method. We randomly generate a network with 1200 nodes and $K = 3$ clusters, that contain 500, 400 and 300 nodes, respectively. There are 54675 directional edges, among which 27336 edges are bidirectional and 27339 edges are unidirectional. We are randomly placed 90.02% of the bidirectional edges *in* clusters, and 89.6% of the unidirectional edges *across* clusters. Fig. 2(i)-(iii) show that traditional spectral clustering algorithm detects clusters with 661, 538 and 1 entities, respectively, while our method identify correct clusters (See Fig. 3(i)-(iii)).

Real Social Networks. In the second set of simulations, we applied our *mutuality-tendency-aware* spectral clustering algorithm to several real social network datasets, e.g., Slashdot [23], Epinions [19], and email communication network [11] datasets, and compare with various symmetrization methods based digraph clustering algorithms, such as $A = (A + A^T)/2$, AA^T and $F_\pi = \Pi P$. Here we only show the comparison results with adjacency matrix symmetrization based digraph spectral clustering on Slashdot dataset. All other settings lead to similar results and we omit them here.

Slashdot is a technology-related news website founded in 1997. Users can submit stories and it allows other users to comment on them. In 2002, Slashdot introduced

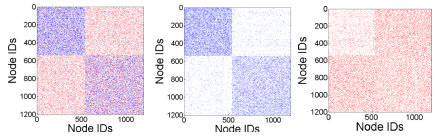


Fig. 2. (i)All edges, (ii)Bidirectional edges, (iii)Unidirectional edges (Traditional spectral clustering method in synthetic dataset)

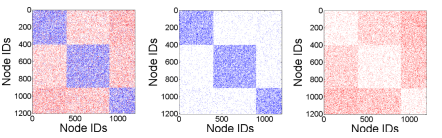


Fig. 3. (i)All edges, (ii)Bidirectional edges, (iii)Unidirectional edges (Tendency aware spectral clustering in synthetic dataset)

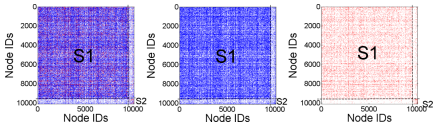


Fig. 4. (i)All edges, (ii)Bidirectional edges, (iii)Unidirectional edges (Traditional spectral clustering method in Slashdot dataset)

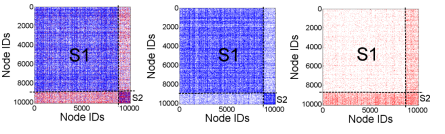


Fig. 5. (i)All edges, (ii)Bidirectional edges, (iii)Unidirectional edges (Tendency aware spectral clustering in Slashdot dataset)

the Slashdot Zoo feature which allows users to tag each other as friends or foes. The network data we used is the Slashdot social relation network, where a directed edge from i to j indicates an interest from i to j's stories (or topics). Hence, two people with mutual connections thus share some common interests, while one-way connections infer that one is interested in the other's posts, but the interests are not reciprocated back. The Slashdot social network data was collected and released by Leskovec [23] in November 2008.

The statistics[1] are shown in Table 1. It shows that the largest strongly connected component (SCC) include about 70355 nodes. Then, we remove those nodes with very low in-degrees and out-degrees, say no more than or equal to 2. By finding the largest strongly connected component of the remaining graph, we extract a "core" of the network with 10131 nodes and 197378 edges, among which there are 21404 unidirectional edges and 175974 bidirectional edges, respectively. In our evaluations, we observe that there is a large "core" of the network, and all other users are attached to this core network. In our study, we are interested in extracting the community structure from the "core" network. When applying our spectral clustering algorithm to the "core" network, two clusters with 8892 and 1239 nodes are detected (shown in Fig.5(i)-(iii)). In our result, a large portion (about 35.04%) of cross-cluster edges are unidirectional edges which in turn yield lower mutuality tendency across clusters. On the other hand, when using the traditional symmetrized $\bar{A} = (A + A^T)/2$, two clusters with 9640 and 491 nodes are extracted instead (shown in Fig.4(i)-(iii)). We can see that the clustering result obtained using the traditional spectral clustering method has only around 5.75% of the total edges across clusters as unidirectional edges, which boost up the mutuality tendency across clusters. However, in our clustering result, we have more unidirectional edges placed across clusters, which decreases the mutuality tendency

[1] Here, the total number of edges is smaller than that is shown on the website [23], because we do not count for those selfloops.

Table 1. Statistics of Slashdot Dataset (U-edge: Unidirectional edge, B-edge: Bidirectional edge)

Nodes	77360	Nodes in largest SCC	70355	Nodes in the "core" component	10131
Edges	828161	Edges in largest SCC	818310	Edges in the "core" component	197378
U-edges	110199	U-edges in largest SCC	100930	U-edges in the "core" component	21404
B-edges	717962	B-edges in largest SCC	717380	B-edges in the "core" component	175974

Table 2. Ave. mutuality tendency comparison on Slashdot dataset

.	θ_G	θ_{S1}	θ_{S2}	$\theta_{\partial S}$
Mutuality tendency aware clustering	0.0017	**0.0049**	**0.0028**	**0.00033**
Traditional clustering	0.0017	0.0018	0.0021	0.00070

across clusters. From Fig. 5(i), we can clearly see that we have unidirectional (red) edges dominating the cross-cluster parts. Moreover, Table 2 shows the average mutuality tendency comparison between different clustering methods, where we can see that the mutuality-tendency-aware spectral clustering algorithm can group nodes together with higher within-cluster tendencies than that of traditional spectral clustering.

References

1. Berscheid, E., Regan, P.: The psychology of interpersonal relationships. Pearson Prentice Hall (2005)
2. DeScioli, P., Kurzban, R., Koch, E., Liben-Nowell, D.: Best friends: Alliances, friend ranking, and the myspace social network. Perspectives on Psychological Science 6(1), 6–8 (2011)
3. Golder, S., Yardi, S., Marwick, A.: A structural approach to contact recommendations in online social networks. In: Workshop on Search in Social Media, SSM (2009)
4. Gouldner, A.: The norm of reciprocity: A preliminary statement. American Sociological Review, 161–178 (1960)
5. Heider, F.: Attitudes and cognitive organization. Journal of Psychology 21(1), 107–112 (1946)
6. Jamali, M., Haffari, G., Ester, M.: Modeling the temporal dynamics of social rating networks using bidirectional effects of social relations and rating patterns. In: WWW (2011)
7. Katz, L., Powell, J.: Measurement of the tendency toward reciprocation of choice. Sociometry 18(4), 403–409 (1955)
8. Kurucz, M., Benczur, A., Csalogany, K., Lukacs, L.: Spectral clustering in telephone call graphs. In: WebKDD (2007)
9. Kwak, H., Chun, H., Moon, S.: Fragile online relationship: A first look at unfollow dynamics in twitter. In: CHI (2011)
10. Leicht, E., Newman, M.: Community structure in directed networks. Physical Review Letters 100(11), 118703 (2008)
11. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data (TKDD) 1(1), 1–41 (2007)
12. Leskovec, J., Lang, K., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: WWW (2010)
13. Li, Y., Zhang, Z.-L.: Random Walks on Digraphs, the Generalized Digraph Laplacian and the Degree of Asymmetry. In: Kumar, R., Sivakumar, D. (eds.) WAW 2010. LNCS, vol. 6516, pp. 74–85. Springer, Heidelberg (2010)

14. Li, Y., Zhang, Z.-L., Bao, J.: Mutual or unrequited love: Identifying stable clusters in social networks with uni- and bi-directional links. Arxiv preprint arXiv:1203.5474 (March 2012)
15. Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17, 395–416 (2007)
16. Miller, H., Geller, D.: Structural balance in dyads. Journal of Personality and Social Psychology 21(2), 135 (1972)
17. Mishra, N., Schreiber, R., Stanton, I., Tarjan, R.E.: Clustering Social Networks. In: Bonato, A., Chung, F.R.K. (eds.) WAW 2007. LNCS, vol. 4863, pp. 56–67. Springer, Heidelberg (2007)
18. Price, K., Harburg, E., Newcomb, T.: Psychological balance in situations of negative interpersonal attitudes. Journal of Personality and Social Psychology 3(3), 265 (1966)
19. Richardson, M., Agrawal, R., Domingos, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 351–368. Springer, Heidelberg (2003)
20. Roudrigues, A.: Effects of balance, positivity, and agreement in triadic social relations. Journal of Personality and Social Psychology 5(4), 472 (1967)
21. Satuluri, V., Parthasarathy, S.: Symmetrizations for clustering directed graphs. In: EDBT/ICDT (2011)
22. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 888–905 (2000)
23. Slashdot. dataset, http://snap.stanford.edu/data/soc-Slashdot0811.html
24. Smyth, S.: A spectral clustering approach to finding communities in graphs. In: SDM (2005)
25. von Luxburg, U.: A tutorial on spectral clustering. Technical Report No.TR-149, Max Planck Institute for Biological Cybernetics (2006)
26. Wang, X., Davidson, I.: Flexible constrained spectral clustering. In: KDD 2010, pp. 563–572 (2010)
27. Weston, J., Leslie, C.S., Ie, E., Zhou, D., Elisseeff, A., Noble, W.S.: Semi-supervised protein classification using cluster kernels. Bioinformatics 21(15), 3241–3247 (2005)
28. Wolfe, A.: Social network analysis: Methods and applications. American Ethnologist 24(1), 219–220 (1997)
29. Zhou, D., Huang, J., Schölkopf, B.: Learning from labeled and unlabeled data on a directed graph. In: ICML (2005)

Dynamic PageRank Using Evolving Teleportation

Ryan A. Rossi and David F. Gleich

Purdue University
Department of Computer Science
305 N. University St., West Lafayette, IN 47906
{rrossi,dgleich}@purdue.edu

Abstract. The importance of nodes in a network constantly fluctuates based on changes in the network structure as well as changes in external interest. We propose an evolving teleportation adaptation of the PageRank method to capture how changes in external interest influence the importance of a node. This framework seamlessly generalizes PageRank because the importance of a node will converge to the PageRank values if the external influence stops changing. We demonstrate the effectiveness of the evolving teleportation on the Wikipedia graph and the Twitter social network. The external interest is given by the number of hourly visitors to each page and the number of monthly tweets for each user.

1 Introduction

Finding important nodes in a graph is a key task in a variety of applications: search engines [24,18], network science [17,8,14], and bioinformatics [27,22], among many others. By and large, these are global measures of node importance and one of the most well-studied measures is PageRank [24,20].

PageRank computes the importance of each node in a directed graph under a random surfer model. When at a node, the random surfer can either:

1. transition to a new node from the set of out-edges, or
2. do something else (e.g., execute a search query, use a bookmark).

The probability that the surfer performs the first action is known as the damping parameter in PageRank. We use α to denote the damping parameter. The second action is called teleporting and is modeled by the surfer picking a node at random according to a distribution called the teleportation distribution vector or personalization vector. These choices only depend on the current node and, consequently, define a Markov chain. This PageRank Markov chain always has a unique stationary distribution for any $0 \leq \alpha < 1$. The importance of a node is proportional to its stationary distribution in this Markov chain. Thus, the computation is governed by the graph, a teleportation parameter α, and a teleportation distribution vector.

The PageRank score is a simple model for the importance of a node in a graph, and there are many variations that may yield more useful scores (for instance [21] models a random walk with a back button). A common complaint

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 126–137, 2012.

about PageRank models is that they are only defined for static graphs. Motivated by the idea of studying PageRank with dynamic graphs, we formulate a dynamic PageRank model for a static graph with a time-dependent, or evolving, teleportation vector. Intuitively, the teleportation distribution changes based on human dynamics such as recent news and seasonal preferences. For example, in our forthcoming experiments (Section 6), the time-dependent vector is the number of hourly page visits for each page from Wikipedia. We derive the model and algorithms for this dynamic version of PageRank in Section 4. The resulting algorithms scale to large graphs. Moreover, we show that the new model is a generalization of PageRank in the sense that *if the time-dependent vector stops changing* then *our dynamic score vector converges to the standard PageRank score.*

We make our code and data available in the spirit of reproducible research:
`http://www.cs.purdue.edu/homes/dgleich/codes/dynsyspr-waw`

2 PageRank Notation

In order to place our work in context, we first introduce some notation. Let $\mathbf{A}$ be the adjacency matrix for a graph where $A_{i,j}$ denotes an edges from node i to node j. In order to avoid a proliferation of transposes, we define $\mathbf{P}$ as the transposed transition matrix for a random-walk on a graph:

$$P_{j,i} = \text{ probability of transitioning from node } i \text{ to node } j.$$

Hence, the matrix $\mathbf{P}$ is *column-stochastic* instead of row-stochastic, which is the standard in probability theory. Throughout this manuscript, we utilize uniform random-walks on a graph, in which case $\mathbf{P} = \mathbf{A}^T\mathbf{D}^{-1}$ where $\mathbf{D}$ is a diagonal matrix with the degree of each node on the diagonal. However, none of the theory is restricted to this type of random walk and any column-stochastic matrix will do. The PageRank vector $\mathbf{x}$ is the solution of the linear system:

$$(\mathbf{I} - \alpha\mathbf{P})\mathbf{x} = (1-\alpha)\mathbf{v}$$

for any $0 \le \alpha < 1$ and any teleportation distribution vector $\mathbf{v}$ such that $v_i \ge 0$ and $\sum v_i = 1$. Table 1 summarizes these notation conventions, and has a few other elements that will be discussed in the forthcoming sections.

3 Dynamic and Evolving Rankings

The PageRank literature is vast, and we now survey some of the other ideas related to incorporating graph dynamics into a PageRank vector, more general models for studying dynamic graphs, and updating PageRank vectors.

Our proposed method is related to changing the teleportation vector in the power method as its being computed. Bianchini et al. [5] noted that the power method would still converge if either the graph or the vector $\mathbf{v}$ changed during the method, albeit to a new solution given by the new vector or graph.

Table 1. Summary of notation. Matrices are bold, upright roman letters; vectors are bold, lowercase roman letters; and scalars are unbolded roman or greek letters.

n	number of nodes in a graph
$\mathbf{e}$	the vector of all ones
$\mathbf{P}$	column stochastic matrix
α	damping parameter in PageRank
$\mathbf{v}$	teleportation distribution vector
$\mathbf{x}$	solution to the PageRank computation
$\mathbf{v}(t)$	a teleportation distribution vector at time t
$\mathbf{x}(t)$	solution to the Dynamic PageRank computation for time t
θ	decay parameter for time-series smoothing

Our method capitalizes on a closely related idea and we utilize the intermediate quantities explicitly. Another related idea is the Online Page Importance Computation (OPIC) [1], which integrates a PageRank-like computation *with* a crawling process. The method does nothing special if a node has changed when it is crawled again. A more detailed study of how PageRank values evolve during a web-crawl was done by Boldi et al. [7]. Other work has approximated PageRank on graph streams [11].

Outside of the context of web-ranking, O'Madadhain and Smyth propose EventRank [23], a method of ranking nodes in dynamic graphs, that uses the PageRank propagation equations for a sequence of graphs. We utilize the same idea but place it within the context of a dynamical system.

While we described PageRank in terms of a random-surfer model above, another characterization of PageRank is that it is a sum of damped transitions:

$$\mathbf{x} = (1-\alpha)\sum_{k=0}^{\infty}(\alpha\mathbf{P})^k\mathbf{v}.$$

These transitions are a type of probabilistic walk and Grindrod et al. [16] introduced the related notion of dynamic walks for dynamic graphs.

In the context of popularity dynamics [25], our method captures how changes in external interest influence the popularity of nodes and the nodes linked to these nodes in an implicit fashion. Our work is also related to modeling human dynamics, namely, how humans change their behavior when exposed to rapidly changing or unfamiliar conditions [3]. In one instance, our method shows the important topics and ideas relevant to humans before and after one of the largest Australian Earthquakes.

In closing, we wish to note that our proposed method *does not involve* updating the PageRank vector, a related problem which has received considerable attention [9,19]. Nor is it related to tensor methods for dynamic graph data [26,12].

4 PageRank with Dynamic Teleportation

In order to incorporate dynamics into PageRank, we reformulate a standard PageRank algorithm in terms of changes to the PageRank values for each page. This step allows us to state PageRank as a dynamical system, in which case we can easily incorporate changes into the vector.

The standard PageRank algorithm is the classical Richardson iteration:

$$\mathbf{x}^{(k+1)} = \alpha \mathbf{P}\mathbf{x}^{(k)} + (1-\alpha)\mathbf{v}.$$

(Note that this iteration is identical to the power method for the PageRank Markov chain.) By rearranging this equation into a difference form, we have

$$\Delta\mathbf{x}^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} = \alpha \mathbf{P}\mathbf{x}^{(k)} + (1-\alpha)\mathbf{v} - \mathbf{x}^{(k)} = (1-\alpha)\mathbf{v} - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}^{(k)}.$$

Thus, changes in the PageRank values at a node *evolve* based on the value $(1-\alpha)\mathbf{v}-(\mathbf{I}-\alpha\mathbf{P})\mathbf{x}^{(k)}$. We reinterpret this update as a continuous time dynamical system:

$$\mathbf{x}'(t) = (1-\alpha)\mathbf{v} - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}(t). \tag{1}$$

Other iterative methods also give rise to related dynamical systems, as utilized by [13] for studying eigenvalue solvers.

In the dynamic teleportation model, $\mathbf{v}$ is no longer fixed, but is instead a function of time $\mathbf{v}(t)$:

$$\mathbf{x}'(t) = (1-\alpha)\mathbf{v}(t) - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}(t). \tag{2}$$

Note that this means the PageRank values $\mathbf{x}(t)$ may not "settle" or converge. We see this as a feature of the new model as we plan to utilize information from the evolution and changes in the PageRank values.

Standard texts on dynamical system show that the solution $\mathbf{x}(t)$ is:

$$\mathbf{x}(t) = \exp[-(\mathbf{I} - \alpha\mathbf{P})t]\mathbf{x}(0) + (1-\alpha)\int_0^t \exp[-(\mathbf{I} - \alpha\mathbf{P})(t-\tau)]\mathbf{v}(\tau)\, d\tau.$$

If $\mathbf{v}(t) = \mathbf{v}$ is constant with respect to time, then

$$\int_0^t \exp[-(\mathbf{I} - \alpha\mathbf{P})(t-\tau)]\mathbf{v}(\tau)\, d\tau = (\mathbf{I} - \alpha\mathbf{P})^{-1}\mathbf{v} - \exp[-(\mathbf{I} - \alpha\mathbf{P})t](\mathbf{I} - \alpha\mathbf{P})^{-1}\mathbf{v}.$$

Hence, for constant $\mathbf{v}(t)$:

$$\mathbf{x}(t) = \exp[-(\mathbf{I} - \alpha\mathbf{P})t](\mathbf{x}(0) - \mathbf{x}) + \mathbf{x},$$

where $\mathbf{x}$ is the solution to static PageRank: $(\mathbf{I} - \alpha\mathbf{P})\mathbf{x} = (1-\alpha)\mathbf{v}$. Because all the eigenvalues of $-(\mathbf{I} - \alpha\mathbf{P}) < 0$, the matrix exponential terms disappear in a sufficiently long time horizon. Thus, when $\mathbf{v}(t) = \mathbf{v}$, nothing has changed. We recover the original PageRank vector $\mathbf{x}$ as the steady-state solution:

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x} \text{ the PageRank vector.}$$

This derivation shows that dynamic teleportation PageRank is a generalization of the PageRank vector.

Require:
a graph $G = (V, E)$ and a procedure to compute $\mathbf{Px}$ for this graph
a maximum time $t_{\max}$
a function to return $\mathbf{v}(t)$ for any $0 \le t \le t_{\max}$
a damping parameter α
a time-step h
Ensure: $\mathbf{X}$ where the kth column of $\mathbf{X}$ is $\mathbf{x}(0 + kh)$ for all $1 \le k \le t_{\max}/h$ (or any desired subset of these values)

$t \leftarrow 0$; $k = 1$
$\mathbf{x}(0) \leftarrow \mathbf{v}(0)$ (or any other desired initial condition)
while $t \le t_{\max} - h$ **do**
  $\mathbf{x}(t + h) \leftarrow \mathbf{x}(t) + h\left[(1 - \alpha)\mathbf{v}(t) - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}(t)\right]$
  $\mathbf{X}(:, k) \leftarrow \mathbf{x}(t + h)$
  $t \leftarrow t + h$; $k \leftarrow k + 1$
end while

Fig. 1. In order to compute a sequence of dynamic teleportation PageRank values, we utilize a forward Euler method for the dynamical system: $\mathbf{x}'(t) = (1 - \alpha)\mathbf{v}(t) - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}(t)$. The resulting procedure looks remarkably similar to the standard Richardson iteration to compute a PageRank vector. A key difference is that there is no notion of convergence.

4.1 Algorithms

In order to compute the time-sequence of PageRank values $\mathbf{x}(t)$, we can evolve the dynamical system (1) using any standard method, for instance a forward Euler or a Runge-Kutta method. At the moment, we only use the forward Euler method for simplicity. This method lacks high accuracy, but is fast and straightforward. Forward Euler approximates the derivative with a first order Taylor approximation:

$$\mathbf{x}'(t) \approx \frac{\mathbf{x}(t + h) - \mathbf{x}(t)}{h},$$

and then uses that approximation to estimate the value at a short time-step in the future:

$$\mathbf{x}(t + h) = \mathbf{x}(t) + h\left[(1 - \alpha)\mathbf{v}(t) - (\mathbf{I} - \alpha\mathbf{P})\mathbf{x}(t)\right].$$

Note that if $h = 1$ and $\mathbf{v}(t) = \mathbf{v}$ for all t, then this update becomes the original Richardson iteration. A summary of this derivation as a formal algorithm to compute a dynamic teleportation PageRank time series is given by Figure 1.

4.2 Discussion of the Algorithm & Practical Issues

First, the algorithm we propose easily scales to large networks. This isn't surprising given its close relationship to the Richardson method for PageRank. The major expense is the set of $t_{\max}/h$ matrix-vector products with $\mathbf{P}$ – all of the other work is linear in the number of nodes. It could also be used in a distributed setting if any distributed matrix-vector product is available.

In one sense, the forward Euler method is simply running a power method, but changing the vector $\mathbf{v}$ at every iteration. However, we derived this method based on evolving (2). Thus, by studying the relationship between (2) and the algorithm in Figure 1, we can understand the underlying problem solved by changing the teleportation vector while running the power method. Consequently, we gain additional flexibility in adapting (2) to problems.

Thus far, we also have not discussed how to set $\mathbf{v}(t)$ beyond the brief allusion at the beginning that the dynamic teleportation will be based on Wikipedia pageviews. When we apply the dynamic teleportation PageRank model, we need to pick a relationship between the time-scale of the dynamical system (2) and the time-scale in the underlying application. For instance, does $\mathbf{x}(1)$ correspond to the PageRank values after a second, an hour, a day? There is no "correct" answer and the relationship has implications on the final model.

Suppose that we set $\alpha = 0.85$, $h = 1$, and that $t = 1$ is a minute of time in the application. If we have hourly data on Wikipedia pageviews, then the above algorithm will compute 60 iterations of the power-method between each hour. If we further use the incredibly simple model that $\mathbf{v}(t)$ changes each hour as we get new data, then the forward Euler method is essentially equivalent to running the power-method to convergence after $\mathbf{v}$ changes on the hour. (They are essentially equivalent in the sense that PageRank will have converged to a 1-norm error of 10^{-4} in about 60 iterations.) If, instead, we set $\alpha = 0.85$, $h = 1$, and $t = 1$ to be 20 minutes of time in the application, then we will do 3 iterations of the power method after each hourly change.

In the preceding discussion of the algorithm, we hypothesized that $\mathbf{v}(t)$ changes at fixed intervals based on incoming data. A better idea is to smooth out these "jumps" using an exponentially weighted moving average. We plan to investigate this in the future.

4.3 Ranking from Time-Series

The above equations provide a time-series of dynamic PageRank vectors for the nodes, denoted formally as $\mathbf{x}(t), 0 \leq t \leq t_{\max}$. Most applications, however, want a single score, or small set of scores, to characterize the importance of a node. We now discuss a few ways in which these time series give rise to scores. Reference [23] used similar ideas to extract a single score from a time-series.

Transient Rank. We call the instantaneous values of $\mathbf{x}(t)$ a node's *transient* rank. This score gives the importance of a node at a particular time.

Summary & Cumulative Rank. Any summary function s of the time series, such as the integral, average, minimum, maximum, variance, is a single score that encompasses the entire interval $[0, t_{\max}]$. We utilize the *cumulative rank* in the forthcoming experiments:

$$\mathbf{c} = \int_0^{t_{\max}} \mathbf{x}(t)\, dt \approx h\mathbf{X}\mathbf{e}.$$

Difference Rank. A node's difference rank is the difference between its maximum and minimum rank over all time:

$$\mathbf{d} = \max_t[\mathbf{x}(t)] - \min_t[\mathbf{x}(t)].$$

Nodes with high difference rank should reflect important events that occurred within the range $[0, t_{\max}]$. The underlying intuition is that normal nodes are the pages where the Dynamic PageRanks do not change much. While the pages that have large differences in their time-series of PageRanks are topics or news that went viral or becomes popular over time. See Section 6 for more details and Figure 3 for examples such as Rihanna, PricewaterhouseCoopers, Watchmen, and American Idol (season 8).

Having a variety of different scores derived from the same data frequently helps when using these scores as features in a prediction or learning task [4,10].

4.4 Clustering the Time-Series

After applying our forward Euler based algorithm, we have sampled an approximation of this time-series: $\mathbf{X} = \{\mathbf{x}(hk) : k = 1, \ldots, t_{\max}/h\}$. By *clustering* these discrete time-series, we can automatically discover patterns such as increasing or decreasing trends, periodic bursts at certain times of the year, and their ilk. Our initial experiments were promising but were omitted due to space.

5 Datasets

In both of the following experiments, we set $h = 1$, and $t = 5$ to represent one period of data – one hour for Wikipedia and one month for Twitter – so that we do 5 iterations of the forward Euler method before incorporating the new data. In each period $\mathbf{v}(t)$ is normalized to sum to 1, but is otherwise unchanged.

Wikipedia Article Graph and Hourly Pageviews. Wikipedia provides access to copies of its database [28]. We downloaded a copy of its database on March 6th, 2009 and extracted an article-by-article link graph, where an article is a page in the main Wikipedia namespace, a category page, or a portal page. All other pages and links were removed. See [15] for more information.

Wikipedia also provides hourly pageviews for each page [29]. These are the number of times a page was viewed for a given hour. These are not unique visits. We downloaded the raw page counts and matched the corresponding page counts to the pages in the Wikipedia graph. We used the page counts starting from March 6, 2009 and moving forward in time.

As an aside, let us note that vertex degrees and cumulated pageviews are uncorrelated with a correlation coefficient of 0.02, indicating that using pageviews will not reinforce any degree bias in the dynamic ranks. In fact, pages with a large number of pageviews may not have high in-degree at all, which provides evidence that pages with large in-degree are not always visited more frequently.

Twitter Social Network and Monthly Tweet Rates. We use a follower graph generated by starting with a few seed users and crawling follows links from 2008. We extract the user tweets over time from 2008 – 2009. A tweet is represented as a tuple ⟨user, time, tweet⟩. Using the set of tweets, we construct a sequence of vectors to represents the number of tweets for a given month.

Table 2. Dataset Properties. The pageviews or tweets is denoted as $\mathbf{p}$.

Dataset	Nodes	Edges	$t_{\max}$	Period	Average p_i	Max p_i
WIKIPEDIA	4,143,840	72,718,664	20	hours	1.3225	334,650
TWITTER	465,022	835,424	6	months	0.5569	1056

6 Empirical Results

In this section, we demonstrate the effectiveness of Dynamic PageRank as a method for automatically adapting page importance based on graph structure and external influence by showing that it provides different insights (§6.1), finds interesting pages (§6.2), and helps predict pageviews (§6.3).

6.1 Ranking from Time-Series

We first use the intersection similarity measure to evaluate the rankings [6]. Given two vectors $\mathbf{x}$ and $\mathbf{y}$, the intersection similarity metric at k is the average symmetric difference over the top-j sets for each $j \leq k$. If $\mathcal{X}_k$ and $\mathcal{Y}_k$ are the top-k sets for $\mathbf{x}$ and $\mathbf{y}$, then $\text{isim}_k(\mathbf{x}, \mathbf{y}) = \frac{1}{k} \sum_{j=1}^{k} \frac{|\mathcal{X}_j \Delta \mathcal{Y}_j|}{2j}$, where Δ is the symmetric set-difference operation. Identical vectors have an intersection similarity of 0.

For the Wikipedia graph, Figure 2 shows the similarity profile comparing $\mathbf{d}$ (from §4.3) to static PageRank, degree, cumulative pageviews $\mathbf{p}_c$, maximum pageviews difference $\mathbf{p}_d$, and two other Dynamic PageRank vectors: transient $\mathbf{x}(t_{\max})$ and cumulative $\mathbf{c}$. The figure suggests that Dynamic PageRank is different from the other measures, even for small values of k. In particular, combining the external influence with the graph appears to produce something new.

6.2 Top Dynamic Ranks

Figure 3 shows the time-series of the top 100 pages by the difference measure. Many of these pages reveal the ability of Dynamic PageRank to mesh the network structure with changes in external interest. This became immediately clear after reviewing significant events from this time period. We find pages related to an Australian earthquake (40, 72, 70), a just released movie "Watchmen" (94, 39, 99), a famous musician that died (2, 95, 68), recent "American Idol" gossip (32, 96, 56), a remembrance of Eve Carson from a contestant on "American Idol" (80, 88, 27), news about the murder of a Harry Potter actor (77), and the

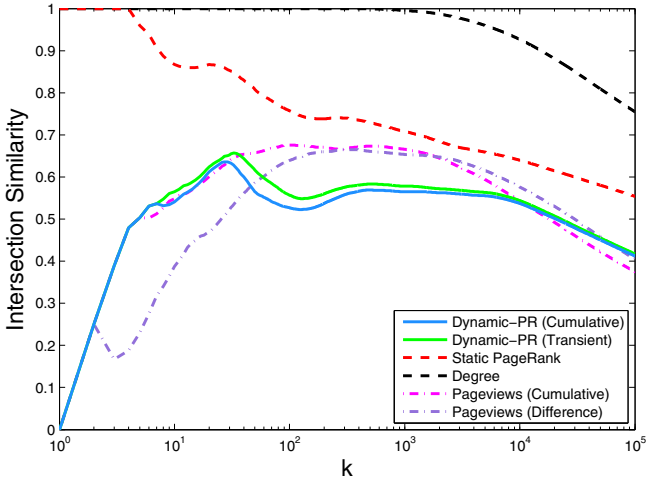


Fig. 2. Intersection similarity between Dynamic PageRank's difference ranking $\mathbf{d}$ and the other ranking vectors. To more appropriately see the differences, we zoom in on the top 10^5 nodes. See the discussion in the text.

Skittles social media mishap (87). These results demonstrate the effectiveness of the Dynamic PageRank to identify interesting pages that pertain to external interest. The influence of the graph results in the promotion of pages such as Richter magnitude (72). That page was not in the top 200 from pageviews.

In another study, omitted due to space, we performed a clustering of these time-series to identify pages with similar trends. For instance, pages such as Watchmen (37) and Rorschach (94) share strikingly similar patterns. These patterns indicate the page that became important first and the amount of traffic or popularity that diffused over time.

6.3 Predicting Future Pageviews & Tweets

We conclude by studying how well the dynamic PageRank values *predict* future pageviews. Formally, given a lagged time-series [2], the goal is to predict the future value $\mathbf{p}_{t+1}$ (actual pageviews or number of tweets). This type of temporal prediction task has many applications, such as actively adapting caches in large database systems, or dynamically recommending pages.

We performed one-step ahead predictions $(t+1)$ using linear regression. That is, we learn a model of the form:

$$\left[\bar{\mathbf{f}}(t-1;\theta)\ \bar{\mathbf{f}}(t-2;\theta)\ \ldots\ \bar{\mathbf{f}}(t-w;\theta)\right]\ \mathbf{b}\ \approx\ \mathbf{p}(t)$$

where w is the window-size, and $\bar{\mathbf{f}}(\cdot;\theta)$ is an exponentially damped moving average computed from either pageviews, dynamic PageRanks, or both. Using this

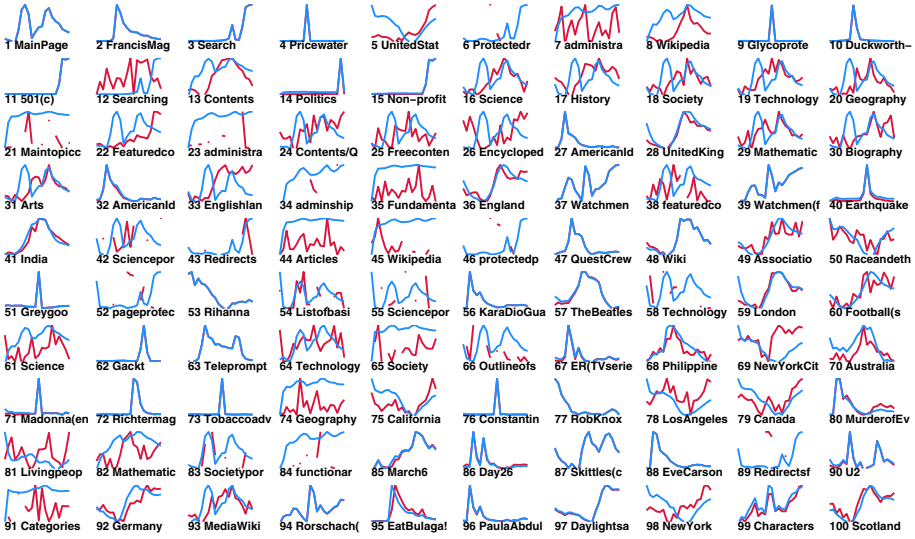


Fig. 3. The top-100 Wikipedia pages that fluctuate the most as determined by the difference ranking from our Dynamic PageRank approach. The x-axis represents time (in hours) while the y-axis represents the Dynamic PageRank value. The blue line represents Dynamic PageRank and the red line represents the hourly pageviews. There exist many interesting time-series patterns such as spikes (40), cyclic/seasonality trends (16-20), and increasing/decreasing trends (39 and 77), among many others. Further analysis and anecdotal evidence was removed due to space.

average is a standard forecasting technique. Specifically, the exponentially damped moving average of a time-series feature $\mathbf{f}(t)$ is:

$$\bar{\mathbf{f}}(t;\theta) = \underbrace{\theta \mathbf{f}(t)}_{\text{new data}} + \underbrace{(1-\theta)\bar{\mathbf{f}}(t-1;\theta)}_{\text{old data}}.$$

The exponential factor was $\theta = 0.3$ for Twitter and $\theta = 0.7$ for Wikipedia. Due to the scarcity of the data, we used 0.3 for Twitter since this choice weights past observations more heavily. In the future, we plan to use cross-validation. After fitting, the model predicts $\mathbf{p}(t+1)$ as $\left[\bar{\mathbf{f}}(t;\theta)\ \bar{\mathbf{f}}(t-1;\theta) \cdots \bar{\mathbf{f}}(t-w+1;\theta)\right]\ \mathbf{b}$. To measure the error, we use symmetric Mean Absolute Percentage Error (or sMAPE) [2].

We study two models.

Base Model. This model uses only the time-series of pageviews or tweet-rates to predict the future pageviews or number of tweets.

Dynamic PageRank Model. This model uses both the Dynamic PageRank time-series and pageviews to predict the future pageviews.

We evaluate these models for prediction on *stationary* and *non-stationary* time-series. Informally, a time-series is weakly stationary if it has properties

(mean and covariance) similar to that of the time-shifted time-series. We consider the top and bottom 1000 nodes from the difference ranking as nodes that are approximately non-stationary (volatile) and stationary (stable), respectively. Table 3 compares the predictions of the models across time for non-stationary and stationary prediction tasks. Our findings indicate that the Dynamic PageRank time-series provides valuable information for forecasting future pageviews.

Table 3. Average SMAPE over all nodes for the two models (lower is better). We also measure the performance of the models for predicting highly volatile nodes (non-stationary) and nodes with relatively stable behavior (stationary). In all cases, the Dynamic PageRank model is more accurate than the base model.

Dataset	Forecasting	Dynamic PageRank	Base Model
WIKIPEDIA	*Non-stationary*	**0.4349**	0.5028
	Stationary	**0.3672**	0.4373
TWITTER	*Non-stationary*	**0.4852**	1.2333
	Stationary	**0.6690**	0.9180

7 Conclusion

We proposed an evolving teleportation adaptation of the PageRank method to capture how changes in external interest influence the importance of a node. This proposal lets us treat PageRank as a dynamical system and seamlessly incorporate changes in the teleportation vector. Furthermore, we demonstrated the utility of using Dynamic PageRank for predicting pageviews. In future work, we hope to include dynamic and evolving graphs into this framework as well.

References

1. Abiteboul, S., Preda, M., Cobena, G.: Adaptive on-line page importance computation. In: WWW, pp. 280–290. ACM (2003)
2. Ahmed, N., Atiya, A., El Gayar, N., El-Shishiny, H.: An empirical comparison of machine learning models for time series forecasting. Econ. Rev. 29(5-6), 594–621 (2010)
3. Bagrow, J., Wang, D., Barabási, A.: Collective response of human populations to large-scale emergencies. PloS one 6(3), e17680 (2011)
4. Becchetti, L., Castillo, C., Donato, D., Baeza-Yates, R., Leonardi, S.: Link analysis for web spam detection. ACM Trans. Web 2(1), 1–42 (2008)
5. Bianchini, M., Gori, M., Scarselli, F.: Inside PageRank. ACM Transactions on Internet Technologies 5(1), 92–128 (2005)
6. Boldi, P.: TotalRank: Ranking without damping. In: WWW, pp. 898–899 (2005)
7. Boldi, P., Santini, M., Vigna, S.: Paradoxical effects in PageRank incremental computations. Internet Mathematics 2(2), 387–404 (2005)
8. Bonacich, P.: Power and centrality: A family of measures. American Journal of Sociology, 1170–1182 (1987)

9. Chien, S., Dwork, C., Kumar, R., Simon, D., Sivakumar, D.: Link evolution: Analysis and algorithms. Internet Mathematics 1(3), 277–304 (2004)
10. Constantine, P., Gleich, D.: Random alpha PageRank. Internet Mathematics 6(2), 189–236 (2009)
11. Das Sarma, A., Gollapudi, S., Panigrahy, R.: Estimating PageRank on graph streams. In: SIGMOD, pp. 69–78. ACM (2008)
12. Dunlavy, D.M., Kolda, T.G., Acar, E.: Temporal link prediction using matrix and tensor factorizations. TKDD 5(2), 10:1–10:27 (2011)
13. Embree, M., Lehoucq, R.B.: Dynamical systems and non-hermitian iterative eigensolvers. SIAM Journal on Numerical Analysis 47(2), 1445–1473 (2009)
14. Freeman, L.: Centrality in social networks conceptual clarification. Social Networks 1(3), 215–239 (1979)
15. Gleich, D., Glynn, P., Golub, G., Greif, C.: Three results on the PageRank vector: eigenstructure, sensitivity, and the derivative. In: Web Information Retrieval and Linear Algebra Algorithms (2007)
16. Grindrod, P., Parsons, M., Higham, D., Estrada, E.: Communicability across evolving networks. Physical Review E 83(4), 046120 (2011)
17. Katz, L.: A new status index derived from sociometric analysis. Psychometrika 18(1), 39–43 (1953)
18. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of the ACM (JACM) 46(5), 604–632 (1999)
19. Langville, A.N., Meyer, C.D.: Updating PageRank with iterative aggregation. In: WWW, pp. 392–393 (2004)
20. Langville, A.N., Meyer, C.D.: Google's PageRank and Beyond: The Science of Search Engine Rankings. Princeton University Press (2006)
21. Mathieu, F., Bouklit, M.: The effect of the back button in a random walk: application for PageRank. In: WWW, pp. 370–371 (2004)
22. Morrison, J.L., Breitling, R., Higham, D.J., Gilbert, D.R.: GeneRank: using search engine technology for the analysis of microarray experiments. BMC Bioinformatics 6(1), 233 (2005)
23. O'Madadhain, J., Smyth, P.: Eventrank: A framework for ranking time-varying networks. In: LinkKDD, pp. 9–16. ACM (2005)
24. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web (1998)
25. Ratkiewicz, J., Fortunato, S., Flammini, A., Menczer, F., Vespignani, A.: Characterizing and modeling the dynamics of online popularity. Physical Review Letters 105(15), 158701 (2010)
26. Sun, J., Tao, D., Faloutsos, C.: Beyond streams and graphs: dynamic tensor analysis. In: SIGKDD, KDD 2006, pp. 374–383. ACM, New York (2006)
27. Suzuki, Y., et al.: Identification and characterization of the potential promoter regions of 1031 kinds of human genes. Genome Research 11(5), 677–684 (2001)
28. Various. Wikipedia database dump, Version from (March 6, 2009), http://en.wikipedia.org/wiki/Wikipedia:Database_download
29. Various. Wikipedia pageviews (2011), http://dumps.wikimedia.org/other/pagecounts-raw/ (accessed in 2011)

Multi-commodity Allocation for Dynamic Demands Using PageRank Vectors

Fan Chung[1], Paul Horn[2], and Jacob Hughes[1]

[1] University of California, San Diego
[2] Harvard University

Abstract. We consider a variant of the contact process concerning multi-commodity allocation on networks. In this process, the demands for several types of commodities are initially given at some specified vertices and then the demands spread interactively on a contact graph. To allocate supplies in such a dynamic setting, we use a modified version of PageRank vectors, called Kronecker PageRank, to identify vertices for shipping supplies. We analyze both the situation that the demand distribution evolves mostly in clusters around the initial vertices and the case that the demands spread to the whole network. We establish sharp upper bounds for the probability that the demands are satisfied as a function of PageRank vectors.

1 Introduction

Efficient allocation of resources to meet changing demands is a task arising in numerous applications. For example, institutions such as governments or corporations respond to the needs of a populace, and wish to meet the demands within allowed expenditure of resources. In some cases where demand spreads, one has to be able to act before demand becomes unmanageable. In the case of an epidemic, for instance, one desires to find a way to distribute medicine so that the disease will be contained. Such problems have been studied in several contexts using the contact process model [9], [7], [2], [11], [5]. In [5], it was demonstrated how to use PageRank vectors to both restrict the number of nodes inoculated and to provide certain containment guarantees.

In this paper, we study a variant of the classical contact process, a continuous time Markov process on a contact graph. In our scenario, vertices in the graph each have varying levels of demand for multiple commodities. Demand at a vertex propagates to its neighbors at a rate depending on the current demand. Our model allows for interactions between different commodities; demand for one commodity may influence demand for another. This fits many scenarios that arise, for instance demand for iPhones may accelerate the demand for iPads. As another example, demand at a node can might be viewed as a measure of discontent with the current supply of a resource. It is natural for an unhappy node to create unrest in its neighbors. As the modified contact process continues, demands at a vertex are increased at a rate based on the demands at neighboring vertices. There are also decreased at a satisfaction rate, which can be thought

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 138–152, 2012.

of as a frequency of shipments. Demand spreads at rates which are a linear combination of demands from neighboring vertices. These rates are encapsulated in a spread matrix, B, roughly analogous to the infectivity parameter in the classical contact process. The goal of this paper is to find satisfaction rates, dependent on the spread matrix B and the geometry of the contact graph which ensure that eventually all vertices have no demand and the process dies out. Our process will be defined, in detail, in Section 2.

To satisfy the demands which evolve according to our model as defined in Section 2, the goal is to ship commodities and supply vertices with unsatisfied demands in an efficient way. The model here differs somewhat from typical resource allocation problems in the sense that we do not specify the location of the "warehouses" for the supply. We will not be concerned with either the sources of the supply or the detailed incremental costs of shipping supply. Instead, our goal is to identify *how often* to ship each commodity to a particular vertex, in order to contain and satisfy demands, given an initial seed set. The reader is referred to [4] for the usual resource allocation problem.

Contact graphs of interest take many forms: Cities and countries exert trade pressure on neighboring cities and countries. Communication on the internet can also spread demand for products, or discontent leading to a revolution. Instead of studying this problem on particular models for these contact graphs, we study the problem on arbitrary finite graphs. Two schemes of making shipments are considered. First is a global solution which involves "scheduling shipments" to all vertices in the graph, and ensures that all demand is satisfied in $O(\log n)$ time, with high probability and regardless of the initial demand. This is made precise in Theorem 1 once the model is formally defined. The next scheme is a local solution, in the sense that shipments are scheduled to only a subset of vertices which contain the initial demand. In particular, when the contact graph has some clustering structure we are interested in subsets so that the demand within the subset is satisfied quickly (in $O(\log n)$ time) and demands reach a vertex not receiving shipments with low probability. Precise results to this end are given in Section 5.

This latter scheme relies on understanding the geometry of the particular contact graph being studied. Our scheme uses PageRank to identify important vertices and to bound the probability that demand in our process leaves a set. We also introduce a variant of PageRank, which we call Kronecker PageRank and is introduced in Section 3, which provides sharper bounds by better utilizing the structure of the spread matrix B as well as the geometry of the graph in its estimates. Our analysis provides a tradeoff in the following sense: we may use PageRank estimates to *identify* a set of vertices containing the initial demand which are important to ship to or we may use our PageRank estimates to give a *guarantee* on the escape probability of leaving a particular set of our choosing. Precise results to this end are given in Theorems 3 and 4, using standard PageRank and Kronecker PageRank respectively.

2 Preliminaries and the Demand Model

We model demand spreading within an undirected simple graph, $G = (V, E)$. We write this $v \sim w$ when v and w are adjacent. For each vertex $v \in V$, let d_v be the degree of v, which is the number of neighbors of v. While not strictly necessary, we assume that there is a self loop at each vertex. In this case, d_v includes v in the count of neighbors and hence the loop counts as 1 towards the degree. We let $n = |V|$, the number of nodes of G. An exponential random variable with parameter λ has probability density function given by $f(x) = \lambda e^{-\lambda x}$ for $x \geq 0$, and 0 for $x < 0$. This distribution will be denoted $\text{Exp}(\lambda)$. One important property of exponential random variables is the memoryless property: if X is an exponential random variable then for any constants $a, b > 0$,

$$\mathbb{P}(X > a + b | X > a) = \mathbb{P}(X > b).$$

If X and Y are independent and $X \sim \text{Exp}(\lambda_1)$, $Y \sim \text{Exp}(\lambda_2)$ then $\min\{X, Y\} \sim \text{Exp}(\lambda_1 + \lambda_2)$. A Poisson point process at rate λ is a sequence of random variables $\{X_i\}_{i=1}^{\infty}$ so that X_1 and $X_i - X_{i-1}$, for $i \geq 2$, has distribution $\text{Exp}(\lambda)$.

Before we describe our model, let us briefly recall the contact process on a graph G, which we denote $CP(T, \beta, \boldsymbol{\sigma}, G)$. In the contact process (see for example [2] or [5]), a disease initially infects a set $T \subseteq V(G)$. The disease has an infectivity parameter, β, and each vertex has a certain amount of "medicine" σ_v. An infected vertex v infects its neighbor u at times given by a Poisson point process $\{X^{uv}\}$ at rate β, and each infected vertex is cured at times given by a Poisson point process at rate σ_v. In the most frequently studied case, σ is constant and the host graph is an infinite graph. The process ends when all vertices are cured, and the basic problem is to determine under which conditions on σ, and β the process ends almost surely. In the case of finite graphs, if $\sigma_v > 0$ for every vertex, it is easy to observe that the process ends a.s., so the problem becomes determining *how fast* the process ends.

The $k-$commodity dynamic demand model on a graph G is a variant of the contact process, $DD(\boldsymbol{\tau}(0), B, \boldsymbol{\sigma}, G, N)$. In this situation, B is a real valued $k \times k$-matrix (not assumed to be symmetric, or even non-negative), which we call the spread matrix. The supply function is $\boldsymbol{\sigma} : V \to \mathbb{R}^k$, and $\boldsymbol{\tau}(0) : V \to \mathbb{N}^k$ is the initial demand. The state of the process at time t is given by $\boldsymbol{\tau}(t) : V \to \mathbb{R}^k$, which gives the demands for each vertex at time t. We use $\boldsymbol{\tau}_v(t) \in \mathbb{N}^k$ to denote the demand at vertex v at time t, and $\tau_v^j(t) \in \mathbb{N}$ to denote the demand for commodity j at time t. A node v is said to be *satisfied* at time t if $\boldsymbol{\tau}_v(t) = \mathbf{0}$, and *unsatisfied* otherwise. $N \in \mathbb{N}$ serves as a uniform bound for the maximum demand for any resource at any point (for instance, it could be the population of a city, if a vertex is a city.) The existence of N is simply to ensure integrability of some random variables. In the case of the contact process, $N = 1$.

The spread matrix $B = [\beta_{ij}]$ describes how the demand for one commodity influences demands for other commodities. The i, j entry of B, β_{ij}, determines

the spread rate of the demand for commodity j that is caused by demand for commodity i. In particular, we can describe the rate of spread events as follows. If v is a node that is unsatisfied at time t, and w an adjacent vertex, then there are spread events from v to w with rates $\max\{\boldsymbol{\tau}_v(t)B, 0\}$. That is the rate at which τ_w^j increases due to the demand at v is given by $\max\{\sum_i \tau_v^i(t)\beta_{ij}, 0\}$. Here, when we say an event occurs with rate λ, we mean that the elapsed time until that event takes place is distributed as $\text{Exp}(\lambda)$. Because the minimum of exponential random variables is itself an exponential random variable, we can capture the total spreading rates in a condensed form. We define the rate function at time t, $\boldsymbol{\rho}(t) : V \to \mathbb{R}^k$, by

$$\boldsymbol{\rho}_v = \sum_{w \sim v} \boldsymbol{\tau}_w(t)B = (\boldsymbol{\tau}(t)(A \otimes B))_v \,,$$

where $\boldsymbol{\tau}(t)$ is viewed as a vector with indices indexed by $V \times k$. $\rho_v^i(t)$ is the rate at which τ_v^i is increasing at time t. Any spread events that would raise τ_v^i above N are ignored.

Supply events occur with rates given by $\boldsymbol{\tau}(t)\text{Diag}(\boldsymbol{\sigma})$, independently of any neighboring supply events. That is, the time until τ_v^i is decreased by 1 is distributed as $\text{Exp}(\sigma_v^i \tau_v^i)$.

We briefly give a construction of the process, to show it is well-defined. Let $\overrightarrow{E}$ denote the set of ordered edges; that is, ordered pairs that are edges in the graph, so that uv and vu are distinct. We run independent Poisson point processes $\{X_e^{j,\boldsymbol{\rho}}\}_{e \in \overrightarrow{E}(G), j \in [k], \boldsymbol{\rho} \in [N]^k}$ so that $X_e^{j,\boldsymbol{\rho}}$ is at rate $\max\{0, [\boldsymbol{\tau}_v B]_j\}$ and independent Poisson point processes $\{X_v^{i,n}\}_{v \in V(G), i \in [k], n \in [N]}$ so that $X_v^{i,n}$ is at rate $n\sigma_v^i$. Then these finitely many point processes can easily be seen to define the entire process; a spread event of type j from a vertex v to a vertex u which is currently in state $\boldsymbol{\rho}$ is controlled by the point process $X_{vu}^{j,\boldsymbol{\rho}}$ with satisfaction events handled similarly.

Such a formulation is that it gives an easy coupling between processes that shows that if $B' \leq B$ pointwise, the stochastic process $DD(\boldsymbol{\tau}(0), B, \boldsymbol{\sigma}, G, N)$ stochastically dominates $DD(\boldsymbol{\tau}(0), B', \boldsymbol{\sigma}, G, N)$. That is, in the coupling the demands in the B process are always at least those in the B' process. This is accomplished by noting that the rates $\boldsymbol{\rho}B \geq \boldsymbol{\rho}B'$ pointwise for all $\boldsymbol{\rho} \in \mathbb{N}^k$. We thus take point processes $Y_e^{j,\rho}$ at rate $[\boldsymbol{\rho}B - \boldsymbol{\rho}B']_i$. If the point processes $\{X_e^{j,\rho}\}$ and $\{X_v^{i,n}\}$ are used to determine $DD(\boldsymbol{\tau}(0), B, \boldsymbol{\sigma}, G, N)$, then the point processes $\{X_e^{j,\rho} \cup Y_e^{j,\rho}\}$ and $\{X_v^{i,n}\}$ are used to determine $DD(\boldsymbol{\tau}(0), B', \boldsymbol{\sigma}, G, N)$.

In particular, this allows us to replace B with B', where $B'_{ij} = \max\{B_{ij}, 0\}$, and conclusions about the extinction of the B' process still hold for B. Furthermore, this turns out not to be entirely unreasonable. One hopes that the negative entries in B would afford better bounds on the extinction time, but in many cases with negative entries in B extinctions of some demand types mean that the process is eventually run in a non-negative case. In light of this, we will assume for the rest of this paper that B is *non-negative* for convenience.

Given an initial demand $\boldsymbol{\tau}(0)$ and spread matrix B, our goal is to find a supply function $\boldsymbol{\sigma}$ such that demand is satisfied. Ideally we would like to do this with small supply rates. Furthermore, the supply rates should only depend on the contact graph G, the spread matrix B, and the initial demand $\boldsymbol{\tau}(0)$, but not on t or $\boldsymbol{\tau}(t)$.

Because we take B to be an arbitrary positive $k \times k$ matrix, we will at times need to use various matrix norms in order to understand the process. For a square matrix B, there are many different matrix norms that can be used (see [8]). We will use the following notation for the following norms:

1. $||B||_1 = \sum_{i,j} |a_{ij}|$ is the ℓ_1 *norm.*
2. $|||B|||_1 = \max_j \sum_i |a_{ij}|$ is the *maximum column sum norm.*
3. $|||B|||_\infty = \max_i \sum_j |a_{ij}|$ is the *maximum row sum norm.*
4. $|||B|||_2 = \max\{\sqrt{\lambda} | \lambda \text{ is an eigenvalue of } A^*A\}$ is the *spectral norm.*

3 PageRank and Kronecker PageRank

The notion of PageRank was first introduced by Brin and Page [3] in 1998 for Google's search algorithms. Although PageRank was originally used for the Web graph, it is well defined on any finite graph G. The basis of PageRank is random walks on graphs. A walk is a sequence of vertices $(v_0, v_1, ..., v_k)$ where $v_i \sim v_{i+1}$. A simple random walk of length k is a sequence of random variables $(x_0, ..., x_k)$ where the starting vertex x_0 is chosen according to some distribution, and

$$\mathbb{P}(x_{i+1} = v | x_i) = \begin{cases} \frac{1}{d_{x_i}} \text{ if } x_i \sim v \\ 0 \quad \text{if } x_i \nsim v \end{cases}.$$

Let D be the diagonal degree matrix with entries $D_{vv} = d_v$, and let A be the adjacency matrix with entries

$$A_{vw} = \begin{cases} 1 \text{ if } v \sim w \\ 0 \text{ if otherwise.} \end{cases}$$

Then the transition probability matrix for a random walk on G is given by $W = D^{-1}A$.

We will mainly use a modified version of the PageRank, called personalized PageRank. Personalized PageRank has two parameters, a jumping constant $\alpha \in [0, 1]$ and a seed $\mathbf{s}$ which is some probability distribution on the vertex set V of G.

The personalized PageRank vector $\mathbf{pr}(\alpha, s)$ for jumping constant α and the seed distribution $\mathbf{s}$ on V is given by

$$\mathbf{pr}(\alpha, \mathbf{s}) = \alpha \sum_{\ell=0}^{\infty} (1-\alpha)^\ell \mathbf{s} W^\ell.$$

Note that here we view $\mathbf{s}$ as a row vector, which is our convention for all vectors throughout this paper. We note that the PageRank vector is also the solution to the recurrence relation

$$\mathbf{pr}(\alpha, \mathbf{s}) = \alpha\mathbf{s} + (1-\alpha)\mathbf{pr}(\alpha, \mathbf{s})W.$$

The original definition of PageRank [3] is the special case where $\mathbf{s}$ is the uniform distribution over all the vertices.

For a subset of vertices $H \subset V$, the volume of H is the sum of degrees of the vertices of H. The Cheeger Ratio of H, $h(H)$, measures the cut between H and $\bar{H}$ via the relationship

$$h(H) = \frac{e(H, \bar{H})}{\min\{\mathrm{vol}(H), \mathrm{vol}(\bar{H})\}}.$$

The $\alpha-$core of a subset H is the set of vertices

$$C_\alpha = \left\{ v \in H | \mathbf{pr}(\alpha, 1_v)1_H \geq 1 - \frac{h}{\alpha} \right\}.$$

Personalized PageRank naturally gives bounds on the probability that demands leave a given set in the k-commodity dynamic demand model. These bounds lose some of the structure of the process given by B, however. In order to better understand the process and get tighter bounds, we will use generalization of the personalized PageRank vector, the Kronecker PageRank vector.

If B is an $k \times k$ matrix, and A an $n \times n$ matrix, then the Kronecker product $A \otimes B$ is the $nk \times nk$ block matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nn}B \end{pmatrix}$$

With this, we can define Kronecker PageRank.

Definition 1 (Kronecker PageRank). *Let B be a square $k \times k$ matrix with spectral radius strictly less than 1, and W be the transition matrix for a random walk on a graph G. Let $\mathbf{s}$ be a non-negative vector in $\mathbb{R}^{k \times |V|}$. The Kronecker PageRank vector with parameters B and $\mathbf{s}$ is defined as*

$$\mathbf{Kpr}(B, \mathbf{s}) = \sum_{\ell=0}^{\infty} \mathbf{s}(W \otimes B)^\ell = \sum_{l=0}^{\infty} \mathbf{s}(W^\ell \otimes B^\ell)$$

Requiring the spectral radius of B less than 1 is necessary to ensure convergence of the infinite sum, as the spectrum of $W \otimes B$ is the product of the spectra of W and B. Since the eigenvalues of W have absolute value at most 1, the sum will converge.

We note that in the case where B is a 1×1 matrix $B = \beta < 1$ and $\mathbf{s}$ is a probability distribution, then we have the relationship

$$\mathbf{Kpr}(B, \mathbf{s}) = \sum_{l=0}^{\infty} \mathbf{s}(W \otimes \beta)^l = \sum_{l=0}^{\infty} \mathbf{s}\beta^l W^l = \frac{1}{1-\beta}\mathbf{pr}(1-\beta, \mathbf{s}),$$

so the Kronecker PageRank is a natural extension of personalized PageRank. We will see in Theorem 4 that the Kronecker PageRank will arise naturally in our analysis in Section 3, and give better bounds than those that will be afforded by standard PageRank by incorporating the spread matrix.

4 Global Analysis: Supplying Every Vertex

Here we show that if supply rates are above a certain threshold, then with probability approaching 1 demands will be satisfied. Recall that B is a non-negative real valued $k \times k$ matrix, and $\boldsymbol{\sigma}$ is the vector of supply rates for the process $DD(\boldsymbol{\tau}(0), B, \boldsymbol{\sigma}, G, N)$.

Theorem 1. *Consider the $k-$commodity demand model on a graph G with n vertices parameterized by spread matrix $B = [\beta_{ij}]$. Let $X(t) = ||\boldsymbol{\tau}(t)||_1$, the total amount of demand at time t. If the supply rates to each vertex v are $\sigma_v^i > d_v\left(\sum_j \frac{\beta_{ij}+\beta_{ji}}{2}\right) + \delta$ for $\delta > 0$ then with probability $1-\epsilon$ all vertices are satisfied at time t for all*

$$t > \frac{1}{\delta}\left(\frac{1}{2}\log(nk) + \log(X(0)) + \log\left(\frac{1}{\epsilon}\right)\right).$$

Proof

We begin by considering the quantity $\frac{\partial}{\partial t}\mathbb{E}[\boldsymbol{\tau}(t)]$. From the discussion in Section 2, we know that demand is increasing with rates given by $\boldsymbol{\rho}(t) = \boldsymbol{\tau}(t)\,(A \otimes B)$, but also demand decreases proportionally to the supply rates and current demand. Indeed, let $S = \mathrm{diag}(\boldsymbol{\sigma})$, the diagonal $nk \times nk$ matrix with entries given by the supply vector. Then demand decreases at each vertex according to rates given by the supply rate vector $\boldsymbol{\tau}(t)S$.

It is not difficult to encapsulate all of the above information in the simple expression

$$\frac{\partial}{\partial t}\mathbb{E}[\boldsymbol{\tau}(t)] = \mathbb{E}[\boldsymbol{\rho}(t) - \boldsymbol{\tau}(t)S] = \mathbb{E}[\boldsymbol{\tau}(t)](A \otimes B - S). \tag{1}$$

A detailed proof of (1) is left to the appendix for space reasons.

Solving the matrix differential equation with initial condition $\mathbb{E}[\boldsymbol{\tau}(0)] = \boldsymbol{\tau}(0)$ yields

$$\mathbb{E}[\boldsymbol{\tau}(t)] = \boldsymbol{\tau}(0)e^{t(A\otimes B - S)}. \tag{2}$$

Let $Q = A \otimes B - S$. Then by [6], $|||e^{tQ}|||_2 \leq e^{t\nu}$, where ν is the largest eigenvalue of $\frac{Q+Q^*}{2}$. We note that $\frac{Q+Q^*}{2} = A \otimes (\frac{B+B^*}{2}) - S$, which has diagonal terms $\beta_{ii} - \sigma_v^i$, ranging over all values of v and i. By the Gershgorin Circle Theorem, the eigenvalues of $\frac{Q+Q^*}{2}$ are contained in the intervals

$$\left[-(d_v - 2)\beta_{ii} - d_v \left(\sum_{j \neq i} \frac{\beta_{ij} + \beta_{ji}}{2} \right) - \sigma_v^i, \;\; d_v \left(\sum_j \frac{\beta_{ij} + \beta_{ji}}{2} \right) - \sigma_v^i \right].$$

Since $\sigma_v^i > d_v \left(\sum_j \frac{\beta_{ij}+\beta_{ji}}{2} \right) + \delta$ all the eigenvalues of $\frac{Q+Q^*}{2}$ are less than $-\delta$. Therefore

$$\begin{aligned} \mathbb{E}[X(t)] &= ||\tau(0)e^{t(A\otimes B-S)}||_1 \leq \sqrt{nk}||\tau(0)e^{t(A\otimes B-S)}||_2 \\ &\leq \sqrt{nk}||\tau(0)||_2|||e^{t(A\otimes B-S)}|||_2 \leq \sqrt{nk}||\tau(0)||_1 e^{t\nu} \\ &\leq \sqrt{nk}X(0)e^{-t\delta} \end{aligned}$$

Thus Markov's inequality gives that
$\mathbb{P}(X(t) > 0) < \epsilon$ if $t > \frac{1}{\delta}\left(\frac{1}{2}\log(nk) + \log(X(0)) + \log\left(\frac{1}{\epsilon}\right)\right)$. □

We note that this approach works for all initial distributions $\boldsymbol{\tau}(0)$, and on any graph G. In particular, it is agnostic to the shape of the graph. This indicates that in many situations this approach may be overkill and that we could have used smaller supply rates. In the next section, we analyze the process more carefully and give conditions that depend on the initial distribution of demand and the geometry of the underlying contact graph.

5 Local Analysis: Supplying a Small Subset

For the remainder of the discussion, it is convenient to introduce reformulation of the model that takes advantage of the fact that demands take on integer values. Rather than view demands as a function $\tau : V \to \mathbb{N}^k$, we view demands as discrete objects sitting on each node. Borrowing language from chip-firing games on graphs (see, for example, [10]) we view units of the demand as chips located on vertices of the graph. For example, if $k = 7$ then for a vertex v with $\boldsymbol{\tau}_v(t) = (0, 1, 2, 0, 2, 0, 3)$ then we would say that at time t there was 1 2-chip, 2 3-chips, 2 5-chips, and 3 7-chips at vertex v, corresponding to 1 "unit of demand" for commodity 2, etc. Unlike in classical chip-firing games, the number of chips is not static, and the process is continuous time. We restate the possible transitions in terms of demand chips. For an $i-$chip at vertex v, there are two types of transition events:

- For each vertex $w \sim v$ and each $j = 1, ..., \ell$, a $j-$chip is added at w at rate β_{ij}. When this occurs we say that the new $j-$chip is created by the $i-$chip.
- The $i-$chip itself is removed with rate σ_v^i.

Due to the properties of exponential random variables, the rates add linearly, and the model is equivalent to the original description discussed in Section 2. The main advantage of this reformulation is the ability to trace back the history of a chip. If there is a chip c at vertex v at time t, then either c existed at time $t = 0$, or there is a sequence of ℓ chips $(c_0, ..., c_\ell = c)$ located at vertices along a walk $\pi = (v_0, v_1, ..., v_\ell = v)$ with the following properties:

1. c_0 existed at $t = 0$
2. c_r is created by c_{r-1} for $r = 1, ..., \ell$.

We allow π to have repeated vertices to allow for the case where demand created more demand at the same vertex. If a chip c exists at time 0, we refer to it as an *initial chip*.

For a path $\pi = (v_0, v_1, ..., v_\ell)$ and a chip c_0 located at v_0, we define the event $\mathcal{S}_{\pi,c_0}$ to be the event there is a sequence of m chips $(c_0, ..., c_\ell)$ located respectively at $(v_0, v_1, ..., v_\ell)$ and c_r is created by c_{r-1} for $r = 1, ..., \ell$.

It is important to note that $\mathcal{S}_{\pi,c_0}$ occurring does not imply that there is any demand at v_ℓ at time t because it could be satisfied sometime before t. However, if there is a demand at v_ℓ at time t, then $\mathcal{S}_{\pi,c}$ must have occurred for some initial chip c at vertex v_0 and some walk π from v_0 to v_ℓ.

We begin by relating $\mathbb{P}(S_{\pi,c_0})$ to the length of the walk π. Inspired by Theorem 1 we make the assumption that supply rates are proportional to the degree of the vertices. That is, we assume that $\sigma_v^i > \mu_i(d_v)$ for all v for constants $\mu_i > 0$.

Lemma 2. *Let $M = diag(\mu_1, ..., \mu_k)$, $\hat{B} = M^{-1}B$ and $\zeta = \min\{|||\hat{B}|||_1, |||\hat{B}|||_\infty\}$ Then for any chip c_0 located at v_0 and any walk $\pi = (v_0, ..., v_\ell)$ of length ℓ,*

$$\mathbb{P}(S_{\pi,c_0}) \leq k\zeta^\ell \prod_{j=0}^{\ell} \frac{1}{d_{v_j}}$$

Proof

Let S_r denote the event that a chip c_r at v_r creates a chip at v_{r+1}. If c_r is an $i-$chip, then for it to create any chip at v_{r+1} a spread event must occur before c_r is removed. The time until c_r creates a $j-$chip at v_{r+1} is an exponential random variable with rate β_{ij}. Since the time until c_r is removed is given by $\text{Exp}(\sigma_v^i)$, the probability of c_r creating a $j-$chip is $\frac{\beta_{ij}}{\beta_{ij}+\sigma_v^i} \leq \frac{\beta_{ij}}{\sigma_v^i} < \frac{\beta_{ij}}{\mu_i d_{v_r}}$. Thus $\mathbb{P}(S_r) < \sum_{i,j} \frac{\beta_{ij}}{\mu_i d_{v_r}} = \frac{1}{d_{v_r}} \mathbf{1}\hat{B}\mathbf{1}^*$.

For a walk π of length ℓ, we want to consider the intermediate steps more carefully. Since there are ℓ transitions that occur, we can use the same reasoning as above to obtain the bound

$$\mathbb{P}(S_{\pi,c}) < \prod_{r=0}^{\ell} \frac{1}{d_{v_r}} \mathbf{1}\hat{B}^\ell \mathbf{1}^* = \prod_{r=0}^{\ell} \frac{1}{d_{v_r}} ||\hat{B}^\ell||_1 \leq k \prod_{r=0}^{\ell} \frac{1}{d_{v_r}} |||\hat{B}^\ell|||_1 \leq k \prod_{r=0}^{\ell} \frac{1}{d_{v_r}} |||\hat{B}|||_1^\ell.$$

The factor of k that appears in the final lines above is just a consequence of switching from the vector 1-norm $||\hat{B}^\ell||_1$ to maximum column sum norm $|||\hat{B}^\ell|||_1$ (see [8]).

We could have just as easily switched to the maximum row sum norm and obtained the term $k|||\hat{B}|||_\infty^\ell$, and so it follows that

$$\mathbb{P}(S_{\pi,c}) < \min\{k\prod_{r=0}^{\ell}\frac{1}{d_{v_r}}|||\hat{B}|||_1^\ell, k\prod_{r=0}^{\ell}\frac{1}{d_{v_r}}|||\hat{B}|||_\infty^\ell\} = k\zeta^\ell\prod_{j=0}^{\ell}\frac{1}{d_{v_j}}.$$

□

We note that the use of $\zeta = \min\{|||\hat{B}|||_1, |||\hat{B}|||_\infty\}$ in Lemma 2 reflects the difficulty in working with arbitrary spread matrices B. For certain classes of spread matrices (e.g. if B is symmetric or diagonalizable) it is possible to obtain tighter bounds. Lemma 2 will be allow us to obtain a bound using PageRank, but note that our use of matrix norms ignores some of the structure of the spread matrix B. Following the proof of Theorem 3 below, a more careful analysis fully using the structure of the spread matrix B will lead naturally to use of Kronecker PageRank, which we explore in Theorem 4.

Theorem 3. *Suppose that initial demand is contained in $S \subset H \subset V$ with and each vertex $v \in H$ has supply rates $\sigma_v^i > \mu_i d_v$, and $\sigma_w^i = 0$ for $w \in \bar{H}$. Let $M = diag(\mu_1, ..., \mu_k)$, $\hat{B} = M^{-1}B$ and $\zeta = \min\{|||\hat{B}|||_1, |||\hat{B}|||_\infty\}$. Let $\boldsymbol{x}(t)$ be defined by $x_v(t) = \sum_i \tau_v^i(t)$, and $X(t) = ||\boldsymbol{\tau}(t)||_1$. Let E_H denote the event that demands spread outside the set H. Then*

1. $\mathbb{P}(E_H) \leq \frac{X(0)}{\zeta} pr\left(1-\zeta, \frac{\tau(0)}{X(0)}\right) 1_{\bar{H}}^*$
2. *If S in the $(1-\zeta)$ core of H, then $\mathbb{P}(E_H) \leq \frac{2X(0)h(H)}{\zeta(1-\zeta)}$, where $h(H)$ is the Cheeger ratio of H.*

Proof. Let P_ℓ denote the set of all paths of length ℓ from an initial chip in S to $\bar{H}$ such that the first $k-1$ steps are in H. Let $P = \bigcup_{l=1}^{\infty} P_\ell$. The key observation is that if $w \in \bar{H}$ ever has demand, then $S_{\pi,c}$ must have occurred for some initial chip c and path π from the location of c to w. Thus we can use the union bound to get that

$$\begin{aligned}
\sum_{\pi\in P}\mathbb{P}(S_{\pi,c}) &\leq \sum_{\ell}\sum_{(\pi,c)\in P_\ell}\mathbb{P}(S_{\pi,c}) \\
&\leq \sum_{\ell}\sum_{v_0\in S}\sum_{c\ at\ v_0}\sum_{v_\ell\in\bar{H}}\sum_{\pi=(v_0,\ldots,v_\ell)\in P_\ell}\mathbb{P}(S_{\pi,c}) \\
&\leq \sum_{\ell}\sum_{v_0\in S}\sum_{c\ at\ v_0}\sum_{v_\ell\in\bar{H}}\sum_{\pi=(v_0,\ldots,v_\ell)\in P_\ell}\zeta^\ell\prod_{r=0}^{\ell}\frac{1}{d_{v_r}} \\
&= \sum_{\ell}\mathbf{x}(0)\zeta^\ell(D^{-1}A)^\ell 1_{\bar{H}}^* \\
&= \sum_{\ell}\mathbf{x}(0)\zeta^\ell W^\ell 1_{\bar{H}}^* \quad = \quad \frac{X(0)}{\zeta}\mathbf{pr}(1-\zeta, \frac{\mathbf{x}}{X(0)})1_{\bar{H}}^*.
\end{aligned}$$

proving the first statement. The second statement follows the same proof as Theorem 3.2 of [5]. □

Finally we show how Kronecker PageRank arises in a natural way as the bound of the escape probability of this process.

Theorem 4. *Suppose that the initial demand is contained in $S \subset H \subset V$ with and each vertex $v \in H$ has supply rates $\sigma_v^i \geq \mu_i d_v$. Let $M = diag(\mu_1, ..., \mu_k)$, $\hat{B} = M^{-1}B$ and $\zeta = ||\hat{B}||_1$. Let $X(t) = ||\boldsymbol{\tau}(t)||_1$, the total amount of demands at time t. Let $\mathcal{E}_H$ denote the event that demands spread outside the set H. Then $\mathcal{E}_H$ can be bounded above using the Kronecker PageRank vector via the relationship:*

$$\mathbb{P}(\mathcal{E}_H) \leq X(0)\boldsymbol{Kpr}\left(\hat{B}, \frac{\tau(0)}{X(0)}\right) 1_{\bar{H}}$$

Proof. Let f be a vector indicator function of commodity type on chips, that is $f(c) = \mathbf{e}_i$ if c is an $i-$chip, where $\mathbf{e}_i$ denotes the ith standard basis vector for $\mathbb{R}^k$. Let C_0 denote the set of initial chips. By the same methods that were used in the proof of Lemma 2, we can bound the probability that demand originating from c ever spreads along a path $\pi = (v_0, v_1, ..., v_\ell)$ by the sum

$$\mathbb{P}(\mathcal{S}_{\pi,c}) \leq f(c)\hat{B}^\ell \mathbf{1}^* \prod_{r=0}^{\ell} \frac{1}{d_{v_r}}$$

Therefore using the same technique as in the proof of Theorem 3 we obtain the bound

$$\sum_{\pi \in P} \mathbb{P}(\mathcal{S}_{\pi,c}) \leq \sum_{\ell} \sum_{u \in S} \sum_{\pi \in B_\ell} \mathbb{P}(\mathcal{S}_{\pi,u}) \leq \sum_{\ell} \sum_{c \in C_0} \sum_{v_\ell \in \bar{H}} \sum_{\pi=(v_0,...,v_\ell) \in P_\ell} \mathbb{P}(\mathcal{S}_{\pi,u})$$

$$\leq \sum_{\ell} \sum_{c \in C_0} \sum_{v_\ell \in \bar{H}} \sum_{\pi=(v_0,...,v_\ell) \in P_\ell} f(c)\hat{B}^\ell \mathbf{1}^* \prod_{r=0}^{\ell} \frac{1}{d_{v_r}} = \sum_{\ell} \tau(0)(D^{-1}A \otimes \hat{B})^\ell \mathbf{1}_{\bar{H}}$$

$$= \sum_{\ell} \tau(0)(W \otimes \hat{B})^\ell \mathbf{1}_{\bar{H}} = X(0)\mathbf{Kpr}(\hat{B}, \frac{\tau(0)}{X(0)})\mathbf{1}_{\bar{H}}$$

□

On the event of non-escape, we would like to guarantee that all demand is satisfied quickly. To make this precise, let $\mathcal{S}_t$ denote the event that all of the vertices are satisfied at time t. In order to complete the analysis of the local case, we would like to bound $\mathbb{P}(\mathcal{S}_t|\overline{\mathcal{E}}_H)$, where $\mathcal{E}_H$ is as in Theorems 3 and 4. Such a bound is not immediately given by Theorem 1. To derive a bound on $\mathbb{P}(\mathcal{S}_t|\overline{\mathcal{E}}_H)$, consider running a modified 'Dirichlet' process which is identical to the standard process with the same supply rates, except demand leaving H is ignored. Let $\mathcal{S}'_t$ denote the event that in Dirichlet process, all of the events are satisfied at time t then $\mathbb{P}(\mathcal{S}'_t)$ can be bounded directly by Theorem 1 as this Dirichlet process restricted to vertices in H is the standard process on H. Furthermore $\mathbb{P}(\mathcal{S}_t \cap \mathcal{E}_H) \leq \mathbb{P}(\mathcal{S}'_t)$. Therefore

$$\mathbb{P}(\mathcal{S}_t|\overline{\mathcal{E}}_H) = \frac{\mathbb{P}(\mathcal{S}_t \cap \overline{\mathcal{E}}_H)}{\mathbb{P}(\overline{\mathcal{E}}_H)} \leq \frac{\mathbb{P}(\mathcal{S}'_t)}{\mathbb{P}(\overline{\mathcal{E}}_H)}.$$

Combining this observation along with Theorems 3 and 4, yields that the probability of escape from H is bounded and if the process does not escape from H it dies quickly.

Theorems 3 and 4 can be used in two different ways. As stated, they provide a way to bound the probability demands escape from a given subset. However, they can be also used to construct such a bounding subset. For example, given initial demand $\boldsymbol{\tau}(0)$ contained in an initial set of vertices $S \subset V$, we can algorithmically construct H such that demand stays in H with probability $1 - \epsilon$ by iteratively selecting vertices with the highest (Kroneceker) PageRank.

6 An Example

An immediate question is whether anything is actually gained by the introduction of Kronecker PageRank. Suppose we have a spread matrix B, and some initial demand on a graph G. We wish to identify a subset of vertices $H \subset G$ to make shipments to so that the escape probability is at most ϵ. We may use either Theorem 3 or Theorem 4 to identify such a set. The bound afforded by Theorem 4 is clearly sharper than the bound in Theorem 3 as the structure of the spread matrix B is taken into account, but it is not guaranteed that the identified set is actually smaller. In many cases it actually is, though depending on B it may not be significantly smaller.

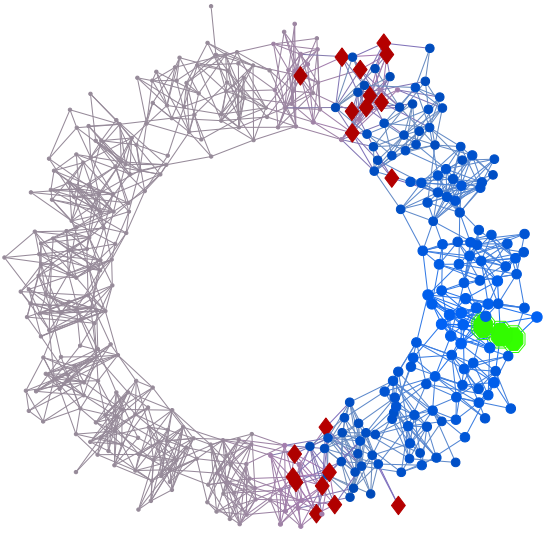
To illustruate, we give a simple example calculation on synthetic data. Our graph G is an instance of the following random process, which is designed to create a graph which contains tighter clusters that are slightly more sparsely connected to neighboring clusters: We begin with a cycle on 20 vertices. Each vertex is then replaced by an instance of the random graph $G_{20,.3}$, that is a graph 20 vertices and each edge existing independently with probability .3. Inter-cluster edges are then created between vertices in neighboring clusters with probability .05.

We consider the case where $k = 4$, and

$$B = \begin{pmatrix} .8 & .4 & .3 & .2 \\ .2 & .7 & .2 & .1 \\ .1 & .2 & .9 & .3 \\ .4 & .2 & .3 & .6 \end{pmatrix}.$$

The initial demand is given by $\boldsymbol{\tau}(0) = \{2, 1, 2, 0, 0, 1, 0, 0, ..., 0\}$. In addition we assume $\mu_i = 2$, and $\zeta = .85$.

We demonstrate the difference between Theorems 3 and 4 in the following way. The figure below shows the graph G. The demands start in the large outlined vertices and spread outward from there. Theorem 4 states that with 95% probability, demands stay in the circular vertices. Theorem 3 states that with 95% probability, demands stay in the diamond and circular vertices. This small example illustrates how the Kronecker PageRank can be used to obtain improved results.



References

1. Andersen, R., Chung, F., Lang, K.: Local Partitioning for Directed Graphs Using PageRank. In: Bonato, A., Chung, F.R.K. (eds.) WAW 2007. LNCS, vol. 4863, pp. 166–178. Springer, Heidelberg (2007)
2. Borgs, C., Chayes, J., Ganesh, A., Saberi, A.: How to distribute antidote to control epidemics. Random Structures & Algorithms 37, 204–222 (2010)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30, 107–117 (1998)
4. Chevaleyre, Y., Dunne, P., Endriss, U., Lang, J., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodriguez-Aguilar, J., Sousa, P.: Issues in multiagent resource allocation. Informatica (2006)
5. Chung, F., Horn, P., Tsiatas, A.: Distributing antidote using PageRank vectors. Internet Mathematics 6, 237–254 (2009)
6. Dahlquist, G.: Stability and error bounds in the numerical integration of ordinary differential equations. Kungl. Tekn. Högsk. Handl. Stockholm. (130), 87 (1959)
7. Ganesh, A., Massoulie, L., Towsley, D.: The effect of network topology on the spread of epidemics, vol. 2, pp. 1455–1466. IEEE (2005)
8. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press (1990)
9. Kiss, I.Z., Green, D.M., Kao, R.R.: Infectious disease control using contact tracing in random and scale-free networks. Journal of the Royal Society, Interface / the Royal Society 3, 55–62 (2006), PMID: 16849217
10. Merino, C.: The chip-firing game. Discrete Mathematics 302, 188–210 (2005)
11. Newman, M.: Spread of epidemic disease on networks. Physical Review E 66 (2002)

7 Appendix

In this section we establish the differential equation used in the proof of Theorem 1, namely:

Lemma 5. *If $\boldsymbol{\tau}(t)$ denotes the demand vector at time t, A is the adjacency matrix of G, B denotes the spread matrix and $S = diag(\sigma)$ denotes the supply matrix, then*

$$\frac{\partial}{\partial t}\mathbb{E}[\boldsymbol{\tau}(t)] = \mathbb{E}[\boldsymbol{\tau}(t)](A \otimes B - S).$$

The key of Lemma 5 are the following two well known and simple facts concerning exponentially distributed random variables. We use the notation $f(h) = O_{h\to 0}(g(h))$ to indicate that $f(h) \leq C\dot{g}(h)$ for h sufficiently small.

Lemma 6. *Suppose X is an exponentially distributed waiting time with rate λ, then*

$$\mathbb{P}(X < h) = \lambda h + O_{h\to 0}(h^2).$$

An immediate corollary is

Lemma 7. *Suppose X, Y are independent exponentially distributed waiting times with rates λ_1, λ_2. Then*

$$\mathbb{P}(X, Y < h) = O_{h\to 0}(h^2).$$

Proof of Lemma 5.
Fix a vertex v and commodity i. We will show that

$$\frac{\partial}{\partial t}\mathbb{E}\left[\boldsymbol{\tau}_v^i(t)\right] = \left[\mathbb{E}[\boldsymbol{\tau}(t)](A \otimes B - S)\right]_v^i,$$

Since this holds for all v, and i the result will follow.

To do this, we compute the derivative by the definition, that is we compute

$$\lim_{h\to 0} \frac{\mathbb{E}[\boldsymbol{\tau}_v^i(t) - \boldsymbol{\tau}_v^i(t+h)]}{h}.$$

To do this, consider the conditional expectation, $\mathbb{E}[\boldsymbol{\tau}_v^i(t) - \boldsymbol{\tau}_v^i(t+h)|\boldsymbol{\tau}(t)]$. Note that by Lemma 7, then probability that two independent events (either two spread events, or two satisfy events or a spread and a satisfy event) occur is $O_{h\to 0}(h^2)$. On the other hand, given a neighbor u of v, and a commodity j, the probability of a spread event originating from this neighbor and commodity in time $(t, t+h)$ is exactly $B_{ji}\tau_u^j(t)h + O_{h\to 0}(h^2)$. Likewise, the probability of a satisfaction event in this time is $\tau_v^i(t)\sigma_v^i h + O_{h\to 0}(h^2)$. Linearity of expectation yields

$$\begin{aligned}\mathbb{E}[\boldsymbol{\tau}_v^i(t) - \boldsymbol{\tau}_v^i(t+h)|\boldsymbol{\tau}(t)] &= \mathbb{E}[\boldsymbol{\tau}(t)(A \otimes B - S)h + O_{h\to 0}(h^2)|\boldsymbol{\tau}(t)] \\ &= \boldsymbol{\tau}(t)(A \otimes B - S)h + O_{h\to 0}(h^2)|\boldsymbol{\tau}(t).\end{aligned}$$

In particular, note that the $O_{h\to 0}(h^2)$ term means that there is a (large) constant $C = C(\tau(t), A, B)$, so that $O_{h\to 0}(h^2) \leq C \cdot h^2$ for $h \leq 1$. Note that due to our conditioning this constant depends on $\tau(t)$, but critically not on h. Note that this constant is $\sigma(\tau(t))$-measurable.

By the tower property of conditional expectation,

$$\begin{aligned}\lim_{h\to 0}\frac{\mathbb{E}[\boldsymbol{\tau}_v^i(t)-\boldsymbol{\tau}_v^i(t+h)]}{h} &= \lim_{h\to 0}\frac{\mathbb{E}[\mathbb{E}[\boldsymbol{\tau}_v^i(t)-\boldsymbol{\tau}_v^i(t+h)|\boldsymbol{\tau}(0)]]}{h}.\\ &= \lim_{h\to 0}\frac{\mathbb{E}[\boldsymbol{\tau}(t)(A\otimes B-S)h]+\mathbb{E}[\mathbb{E}[O_{h\to 0}(h^2)|\boldsymbol{\tau}(t)]]}{h}\\ &= \boldsymbol{\tau}(t)(A\otimes B-S)+\lim_{h\to 0}\mathbb{E}[O_{h\to 0}(h)].\end{aligned}$$

It suffices to show that

$$\lim_{h\to 0}\left|\mathbb{E}[O_{h\to 0}(h)]\right| \le \lim_{h\to 0}\mathbb{E}[\left|O_{h\to 0}(h)\right|] = 0.$$

But recall that the $O_{h\to 0}(h)$ term is bounded by $C(\boldsymbol{\tau}(t), A, B)\cdot h$ for $h \le 1$. Thus it is enough to show that

$$\lim_{h\to 0}\mathbb{E}[|C(\boldsymbol{\tau}(t), A, B)h|] = 0.$$

This follows from the monotone convergence theorem, so long as

$$\lim_{h\to 0}\mathbb{E}[|C(\boldsymbol{\tau}(t), A, B)|] < \infty.$$

To complete the proof we note that we can give an upper bound on $C(\boldsymbol{\tau}(t), A, B)$ in terms of $||\boldsymbol{\tau}(t)||_1$, n and $\max\{b_{ij}\}$. Indeed, the rates of the active point processes are at most $||\boldsymbol{\tau}(t)||_1 \max\{b_{i,j}\}$; and thus the probability that any pair of point processes both have events in the period $(t, t+h)$ is bounded by

$$C(\boldsymbol{\tau}(t), A, B)h \le C||\boldsymbol{\tau}(t)||_1^2 \max\{b_{i,j}\}^2 n^2 h.$$

But $||\boldsymbol{\tau}(t)||_1 << n\cdot N\cdot k$, where $n = |G|$, k is the number of demands and N is our uniform upper bound for the demand at a point (indeed, this is the precisely the motivation for such a bound.) □

Are We There Yet? When to Stop a Markov Chain while Generating Random Graphs*

Jaideep Ray, Ali Pinar, and C. Seshadhri**

Sandia National Laboratories, Livermore, CA 94550
{jairay,apinar,scomand}@sandia.gov

Abstract. Markov chains are convenient means of generating realizations of networks with a given (joint or otherwise) degree distribution, since they simply require a procedure for rewiring edges. The major challenge is to find the right number of steps to run such a chain, so that we generate truly independent samples. Theoretical bounds for mixing times of these Markov chains are too large to be practically useful. Practitioners have no useful guide for choosing the length, and tend to pick numbers fairly arbitrarily. We give a principled mathematical argument showing that it suffices for the length to be proportional to the number of desired number of edges. We also prescribe a method for choosing this proportionality constant. We run a series of experiments showing that the distributions of common graph properties converge in this time, providing empirical evidence for our claims.

Keywords: graph generation, Markov chain Monte Carlo, independent samples.

1 Introduction

Degree distributions (DD) and joint degree distributions (JDD) are some of the most fundamental properties of real world networks. The degree distribution of an undirected graph G is a vector $\boldsymbol{f}$, where $f(d)$ is the number of vertices of degree d. The *joint degree distribution* is an $n \times n$ matrix $\mathbf{J}$, where the entry $J(i,j)$ is the number of edges between vertices of degree i and degree j. The landmark paper [1] observing heavy-tailed degree distributions in real networks forms the basis of much research on these graphs. Notions like *assortativity* [2], that are captured by the joint degree distribution, are an important metric used to understand these networks. To gain deeper understanding of these graph properties, we often perform experiments trying to understand how the degree

* This work was funded by the Applied Mathematics Program at the U.S. Department of Energy and performed at Sandia National Laboratories, a multiprogram laboratory operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

** This author was supported by an Early-Career award from the Laboratory Directed Research & Development (LDRD) program at Sandia National Laboratories.

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 153–164, 2012.

distribution affects other graph properties. For example, is assortativity correlated with the clustering coefficient [3]? A key ingredient to performing these studies is generating uniform random graphs with a prescribed (joint) DD.

Markov chain Monte Carlo (MCMC) methods are a common means of doing this [4,5,6,7,8]. We start with a given graph with a specified DD (or JDD [8]); it is often a real graph whose properties we are studying. There is a simple and standard procedure that performs a random edge swap preserving the DD [4,9,10] (or JDD [8]). This gives a Markov chain on the space of graphs with the given DD (JDD), and we take many steps to generate a sample. But how many steps should we take to generate a uniform random sample?

If a bound on the mixing time of this chain is known, then that gives a convenient bound on the number of steps to take. For the DD and JDD Markov chains[1], theoretical bounds have been given [4]. These are of the form $O(n^6)$, where n is number of vertices of the graph. Even for a moderate size of $n \approx 1000$, this is quite useless in practice. Empirically, the number of steps is usually chosen quite arbitrarily. Since this sampling can often form the basis of experiments, this is quite dangerous. If a Markov chain has not mixed properly, samples generated may be highly correlated and conclusions drawn from them can be erroneous.

1.1 Results

The primary goal of this paper is to bridge this gap between theory and practice. Our results hold for both DD and JDD Markov chains. The results for JDD are more involved and interesting, so only they are presented in this paper. We give a mathematically principled argument showing that to generate a graph with $|E|$ edges, it suffices to run the Markov chain $O(|E|)$ steps. The constant hidden in the big-Oh depends on a desired accuracy. Our experiments show that $10|E| - 30|E|$ steps are enough for the purpose of ascertaining various graph properties.

1. *Theoretical results:* Mathematically, this range is achieved by approximating the behavior of the entire Markov chain by a set of coupled 2-state Markov chains, one for each pair of vertices. This is a heuristic approximation in case of JDD (but for DD, this is a provable equivalence.) The mixing time of these 2-state chains can be directly bounded by $O(|E|)$ (where the constant is a standard dependence on the desired accuracy). This means that in $O(|E|)$ steps, while we may not be able to assert total mixing, *each edge appears as if we are in the uniform distribution.* Observe that this is certainly a necessary condition for total mixing.

2. *Empirical results:* This is in two parts. First, we give empirical evidence that our predicted length works in practice. It is quite difficult to directly ascertain that a given sample is truly uniform random [11]. So, for a given length ℓ, we generate a number of sample graphs, each with a separate ℓ-length walk, and plot the distribution of a common graph parameter (say, clustering coefficient). We observe that for $\ell > 10|E|$, these distributions converge and do not change

[1] These bounds only hold when the graph generated is not necessarily simple.

further. On the other hand, when ℓ is only $|E|$, the distribution is very far from reaching convergence. Our predictions clearly match the experiments. Next, we justify the approximation of the Markov chain on the space of graphs as a set of coupled 2-state chains. We look at the behavior of an individual edge over a very long walk in the overall Markov chain, i.e., a long binary time-series with 0/1 indicating the absence/presence of an edge at each step of the Markov chain. If our approximations are correct, then thinning this series by a factor of $O(|E|)$ should lead to a sequence of practically independent samples. We run statistical tests to show that this really does happen.

Our idea is similar to Sokal's method [12] for deciding the "sufficiency" of samples obtained from MCMC based on *autocorrelation*. The idea of Sokal, as adopted by Stanton and Pinar [8], was to look at the individual edges as a binary time-series. They then compute the autocorrelation, at different lags, which can be thought of as a measure of how long it takes for the time-series to become uncorrelated. It is suggested to keep walking (in the Markov chain) until all auto-correlations are below a prescribed threshold. However, Sokal's method has two major practical drawbacks - (1) the autocorrelation analysis is performed for all the edges (n^2 in number for a graph with n vertices) that might appear in the MCMC chain and (2) one has to choose a autocorrelation threshold, for which there are no guidelines. In contrast, our method estimates the number of Markov chain steps with a closed-form expression.

2 Theoretical Analysis

We first describe the Stanton and Pinar Markov chain for preserving the joint degree distribution [8]. This is analogous to the degree distribution preserving chain [4,7]. These methods are quite standard and come with schemes to generate a specific graph with a given DD or JDD.

Consider an undirected graph $G = (V, E)$, where $|V| = n$ and $|E| = m$. As mentioned earlier, the *joint degree distribution* is an $n \times n$ matrix $\mathbf{J}$, where $J(i, j)$ is the number of edges between vertices of degree i and degree j. We will also use the degree distribution $\boldsymbol{f}$, where the coordinate $f(d)$ the number of vertices of degree d.

The process of generating a new graph, from an older one, by swapping edges, is called "rewiring". The rewiring is done as follows. We use d_v to denote the degree of v. The process is depicted in Fig. 1. This may lead to self-loops and parallel edges, and there are methods of dealing with this. We will not get into those details, and refer the reader to [8]. Note that every vertex maintains its degree, and the joint degree distribution is always preserved. We also maintain lists of nodes and edges indexed by their degree, so that for a specified degree d, a uniform random edge incident to a degree d vertex can be located. The steps are:

– Pick a uniform random *endpoint*. This is done by choosing a uniform random edge and choosing each endpoint with probability 1/2. Suppose we choose endpoint u_1 incident to edge (u_1, v). d_v is arbitrary. See Fig. 1.

– Choose a uniform random edge with an endpoint of degree d_{u_1}. Let this edge be (u_2, w). d_w is arbitrary.

– Swap edges (u_1, v) and (u_2, w). This adds edges (u_1, w) and (u_2, v) and removes (u_1, v) and (u_2, w).

Details of the rewiring scheme and a Markov chain driving it to generate (correlated) graph samples e.g., discussions of ergodicity etc., can be found in [8].

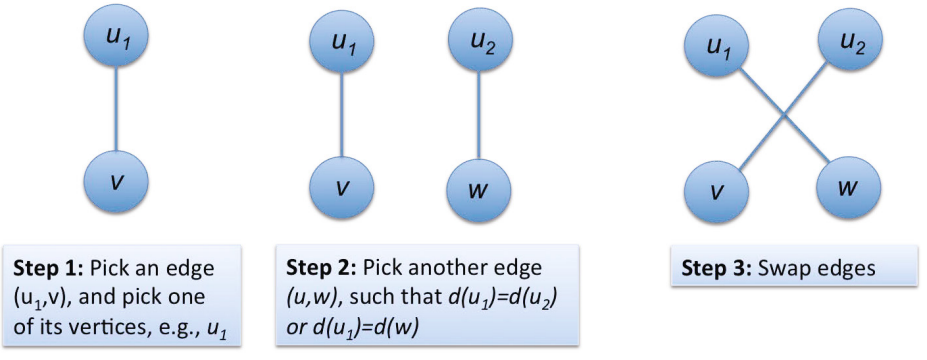


Fig. 1. The swapping operation for the Markov chain algorithm

2.1 Approximation by Many 2-State Markov Chains

Consider a fixed pair of labeled vertices (u, v). Let us try to understand the probability that this edge appears or disappears. Based on this, we can approximate the behavior of the pair (u, v) as a Markov chain. We start with a simple yet important claim.

Claim. Suppose at some stage in the Markov chain, the edge (u, v) is present. The probability that it is removed in the next step is

$$\frac{1}{m} + \frac{f(d_u)d_u + f(d_v)d_v - d_u - d_v}{2m^2}. \tag{1}$$

Proof. The swapping procedure picks two edges, which we shall refer to as e (the first edge) and e' (the second edge). If e is chosen to be (u, v), then (u, v) will definitely be swapped out. The probability of this is $1/m$. On the other hand, e may not be (u, v) but e' could be (u, v). If the random endpoint of e chosen has degree d_u (and is not u), then we might choose e' to be (u, v). The total number of edges incident to degree d_u vertices (but not u) is $(f(d_u) - 1)d_u$. Any of these edges is a potential candidate for e. Hence, the probability of choosing e with this property, and then $e' = (u, v)$ is

$$\frac{(f(d_u) - 1)d_u}{2m} \times \frac{1}{m} = \frac{(f(d_u) - 1)d_u}{2m^2}$$

We could also choose the random endpoint to have degree d_v. So the total probability of choosing $e' = (u, v)$ is

$$\frac{f(d_u)d_u + f(d_v)d_v - d_u - d_v}{2m^2}.$$

The total probability that (u, v) is swapped out is

$$\frac{1}{m} + \frac{f(d_u)d_u + f(d_v)d_v - d_u - d_v}{2m^2}.$$

□

While this claim may look fairly innocuous, it makes a very strong observation. When edge (u, v) is present, the probability that it is swapped out only depends on the values $d_u, d_v, f(d_u), f(d_v)$. These values are the same regardless of where we are in the Markov chain, because we always preserve the degree distribution! Hence, this satisfies the Markov property, and the probability is independent of the graph itself. But what about the probability that (u, v) becomes an edge?

This is unfortunately not truly Markovian, since it could depend on the remainder of the graph. Nonetheless, this dependence appears to be fairly weak. We can obtain a Markovian estimate for this probability with a simple heuristic. We guess the number of edges incident to vertex v that are also incident to a degree d vertex (for some d). Clearly, this number depends on the graph structure, but we can approximate it based on the JDD. The number of edges from degree d to degree d_v vertices is $\mathbf{J}(d, d_v)$. Of these, the average number of edges incident to a fixed vertex of degree d_v is $\mathbf{J}(d, d_v)/f(d_v)$. We shall approximate the number of edges incident to v with the other endpoint of degree d by this quantity.

Claim. Assume the heuristic approximation above. If at any stage of the Markov chain, the edge (u, v) is not present, the probability that edge (u, v) appears is given by

$$\frac{\mathbf{J}(d_u, d_v)}{2m^2}\left(\frac{d_u}{f(d_v)} + \frac{d_v}{f(d_u)}\right) \tag{2}$$

We omit the proof for this claim due space limitations; however, it is available in [13].

We now focus on the presence or absence of the edge (u, v) as we walk through the Markov chain. Based on the claims above, this can be thought of as a 2-state Markov chain (state 0 meaning no edge, and state 1 meaning presence of edge). The transition matrix $\mathbf{T}_{u,v}$ for this chain is

$$\mathbf{T}_{u,v} = \begin{pmatrix} 1 - \alpha_{u,v} & \alpha_{u,v} \\ \beta_{u,v} & 1 - \beta_{u,v} \end{pmatrix}, \tag{3}$$

where $\alpha_{u,v}$ (resp. $\beta_{u,v}$) is the probability that (u, v) appears (resp. disappears). These probabilities are given by Eq. 2 and Eq. 1 respectively. We will denote this Markov chain by $\mathcal{M}_{u,v}$ and the stationary distribution of it by $\boldsymbol{\pi}_{u,v}$. The

eigenvalues of this transition matrix are 1 and $1-(\alpha_{u,v}+\beta_{u,v})$. The important observation is that the second eigenvalue is at most $1-1/m$, by Eq. 1. The next claim follows from standard Markov chain arguments.

Claim. Set $N = m\ln(1/\epsilon)$. Let the final distribution after running $\mathcal{M}_{u,v}$ for N steps be $\mathbf{p}$. Then $\|\mathbf{p} - \boldsymbol{\pi}_{u,v}\| < \epsilon$.

Observe that $\boldsymbol{\pi}_{u,v}$ represents the probability of presence/absence edge (u,v) in the overall stationary distribution of the entire Markov chain. This claim implies that in $N = m\ln(1/\epsilon)$ steps, we are very close to the stationary distribution for each edge. This bound is independent of the edge. So each edge behaves like it should in the stationary distribution (as far as the overall graph is concerned, we cannot make a stronger claim).

Proof. Denote the unit eigenvectors of $\mathbf{T}$, corresponding to the eigenvalues 1 and $1-(\alpha_{u,v}+\beta_{u,v})$, as $\mathbf{e}_1$ and $\mathbf{e}_2$. Since these $\alpha_{u,v}+\beta_{u,v} > 0$, these form a basis. The initial state can be expressed as $\mathbf{v} = c_1\mathbf{e}_1 + c_2\mathbf{e}_2$. After N applications of the transition matrix we get

$$\mathbf{p} = \mathbf{T}^N\mathbf{v} = c_1\mathbf{T}^N\mathbf{e}_1 + c_2\mathbf{T}^N\mathbf{e}_2 = c_1\mathbf{e}_1 + c_2\left(1-(\alpha_{u,v}+\beta_{u,v})\right)^N\mathbf{e}_2.$$

Since $1-(\alpha_{u,v}+\beta_{u,v}) < 1$, the second term decays with N and $c_1\mathbf{e}_1$ is the stationary distribution $\boldsymbol{\pi}_{u,v}$. For convenience, set $\gamma = \alpha_{u,v}+\beta_{u,v}$. The key observation is that $\gamma \geq 1/m$, by Eq. 1. Hence,

$$N = m\ln(1/\epsilon) \geq \ln(1/\epsilon)/\gamma. \tag{4}$$

We can bound the norm of the difference $\mathbf{p} - \boldsymbol{\pi}_{u,v}$ as

$$\|\mathbf{p}-\boldsymbol{\pi}_{u,v}\| = \|(1-\gamma)^N c_2\mathbf{e}_2\|_2 \leq (1-\gamma)^{\ln(1/\epsilon)/\gamma}c_2\|\mathbf{e}_2\|_2 \leq \exp(-\ln(1/\epsilon)) = \epsilon \tag{5}$$

□

3 Verifying the Edge-By-Edge Convergence

The expression for N, as derived in Section 2.1, is based on a heuristic and has to be verified. In addition, the expression is derived strictly applicable to an edge, and it is unlikely that after N steps, *all* edges will become decorrelated. The residual number of partially correlated edges and their effect on graphical metrics have to be quantified.

Below we construct a purely data-driven, non-parametric test for the independence of a edge, in a Markov chain of graphs. Any specified edge in a Markov chain of graphs traces a binary time-series $\{Z_t\}$, indicating the presence/absence of the edge at each step of the chain. Assume that the chain is very long, i.e., it takes $K \gg N$ steps. The time-series so formed will be auto-correlated, as observed by Stanton and Pinar [8]. However, if the time-series is thinned by a factor k (i.e., we retain every k^{th} element to obtain $\{Z_t^k\}$, the $k-$thinned chain),

the auto-correlation of $\{Z_t^k\}$ will decay and it will begin to resemble independent draws from a distribution. If Eq. 4 is correct, then $k = N$ should yield a time-series that resembles independent draws *more* than a first-order Markov process. Resemblance to either process is established by fitting an independent and first-order Markov process models to the thinned data and computing the log-likelihood. This forms the basis of our test. While this technique has been applied in other domains [14,15], this paper is the first application of this technique to graphs.

Consider the chain $\{Z_t^k\}$. We count the number, x_{ij}, of the $(i,j), i,j \in (0,1)$ transitions in it. x_{ij} are used to populate X, a 2×2 contingency table. Dividing each entry by the length of thinned chain $K/k-1$ provides us with the empirical probabilities p_{ij} of observing an (i,j) transition in $\{Z_t^k\}$. Let $\widehat{p_{ij}}$ and $\widehat{x_{ij}} = (K/k-1)\widehat{p_{ij}}$ be the predictions of the probabilities and expected values of the table entries provided by a model. In such a case, the goodness-of-fit of the model is provided by a likelihood ratio statistic (called the G^2-statistic; Chapter 4.2 in [16]) and a Bayesian Information Criterion (BIC) score

$$G^2 = -2\sum_{i=0}^{i=1}\sum_{i=0}^{i=1} x_{ij} \log\left(\frac{\widehat{x_{ij}}}{x_{ij}}\right), \qquad BIC = G^2 + q\log\left(\frac{K}{k} - 1,\right) \tag{6}$$

where q is the number of parameters in the model used to fit the table data. Typically log-linear models are used for the purpose (Chapter 2.2.3 in [16]); the log-linear models for table entries generated by independent sampling and a first-order Markov process are

$$\log(p_{ij}^{(I)}) = u^{(I)} + u_{1,(i)}^{(I)} + u_{2,(j)}^{(I)} \quad \text{and} \quad \log(p_{ij}^{(M)}) = u^{(M)} + u_{1,(i)}^{(M)} + u_{2,(j)}^{(M)} + u_{12,(ij)}^{(M)}, \tag{7}$$

where superscripts I, M indicate an independent and Markov process respectively. The maximum likelihood estimates (MLE) of the model parameters $(u_{b,(c)}^{(W)})$ are available in closed form (Chapters 2.2.3 and 3.1.2 in [16]; also [13]). We compare the fits of the two models thus: $\Delta BIC = BIC^{(I)} - BIC^{(M)}$. Large BIC values indicate a bad fit. A negative ΔBIC indicates that an independent model fits better than a Markov model.

This test is applied as follows. We construct a thinned binary time-series $\{Z_t^k\}$ for $k = N$ for each of the edges. The ΔBIC is computed and edges with negative ΔBIC are deemed to have become independent after N steps of the Markov chain.

4 Tests with Real Graphs

In this section, we estimate an ϵ for Eq. 4 within the context of a set of graphical metrics. We also verify that N steps of the Markov chain results in independent edge instances. All tests are done with four real networks - the neural network of *C. Elegans* [17] (referred to as "C. Elegans"), the power grid of the Western states of US [17] (called "Power"), co-authorship graph of network science

Table 1. Characteristics of the graphs used in this paper. $(|V|, |E|)$ are the numbers of vertices and edges in the graph and G-R statistic is the Gelman-Rubin diagnostic.

Graph name	$(\|V\|, \|E\|)$	G-R diagnostic
C. Elegans	(297, 4296)	1.05
Netscience	(1461, 5484)	1.02
Power	(4941, 13188)	1.006
soc-Epinions1	(75879, 405740)	1.06

researchers [18] (referred to as "Netscience") and a 75,000 vertex graph of the social network at Epinions.com [19] ("soc-Epinions1"). Their details are in Table 1. The first three were obtained from [20] while the fourth was downloaded from [21]. All the graphs were converted to undirected graphs by symmetrizing the edges.

In Fig. 2 we investigate the impact of ϵ in Eq. 4. We generate 1000 samples by running the Markov chain for $1|E|, 5|E|, 10|E|$ and $15|E|$ steps, corresponding to $\epsilon = 0.37, 6.7 \times 10^{-3}, 4.5 \times 10^{-5}$ and 3.06×10^{-7}. The Markov chain is started using the first three networks listed in Table 1. We calculate the global clustering coefficient, the graph diameter and the maximum eigenvalue for each graph and plot their distributions in Fig. 2. We find that for all three, $\epsilon < 5 \times 10^{-3}$ leads to distributions which are very close. We will proceed with $\epsilon = 4.5 \times 10^{-5}$ i.e., we will mix the Markov chain $10|E|$ times before extracting a sample.

We next calculate the fraction of edges that are deemed independent by the test described in Section 3. We run the Markov chain for $K = 1000N$ steps and construct the binary time-series $\{Z_t\}$ for all the edges. Thinned time-series $\{Z_t^k\}, k = N$ are constructed for $N = \{1, 5, 10, \ldots 30\}|E|$ and each time-series tested for independence. In Fig. 3, we plot the fraction of edges deemed independent as a function of $N/|E|$, for "C. Elegans", "Netscience" and "Power". We see that by $N = 10|E|$, more than 95% of the edges test independent, explaining the convergence of the distributions observed in Fig. 2.

The test of independence described in Section 3 can also be used to construct an ensemble of independent graphs, by running a very long Markov chain, and thinning by $k_* > N$, the thinning factor that renders *all* edges independent. Comparisions with graphs generated using $N = 10|E|$ are in [13], and the distributions are found to be very similar. Thus empirically, we find that a Markov chain, run for $10|E|$ steps generates independent, uniformly distributed graphs.

We now address a large graph (soc-Epinions1), where potentially $|V|^2$ distinct edges might be realized during a Markov chain. While $N = 10|E|$ might render a large fraction of edges independent, there may still be a significant number (*not* fraction) of edges that are still correlated with the starting graph. Since certain graphical metrics, like diameter, can be quite sensitive to edges, we check whether a more stringent N is required for large graphs.

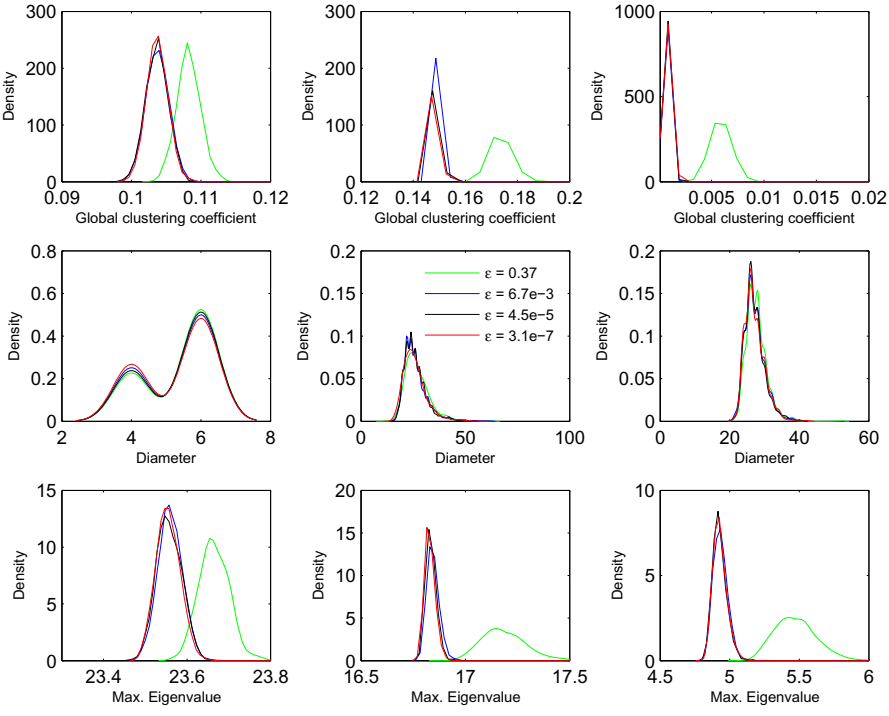


Fig. 2. Plots of the distributions of the global clustering coefficient, the graph diameter and the max eigenvalue of the graph Laplacian for "C. Elegans" (left), "Netscience" (middle) and "Power" (right), evaluated after $1|E|, 5|E|, 10|E|$ and $15|E|$ iterations of the Markov chain (green, blue, black and red lines respectively). The corresponding values of ϵ are in the legend. We see that the distributions converge at $\epsilon \sim 1.0^{-5}$.

We generate an ensemble of 1000 graphs, starting from soc-Epinions1, using $N = 30|E|$. We also run a long Markov chain ($K = 210,000|E|$), and compute the thinning factor k required to render each of the edges independent. Due to the large number of edges realized during the Markov chain, this was calculated for only $0.1|E|$ (40,574) edges, chosen randomly from all the distinct edges that are realized by the Markov chain. In Fig. 4 (left) we plot the distribution of k obtained from the 40,574 sampled edges. We see that most of the k lie between $10|E|$ and $100|E|$; edges with thinning factors outside that range are about two orders of magnitude less abundant. The largest thinning factor identified was $k = 720|E|$. In Fig. 4 (right) we plot the distribution of diameter obtained using $N = 30|E|$, and compare against the distributions obtained from the long run using thinning factors $k = 5N, 9N$, and $13N$. We see small differences in distributions; for practical purposes, $N = 30|E|$ results in a converged distribution.

Finally, we address the question whether the results presented so far are independent of the starting graph. We generate two starting points by marching a Markov chain (initialized by a real network) for $N = 10,000|E|$ steps.

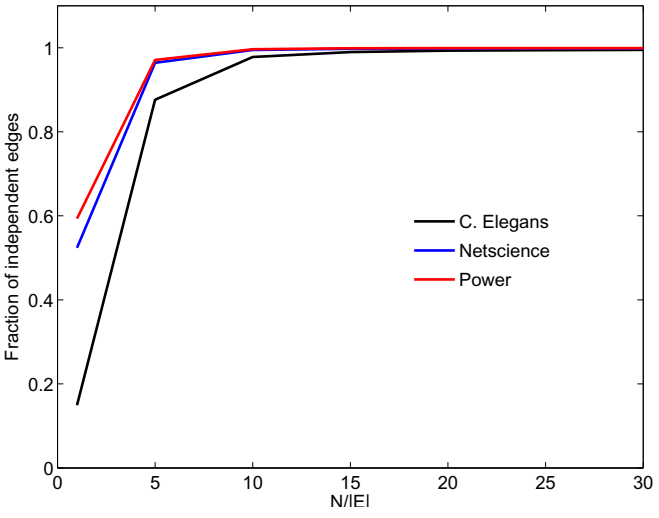


Fig. 3. Fraction of edges testing independent, for "C. Elegans", "Netscience" and "Power" for various values of N. We see that $N = 10|E|$ ensures that at least 95% of the edges become independent.

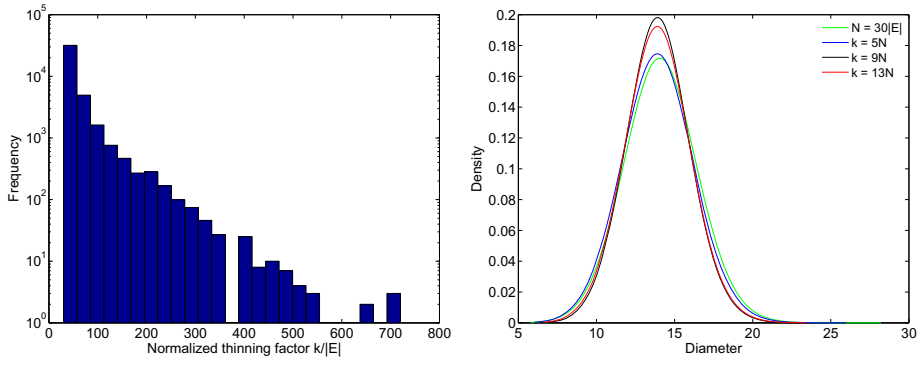


Fig. 4. Left: The normalized thinning factor $k/|E|$ for the soc-Epinions1 graph, as calculated for the 40,574 sampled edges. We see that the most thinning factors are lie in $(10|E|, 100|E|)$. Right: Plot of the graph diameter and distribution generated using $N = 30|E|$ as well as a long Markov chain with thinning factor k equal to various multiples of N. We see that the distributions are very similar.

We initialize 3 concurrent Markov chains with these graphs, and calculate the Gelman-Rubin (G-R) diagnostic [22] using the binary edge time-series. Values of the diagnostic between 1 and 1.1 indicate that the states of the concurrent Markov chain are not dependent on the starting location. We performed this test for all 4 graphs; the corresponding G-R diagnostics are tabulated in Table 1.

5 Conclusions

We have developed a method that allows one to generate a set of independent realizations of graphs with a prescribed joint degree distribution. The graphs are generated using a MCMC approach, employing the algorithm described in [8] as the "rewiring" mechanism. Our method involves running the Markov chain for N steps before extracting a graph realization; the Markov chain is run repeatedly to generate samples. We developed a model (and a closed-form expression) to estimate N that allows the 2-state Markov chain *of an edge* to converge to its stationary distribution. This is a necessary condition for how long a Markov chain *on the space of graphs* has to be run before an independent graph realization can be extracted from it. We find that $10|E| - 30|E|$ steps are sufficient to generate samples of graphs that provide converged distributions of graphical metrics like clustering coefficients, diameter and maximum eigenvalue of the graph Laplacian.

We verified our model (for N) by constructing a non-parametric test for the independence of each edge. It is not dependent on any heuristics or graphical properties. It uses the time-series of the occurrence/non-occurrence of edges, thins them by N and fits a first-order Markov and an independent sampling model to the thinned time-series. Their BICs are used to perform model selection i.e., to decide whether the thinned chain resembles draws from an independent more than a first-order Markov process. The method is not new, but does not seem to have been used in the generation of independent graphs.

Finally, we repeated our tests with concurrent Markov chains, initialized with dispersed starting graphs. We employed the Gelman-Rubin diagnostic to verify that our tests were not being driven by the starting points of the Markov chain.

While this work enables the generation of independent graphs, including large ones, it has only been demonstrated on graphs where the JDD is preserved. Extending our method to the generation of independent graphs when some other graph property is held constant is currently under investigation.

Acknowledgments. We would like to thank Tamara G. Kolda for many helpful discussions.

References

1. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286(5349), 509–512 (1999)
2. Newman, M.E.J.: Assortative mixing in networks. Phys. Rev. Lett. 89, 208701 (2002)
3. Holme, P., Zhao, J.: Exploring the assortativity-clustering space of a network's degree sequence. Phys. Rev. E 75, 046111 (2007)
4. Kannan, R., Tetali, P., Vempala, S.: Simple markov-chain algorithms for generating bipartite graphs and tournaments. Random Struct. Algorithms 14(4), 293–308 (1999)
5. Jerrum, M., Sinclair, A.: Fast uniform generation of regular graphs. Theor. Comput. Sci. 73(1), 91–100 (1990)

6. Jerrum, M., Sinclair, A., Vigoda, E.: A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. J. ACM 51(4), 671–697 (2004)
7. Gkantsidis, C., Mihail, M., Zegura, E.W.: The Markov chain simulation method for generating connected power law random graphs. In: ALENEX, pp. 16–25 (2003)
8. Stanton, I., Pinar, A.: Constructing and sampling graphs with a prescribed joint degree distribution using Markov chains. ACM Journal of Experimental Algorithmics (to appear)
9. Shen-Orr, S.S., Milo, R., Mangan, S., Alon, U.: Network motifs in the transcriptional regulation network of escherichia coli. Nature Genetics 31, 64–68 (2002)
10. Maslov, S., Sneppen, K.: Specificity and stability in topology of protein networks. Science 296(5569), 910–913 (2002)
11. Adams, S.: Dilbert: Random number generator (2001), http://search.dilbert.com/comic/RandomNumberGenerator
12. Sokal, A.: Monte Carlo methods in statistical mechanics: Foundations and new algorithms (1996)
13. Ray, J., Pinar, A., Seshadhri, C.: Are we there yet? when to stop a markov chain while generating random graphs. CoRR abs/1202.3473 (2012)
14. Raftery, A., Lewis, S.M.: Implementing MCMC. In: Gilks, W.R., Richardson, S., Spiegelhalter, D.J. (eds.) Markov Chain Monte Carlo in Practice, pp. 115–130. Chapman and Hall (1996)
15. Raftery, A.E., Lewis, S.M.: How many iterations in the Gibbs sampler? In: Bernardo, J.M., Berger, J.O., Dawid, A.P., Smith, A.F.M. (eds.) Bayesian Statistics, vol. 4, pp. 765–766. Oxford University Press (1992)
16. Bishop, Y.M., Fienberg, S.E., Holland, P.W.: Discrete multivariate analysis: Theory and practice. Springer, New York (2007)
17. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
18. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E 74, 036104 (2006)
19. Richardson, M., Agrawal, R., Domingos, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 351–368. Springer, Heidelberg (2003), doi:10.1007/978-3-540-39718-2_23
20. Newman, M.E.J.: Prof. M. E. J. Newman's collection of graphs at University of Michigan, http://www-personal.umich.edu/~mejn/netdata/
21. Stanford Network Analysis Platform Collection of Graphs: The Epinions social network from the Stanford Network Analysis Platform collection, http://snap.stanford.edu/data/soc-Epinions1.html
22. Gelman, A., Rubin, D.B.: Inference from iterative simulation using multiple sequences. Statistical Science 7, 457–472 (1992)

A Fast Algorithm to Find All High Degree Vertices in Graphs with a Power Law Degree Sequence

Colin Cooper, Tomasz Radzik, and Yiannis Siantos

Department of Informatics, King's College London, UK

Abstract. We develop a fast method for finding all high degree vertices of a connected graph with a power law degree sequence. The method uses a biassed random walk, where the bias is a function of the power law c of the degree sequence.

Let $G(t)$ be a t-vertex graph, with degree sequence power law $c \geq 3$ generated by a generalized preferential attachment process which adds m edges at each step. Let S_a be the set of all vertices of degree at least t^a in $G(t)$. We analyze a biassed random walk which makes transitions along undirected edges $\{x, y\}$ proportional to $(d(x)d(y))^b$, where $d(x)$ is the degree of vertex x and $b > 0$ is a constant parameter. Choosing the parameter $b = (c-1)(c-2)/(2c-3)$, the random walk discovers the set S_a completely in $\widetilde{O}(t^{1-2ab(1-\epsilon)})$ steps with high probability. The error parameter ϵ depends on c, a and m. We use the notation $\tilde{O}(x)$ to mean $O(x \log^k x)$ for some constant $k > 0$.

The cover time of the entire graph $G(t)$ by the biassed walk is $\widetilde{O}(t)$. Thus the expected time to discover all vertices by the biassed walk is not much higher than in the case of a simple random walk $\Theta(t \log t)$.

The standard preferential attachment process generates graphs with power law $c = 3$. Choosing search parameter $b = 2/3$ is appropriate for such graphs. We conduct experimental tests on a preferential attachment graph, and on a sample of the underlying graph of the WWW with power law $c \sim 3$ which support the claimed property.

1 Introduction

Many large networks have a heavy tailed degree sequence. Thus, although the majority of the vertices have constant degree, a very distinct minority have very large degrees. This particular property is the significant defining feature of such graphs. A log-log plot of the degree sequence breaks naturally into three parts. The lower range (small constant degree) where there may be curvature, as the power law approximation is incorrect. The middle range, of large but well represented vertex degrees, which give the characteristic straight line log-log plot of the power law coefficient. In the upper tail, where the sequence is far from concentrated, the plot is a spiky mess. See for example Figure 1 (the degree sequence of a simulated preferential attachment graph with $m = 4$ edges added

A. Bonato and J. Janssen (Eds.): WAW 2012, LNCS 7323, pp. 165–178, 2012.

at each step) and Figure 3 (the degree sequence of the underlying graph of a sample of the WWW). In both cases the x-axis is $a = \log d / \log t$, where d is vertex degree, and t is the size of the graph.

In our work we focus on sampling the higher degree vertices, in both the middle range and upper tail. Our aim is to find *all these vertices*, and we propose a provably efficient method of obtaining those vertices in sub-linear time using a weighted random walk. One reason for finding all the higher degree vertices is that the upper tail is not concentrated, so no sub-sample will be representative. We consider a weighted random walk because, as there are few vertices even in the middle range, a simple random walk may take too long to obtain a statistically significant sample. Coupled with this is the impression that in many networks, for example the WWW, it is the high degree vertices which are important, both as hubs and authorities, and for pagerank calculations.

Previous work on efficient sampling of network characteristics arises in many areas. In the context of search engine design, studies in optimally sampling the URL crawl frontier to rapidly sample (e.g.) high pagerank vertices, based on knowledge of vertex degree in the current sample, can be found in e.g. [3].

Within the random graph community, *traceroute sampling* was used to estimate cumulate degree distributions; and methods of removing the high degree bias from this process were studied in e.g. [1], [10]. Another approach, analysed in [6], is the *jump and crawl* method to find (e.g.) all very high degree vertices. The method uses a mixture of uniform sampling followed by inspection of the neighboring vertices, in a time sub-linear in the network size.

In the context of online social networks, exploration often focused on how to discover the entire network more efficiently. Until recently this was feasible for many real world networks, before they exploded to their current size. It is no longer feasible to get a consistent snapshot of the Facebook network for example. (According to the Facebook statistics page at `www.facebook.com/press/info.php?statistics`, retrieved on 2 June 2011, there were over 500 million active users, and around 36 billion links.)

Methods based on random walks are commonly used for graph searching and crawling. Stutzbach *et al* [14] compare the performance of breadth first search (BFS) with a simple random walk and a Metropolis Hastings random walk on various classes of random graphs as a basis for sampling the degree distribution of the underlying networks. The purpose of the investigation was to sample from dynamic Peer-To-Peer (P2P) networks. In a related study Gjoka *et al* [11] made extensive use of the these methods to collect a sample of Facebook users. As simple random walks are degree biassed they used a re-weighting technique to unbias the sampled degree sequence output by the random walk. This is referred to as a re-weighted random walk in [11]. In both the above cases it was shown the bias could be removed dynamically by using a suitable Metropolis-Hastings random walk.

A simple way to generate a graph with a power law degree sequence is to use the preferential attachment method described by Albert and Barabási [4]. In this model, the graph $G(t) = G(m, t)$ is obtained from $G(t-1)$ by adding a new

vertex v_t with m edges between v_t and $G(t-1)$. The end points of these edges are chosen preferentially, that is to say proportional to the existing degree of vertices in $G(t-1)$. Thus the probability $p(x,t)$ that vertex $x \in G(t-1)$ is chosen as the end point of a given edge is equal to $p(x,t) = d(x,t-1)/(2m(t-1))$, and this choice is made independently for each of the m edges added. A model generated in this way has a power law of $c = 3$ for the degree sequence, irrespective of the number of edges $m \geq 1$ added at each step. For a graph constructed in this way, the expected degree at step t of the vertex added at step s is $\mathbf{E}d(s,t) \sim m(t/s)^{1/2}$.

The preferential attachment model was refined by Bollobas et al [5] who introduced the scale free model to make detailed calculations of degree sequence. The model was generalized by many authors, including the web-graph model of Cooper and Frieze [8]. The web-graph model is more general, and allows the number of edges added at each step to vary, for edges from new vertices to choose their end points preferentially or uniformly at random, and for insertion of edges between existing vertices. By varying these parameters, preferential attachment graphs with degree sequences exhibiting power laws c in the interval $(2, \infty)$ are obtained.

The power law c for preferential attachment graphs and web-graphs can be written explicitly as

$$c = 1 + 1/\eta, \tag{1}$$

where η is the expected proportion of edge end points added preferentially (see [7]). For example in the standard preferential attachment process (e.g. the Barabási and Albert model), $\eta = 1/2$, as each new edge chooses an existing neighbour vertex preferentially; thus explaining the power law of 3 for this model.

In the simplest case, to form $G(t)$, a new vertex v_t is added at each step t with m edges directed towards the existing graph $G(t)$. Each edge chooses its terminal vertex either by preferential attachment or uniformly at random with some probability mixture p or $1-p$. This generates a power law $c \geq 3$. We refer to this generalized process as $G(c,m,t)$. For this example, the parameter $\eta = p/2$ depends on the proportion p of edge end points chosen preferentially (as opposed to uniformly at random). The parameter η in (1) occurs in process models, in the expression for the expected degree of a vertex. Let $d(s,t)$ denote the degree at step t of the vertex v_s added at step s. The expected value of $d(s,t)$ is given by

$$\mathbf{E}d(s,t) \sim m\left(\frac{t}{s}\right)^{\eta}. \tag{2}$$

Thus, in the preferential attachment model of [4], $\mathbf{E}d(s,t) \sim m(t/s)^{1/2}$.

Generalizing this, we consider arbitrary multi-graphs $G(t)$ on t vertices which, have the following properties, which we call *pseudo-preferential*.
(i) When the vertices are relabeled $s = 1, ..., t$ by sorting on vertex degree in descending order, $G(t)$ has a degree sequence which satisfies

$$\left(\frac{t}{s}\right)^{\eta(1-\epsilon)} \leq d(s) \leq \left(\frac{t}{s}\right)^{\eta} \log^2 t, \tag{3}$$

for some $\epsilon > 0$ and $0 < \eta < 1$, and for some range of $s \geq 1$.

(ii) For all vertices s in the sorted order, s has at most m edges to vertices $\sigma \leq s$.

Our particular aim is, given $a > 0$, to find all vertices $v \in V(t)$ of degree $d(v) \geq t^a$. Denote by S_a the set of vertices of $G(t)$ of degree $d(v) \geq t^a$. For the following reason, we will assume $a < \eta$. The maximum degree in (3) is $\widetilde{O}(t^\eta)$, and, from (2), this is also the maximum expected degree in preferential attachment graphs ($\eta = 1/2$) and web-graphs ($0 < \eta < 1$). We use the notation $\widetilde{O}(f(t))$ as shorthand for $O(f(t)\log^k t)$ where t is the size of $G(t)$ and k is a positive constant.

We say a random walk is *seeded* if the walk starts from some vertex s of S. In the context of searching networks such as Facebook, Twitter or the WWW it is not unreasonable to suppose we know *some* high degree vertex without supposing we know all of them. Experimentally, we found the *seeding* condition was not necessary, but a general analysis without this condition would require notions of mixing time and stationarity which our analysis avoids. The following theorem holds for any network with the pseudo-preferential properties given above.

Theorem 1. *Let $G(t)$ be a pseudo-preferential graph with degree sequence satisfying (3). Let $S_a = \{v : d(v) \geq t^a\}$ be connected with diameter $Diam(S_a)$. Let $b = (1-\eta)/(\eta(2-\eta(1-\epsilon))$.*

*A biassed seeded random walk with transition probability along edge $\{x, y\}$ proportional to $(d(x)d(y))^b$, finds all vertices in $G(t)$ of degree at least t^a in $\widetilde{O}(Diam(S_a) \times t^{1-2ab(1-\epsilon)})$ steps, With High Probability (**whp**).*
The cover time of the graph $G(t)$ by this biassed walk is $\widetilde{O}(tDiam(G(t)))$.

In reality the degree sequence (3) of graph $G(t)$ is unknown, but η can be estimated as $\eta = 1/(c-1)$ from the power law c of the degree sequence. Optimistically setting $\epsilon = 0$ then gives a value b for the search algorithm. Its also fair to say that, experimentally, we found putting $b = 1/2$ in the biassed random walk was effective a variety of real networks with a power law degree sequence.

We next give a general result for web-graphs $G(c, m, t)$, which is also valid for related models such as scale free graphs. For the class of graphs $G(c, m, t)$, the lower bound on the degree of vertex s becomes less concentrated as s tends to t, so that the value of ϵ we must choose for our lower bound in (3) increases with s. Thus, as the vertex degree t^a decreases, the upper bound on the algorithm runtime increases in a way which depends on a, c, m. As long as we incorporate this dependence, Theorem 2 says that if we search $G(c, m, t)$ using a random walk with a bias b proportional to the power law c then, (i) we can find all high degree vertices quickly, and (ii) the time to discover all vertices is of about the same order as for a simple random walk.

Theorem 2. *Let $c \geq 3$, and let $m \geq 2$. Let $a < 1/(c-1)$, and let $\epsilon = (1+1/a-c)/m$.*
*Let $b = (c-1)(c-2)/(2c-3)$. For $c \geq 3$, **whp** we can find all vertices in $G(c, m, t)$ of degree at least t^a in $\widetilde{O}(t^{1-2ab(1-\epsilon)})$ steps, using a biassed seeded random walk with transition probability along edge $\{x, y\}$ proportional to $(d(x)d(y))^b$.*
The cover time of the graph $G(c, m, t)$ by this biassed walk is $\widetilde{O}(t)$.

The maximum degree of $G(c,m,t)$ is $\widetilde{O}(t^{\eta})$ **whp**, where $\eta = 1/(c-1)$ which explains the bound on a given above. Using this, a t^{1-2ab} run time can be repackaged as follows. Let $a = \theta\eta$ for $0 < \theta < 1$, then $2ab = \theta(1 - 1/(2c-3))$.

2 Properties of the Web-Graph Process

The actual value of $d(s,t)$ is not concentrated around $\mathbf{E}d(s,t)$ in the lower tail, but the following inequality is adequate for our proof.

Lemma 1. *Given $G(c,m,t)$ and a,ϵ and suppose $m > (1/\epsilon)(1/a - 1/\eta)$.*

With high probability for all vertices s, such that $\mathbf{E}d(s,t) \geq t^a$, we have that $d(s,t) \geq \left(\frac{t}{s}\right)^{\eta(1-\epsilon)}$. For all $s \geq \log^2 t$, $d(s,t) \leq \left(\frac{t}{s}\right)^{\eta}\log^2 t$.

For proof of Lemma 1 see Appendix. We also need lower tail concentration for large sets of vertices.

Lemma 2. *Let $d([s],t)$ denote the total degree at step $t \geq s$ of the set $[s] = \{1,...,s\}$. Let $K > 1$. Then*

$$\mathbf{Pr}\left(d([s],t) \leq \frac{2ms}{K}\left(\frac{t}{s}\right)^{\eta}\right) = O(s^{-mK}).$$

The upshot of this, is that all vertices added after step $v = s\log^{2/\eta+1} t$ have degree $d(v,t) = o\left((t/s)^{\eta}\right)$ **whp**. This observation forms the basis of our sub-linear algorithm. For proof of Lemma 2 see Appendix.

Another piece of the puzzle we will need, is that **whp** web-graphs have diameter

$$\mathrm{Diam}(G(c,m,t)) = O(\log t) \tag{4}$$

Crude proofs of this can be made for the web-graph model based on expansion properties of the graph. For example, in the preferential attachment graph ($\eta = 1/2, c = 3$) when vertex t is added to $G(m,t)$ the probability that t does not select at least one neighbour in $G(t/2)$ is at most

$$\left(1 - \frac{2m(t/2)}{2mt}\right)^{m} = \left(\frac{1}{2}\right)^{m}.$$

Thus $\mathrm{Diam}(G(m,t)) = O(\log t)$ by a 'tracing backwards stochastically' argument.

3 Biassed Random Walks

Let $G = (V,E)$ be a connected undirected graph. A *random walk* W_u, $u \in V$, on G is a Markov chain $X_0 = u, X_1, \ldots, X_t, \ldots$ on the vertices V associated to a particle that moves from vertex to vertex according to a transition rule. The

probability of a transition from vertex i to vertex j is $p(i,j)$ if $\{i,j\} \in E$, and 0 otherwise.

Let $d(v) = d(v,t)$ be the degree of vertex $v \in G(t)$, and let $N(v)$ denote the neighbours of v in this graph. The basis of our algorithm is a degree-biassed random walk, with transition probability $p(u,v)$ given by

$$p(u,v) = \frac{(d(v))^b}{\sum_{w \in N(u)} (d(w))^b}, \tag{5}$$

where $b > 0$ constant. The value of b we will choose in our proof is optimized to depend on η. Thus for Theorem 2, using (1), the value of b can be expressed directly as a function of the degree sequence power law c.

The easiest way to reason about biassed random walks, is to give each edge e a weight $w(e)$, so that transitions along edges are made proportional to this weight. In the case above the weight of the edge $e = (u,v)$ is given by $w(e) = (d(u)d(v))^b$ so that the transition probability (5) is now written as

$$p(u,v) = \frac{(d(u)d(v))^b}{\sum_{w \in N(u)} (d(u)d(w))^b}. \tag{6}$$

The inspiration for the degree biassed walk above, comes from the β-walks of Ikeda, Kubo, Okumoto and Yamashita [12] which use an edge weight $w(x,y) = 1/(d(x)d(y))^\beta$ to favor low degree vertices. When $\beta = 1/2$ this gives an improved worst case bound of $O(n^2 \log n)$ for the cover time of connected n-vertex graphs.

We next note some facts about weighted random walks, which can be found in Aldous and Fill [2] or Lovasz [13]. The weight $w(e)$ of an edge e has the meaning of conductance in electrical networks, and the resistance $r(e)$ of e is given by $r(e) = 1/w(e)$. The commute time $K(u,v)$ between vertices u and v, is the expected number of steps taken to travel from u to v and back to u. The commute time for a weighted walk is given by

$$K(u,v) = w(G) R_{\mathit{eff}}(u,v). \tag{7}$$

Here $w(G) = 2\sum_{e \in E} w(e)$ and $R_{\mathit{eff}}(u,v)$ is the effective resistance between u and v, when G is taken as an electrical network with edge e having resistance $r(e)$. For our proof we do not need to calculate $R_{\mathit{eff}}(u,v)$ very precisely, but rather note that if uPv is any path between u and v then

$$R_{\mathit{eff}}(u,v) \leq \sum_{e \in uPv} r(e).$$

For $u \in V$, and a subset of vertices $S \subseteq V$, let $C_u(S)$ be the expected time taken for W_u to visit every vertex of G. The *cover time* C_S of S is defined as $C_S = \max_{u \in V} C_u(S)$. We define a walk as *seeded* if it starts in S. The *seeded cover time* ${C_S}^*$ of S as ${C_S}^* = \max_{u \in S} C_u(S)$. For a random walk starting in a set S, the cover time of S satisfies the following Matthews bound

$$C_S^* \leq \max_{u,v \in S} H(u,v) \log |S|. \tag{8}$$

For $u \neq v$, the variable $H(u,v)$ is the expected time to reach v starting from u (the hitting time). The commute time $K(u,v)$ is given by $K(u,v) = H(u,v) + H(v,u)$, so $K(u,v) > H(u,v)$.

4 Proof of Theorems 1 and 2

We apply the Matthews bound (8). Clearly $\log|S_a| \leq \log t$. It remains to find

$$\max_{u,v\in S} H(u,v) \leq \max_{u,v\in S} K(u,v).$$

To calculate $K(u,v)$ in (7), we first need to bound $w(G)$

Lemma 3. *By choosing*

$$b = \frac{1-\eta}{\eta(2-\eta(1-\epsilon'))},$$

where $\epsilon' = \epsilon$ *for Theorem 1, and* $\epsilon' = 0$ *for Theorem 2, it follows that* $w(G) = O(t\log^{4b+1} t) = \widetilde{O}(t)$

Proof
We define a graph G^* on vertices $1,2,\ldots,t$ which has the same degree sequence as graph G, and is built in a similar iterative process: for each $v = t_0, t_0+1,\ldots,t$, add m edges from vertex v to some earlier vertices. In graph G, edges are selected according to a random preferential process, while in graph G^* according to the deterministic process which greedily fills the in-degrees of vertices, giving preference to the older vertices. In both graphs, if (x,y) is a directed edge, then $y < x$ (the edges point from x towards the earlier vertex y).

Assume $b > 0$ and define

$$\bar{d}(v) = \left(\frac{t}{v}\right)^{\eta},$$
$$\bar{w}(G) = 2\sum_{\{x,y\}\in E(G)} \left(\bar{d}(x)\bar{d}(y)\right)^b \;\geq\; w(G)\log^{-4b} t.$$

Graph G^* is obtained from G by repeatedly swapping edges. Whenever there is a pair of edges (x,y), (u,v) such that $x < u$ but $y > v$, then replace them with edges (x,v) and (u,y). If $A > B$ and $C > D$ then $(A-B)(C-D) > 0$ so $AC + BD > AD + BC$. Thus each swap increases $\bar{w}(G)$ because

$$(\bar{d}(x))^b > (\bar{d}(u))^b \quad \text{and} \quad (\bar{d}(y))^b < (\bar{d}(v))^b$$

implies

$$(\bar{d}(x))^b(\bar{d}(v))^b + (\bar{d}(u))^b(\bar{d}(y))^b \;>\; (\bar{d}(x))^b(\bar{d}(y))^b + (\bar{d}(u))^b(\bar{d}(v))^b.$$

Therefore, $\bar{w}(G^*) \geq \bar{w}(G)$. By construction, a vertex v in G^* has incoming edges originating from vertices $\text{first}(v), \text{first}(v)+1, \ldots, \text{last}(v)$. Thus we have

$$\begin{aligned}
\bar{w}(G^*) &= 2 \sum_{\{y,x\} \in E(G^*)} \left(\bar{d}(x)\bar{d}(y)\right)^b \\
&= 2 \sum_{x=1}^{t} \sum_{y=\text{first}(x)}^{\text{last}(x)} \left(\bar{d}(x)\bar{d}(y)\right)^b \\
&\leq 2 \sum_{x=1}^{t} d(x) \left(\bar{d}(x)\bar{d}(\text{first}(x))\right)^b \\
&\leq 2 \sum_{x=1}^{t} \left(\bar{d}(x)\right)^{1+b} \left(\bar{d}(\text{first}(x))\right)^b . \qquad (9)
\end{aligned}$$

Now we calculate $\text{first}(x)$. The $m(\text{first}(x)-1)$ edges outgoing from vertices $1, 2, \ldots, \text{first}(x)-1$ fully fill the in-degrees of vertices $1, 2, \ldots, x-1$, so

$$m \cdot \text{first}(x) = 1 + \sum_{z=1}^{x-1} (d(z) - m).$$

Let C be some generic constant whose value can vary. For Theorem 1 choosing $\epsilon' = \epsilon$, (deterministic case),

$$\sum_{z=1}^{x-1} d(z) \geq \sum_{z=1}^{x-1} \left(\frac{t}{z}\right)^{\eta(1-\epsilon)} = C t^{\eta(1-\epsilon')} x^{1-\eta(1-\epsilon')}.$$

For Theorem 2 (web-graph case), choosing $\epsilon' = 0$ we have from Lemma 2 that

$$\sum_{z=1}^{x-1} d(z) \geq mx \left(\frac{t}{x}\right)^{\eta}.$$

Thus

$$\bar{d}(\text{first}(x)) \leq C \left(\frac{t}{t^{\eta(1-\epsilon')} x^{1-\eta(1-\epsilon')}}\right)^{\eta} = C \left(\frac{t}{x}\right)^{\eta(1-\eta(1-\epsilon'))}. \qquad (10)$$

Using (10) in (9), we get

$$\begin{aligned}
\bar{w}(G^*) &\leq C \sum_{x=1}^{t} \left(\frac{t}{x}\right)^{\eta(1+b)} \left(\frac{t}{x}\right)^{b\eta(1-\eta(1-\epsilon'))} \\
&= C \sum_{x=1}^{t} \left(\frac{t}{x}\right)^{\eta(1+b(2-\eta(1-\epsilon')))} . \qquad (11)
\end{aligned}$$

Choosing

$$\eta(1 + b(2 - \eta(1-\epsilon'))) = 1, \qquad (12)$$

the sum in (11) is $O(t \log t)$ and we have

$$w(G) \le \log^{4b} \bar{w}(G) \le \log^{4b} \bar{w}(G^*) = O(t \log^{4b+1} t). \tag{13}$$

□

Details Specific to Theorem 1. The set S_a is connected with diameter $\mathrm{Diam}(S_a)$ is as specified. Let $\Delta(a) = \mathrm{Diam}(S_a)$, then for any $u, v \in S_a$ there is a path uPv of length $O(\Delta(a))$ from u to v in $G(t)$ contained in S_a, and thus consisting of vertices w of degree $d(w,t) \ge (t/s)^{\eta(1-\epsilon)} = d^*$. Thus all edges of this path have resistance at most $1/(d(x)d(y))^b \le 1/(d^*)^{2b}$.

Details Specific to Theorem 2. Suppose we want to find all vertices of degree at least t^a for some $a > 0$ in $G(t) \equiv G(c, m, t)$. Let $S_a = \{v : d(v,t) \ge t^a\}$. Recall that $G(t)$ is generated by a process of attaching v_t to $G(t-1)$. At what steps were the vertices $v \in S_a$ added to $G(t)$? The expected degree of v at step t is given by (2) i.e. $\mathbf{E}d(v,t) = (1 + o(1))m(t/v)^\eta$. This function is monotone decreasing with increasing v. Let σ be given by

$$t^a = \left(\frac{t}{\sigma}\right)^\eta \qquad \text{which implies} \qquad \sigma = t^{1-a/\eta}. \tag{14}$$

Let $s = \sigma \cdot \log^{2/\eta+1} t$, then using (3) all vertices added at steps $w \ge s$ have $d(w,t) = o(t^a)$. On the other hand, using (3) again, all vertices v added at steps $1, ..., s$ have degree $d(v,t) \ge (t/s)^{\eta(1-\epsilon)}$.

For Theorem 2 let $\Delta(a) = \mathrm{Diam}(G(s))$ where s is as defined above. Because $\mathrm{Diam}(G(s)) = O(\log s)$, (see (4)), we know that for any $u, v \in S_a$ there is a path uPv of length $O(\log t)$ from u to v in $G(t)$ contained in $G(s)$, and thus consisting of vertices w of degree $d(w,t) \ge (t/s)^{\eta(1-\epsilon)} = d^*$. Thus all edges of this path have resistance at most $1/(d(x)d(y))^b \le 1/(d^*)^{2b}$.

Proof of Theorems 1 and 2. From (3), d^* satisfies

$$d^* \ge \left(\frac{t}{t^{1-a/\eta} \log^{1+2/\eta} t}\right)^{\eta(1-\epsilon)} \ge \frac{t^{a(1-\epsilon)}}{\log^3 t}.$$

By the discussion above,

$$R_{\mathit{eff}}(u,v) \le \sum_{e \in uPv} r(e) = O\left(\frac{\Delta(a)}{d^*}\right).$$

Using (7), and the value of d^*, we have

$$K(u,v) \le K^* = \widetilde{O}(\Delta(a) t^{1-2ba(1-\epsilon)}).$$

The bound in Theorem 2 on finding all vertices of degree at least t^a is now obtained as follows. The Matthews bound (8) gives the (expected) cover time $C^*_{S_a} = O(K^* \log t)$. Apply the Markov inequality ($\mathbf{Pr}(X > A \cdot \mathbf{E}X) \le 1/A$), with

$\mathbf{E}X = C^*_{S_a}$, and $A = \log t$ to give a **whp** result, that all vertices of degree at least t^a can be found in time

$$T(a) = \widetilde{O}(t^{1-2ba(1-\epsilon)}).$$

For preferential attachment graphs $\eta = 1/2$, and (12) gives $b = 2/3$, and the time $T(a)$ is

$$\widetilde{O}(t^{1-(4/3)a(1-\epsilon)}).$$

Finally we establish the cover time of the graph $G(t)$. This is done by using (8) with $S = V(t)$ the vertex set of $G(t)$, i.e.

$$C_{V(t)} \leq \max_{u,v \in V(t)} H(u,v) \log t. \tag{15}$$

We bound $H(u,v)$ by (7) as usual. The resistance $r(e)$ of any edge $e = \{x,y\}$ is

$$r(e) = \frac{1}{(d(x)d(y))^b} \leq \frac{1}{m^{2b}} = O(1).$$

Let the diameter of $G(t)$ be $\mathrm{Diam}(G)$ which is specified for Theorem 1 and is $O(\log t)$ (**whp**) for Theorem 2. Thus $R_{\mathit{eff}}(u,v) = O(\mathrm{Diam}(G))$, since the effective resistance between u and v is at most the resistance of a shortest path between u and v. This and (13) give $K(u,v) = \widetilde{O}(t\mathrm{Diam}(G))$. Thus the cover time of the graph $G(t)$ is $\widetilde{O}(t\mathrm{Diam}(G))$.

5 Experimental Results

Theorem 2 gives an encouraging upper bound of the order of around $t^{1-(4/3)a}$ for a biassed random walk to the cover all vertices of degree at least t^a in the t-vertex preferential attachment graph $G(3,m,t)$. Our experiments, summarized in Figure 2, suggest that the actual bound is stronger than this. The experiments were made on $G(m,t)$ with $m = 4$, and $t = 5 \times 10^6$ vertices. The degree distribution of this graphs is given in Figure 1, with both axes in logarithmic scale. More precisely, the x-axis is the exponent a in the degree $d = t^a$, i.e. $x = \log d / \log t$, while the y-axis is the frequency of the vertices of degree t^a.

In Figure 2, plot SRW shows the average cover time $\tau(a)$ of all vertices of degree at least t^a by the simple random walk (the uniform transition probabilities). Plot WRW shows the average cover times by the biassed random walk with $b = 1/2$, and $b = 2/3$. The plots are an average of 9 runs (each) of the random walks. Both axes are in logarithmic scale. The y-axis is $y = (\log \tau(a))/\log t$. There are also three reference lines drawn in Figure 2. These lines have slopes $-a$, $-4a/3$ and $-2a$, and are included for visual inspection only. To calculate the speed up, given $x = a$, read off the $y(a)$-values y_S, y_W. The speed up is $t^{y_S - y_W}$, where $t = 5 \times 10^6$. In the upper tail the weighted walks is about 10 times faster. Curiously, the improvement does not seem sensitive to the precise value of b.

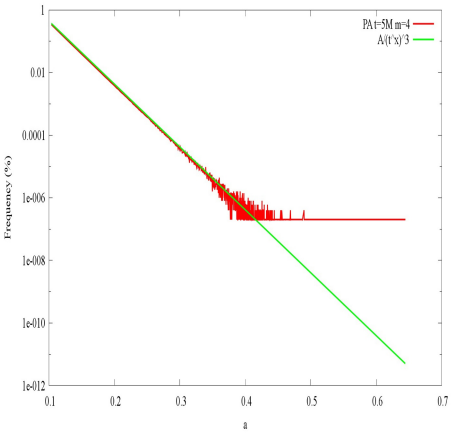


Fig. 1. Degree distribution of a realization of $G(c, m, t)$, $c = 3, m = 4, t = 5 \times 10^6$

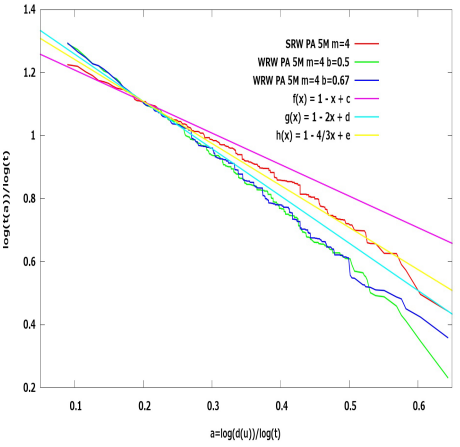


Fig. 2. Cover time of vertices of degree at least t^a in $G(3, 4, 5 \times 10^6)$ as a function of a

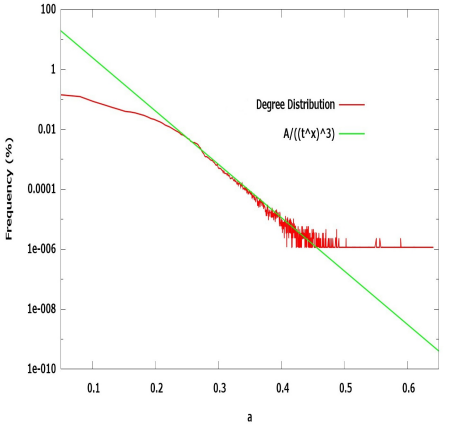


Fig. 3. Degree distribution of sample of size 8.7×10^5 of G_W, the underlying graph of the WWW

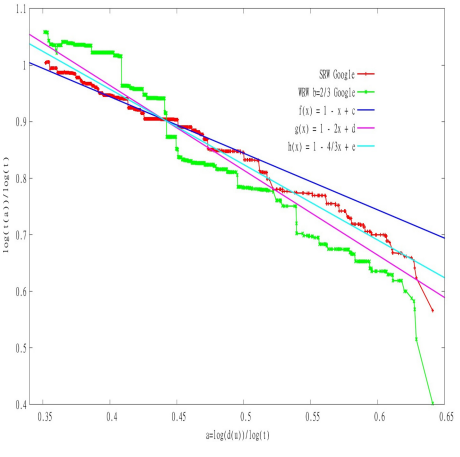


Fig. 4. Cover time of vertices of degree at least t^a in G_W as a function of a

The cover time C_G of a simple random walk on $G(m, t)$ is known and has value $C_G \sim (2m/(m-1))t \log t$, see [9]. The intercept of the y-axis predicted by this is $y_C = \log C_G / \log t$, which when $m = 4$ and $t = 5 \times 10^6$ is $y_C = 1.29$ This agrees well with the experimental intercept of 1.24, and helps confirm the accuracy of our simulations.

Our experimental results for Theorem 1 are less clear cut, but still encouraging. Figure 3 gives the degree distribution of the underlying graph of the WWW, on $t = 8.7 \times 10^5$ vertices obtained from `http://snap.stanford.edu/data/web-Google.html`. The power law exponent

is approximately $c = 3$, and it was crawled using a weight of $b = 2/3$. Figure 4, shows the results obtained by averaging 25 runs of the simple and weighted random walks. The weighted walk is generally about 4 times faster for $a > 0.43$.

References

1. Achlioptas, D., Clauset, A., Kempe, D., Moore, C.: On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. J. ACM 56(4) (2009)
2. Aldous, D., Fill, J.A.: Reversible Markov chains and random walks on graphs (1995), http://stat-www.berkeley.edu/pub/users/aldous/RWG/book.html
3. Baeza-Yates, R., Castillo, C., Marin, M., Rodriguez, A.: Crawling a country: Better strategies than breadth-first for web page ordering. In: Proc. 14th International Conference on World Wide Web, pp. 864–872. ACM Press (2005)
4. Barabási, A., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
5. Bollobás, B., Riordan, O., Spencer, J., Tusnády, G.: The degree sequence of a scale-free random graph process. Random Structures and Algorithms 18, 279–290 (2001)
6. Brautbar, M., Kearns, M.: Local algorithms for finding interesting individuals in large networks. In: Proceedings of ICS 2010, pp. 188–199 (2010)
7. Cooper, C.: The age specific degree distribution of web-graphs. Combinatorics Probability and Computing 15, 637–661 (2006)
8. Cooper, C., Frieze, A.: A general model web graphs. Random Structures and Algorithms 22(3), 311–335 (2003)
9. Cooper, C., Frieze, A.: The cover time of the preferential attachment graphs. Journal of Combinatorial Theory B(97), 269–290 (2007)
10. Flaxman, A.D., Vera, J.: Bias Reduction in Traceroute Sampling – Towards a More Accurate Map of the Internet. In: Bonato, A., Chung, F.R.K. (eds.) WAW 2007. LNCS, vol. 4863, pp. 1–15. Springer, Heidelberg (2007)
11. Gjoka, M., Kurant, M., Butts, C.T., Markopoulou, A.: A walk in Facebook: Uniform sampling of users in online social networks. CoRR, abs/0906.0060 (2009)
12. Ikeda, S., Kubo, I., Okumoto, N., Yamashita, M.: Impact of Local Topological Information on Random Walks on Finite Graphs. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, Springer, Heidelberg (2003)
13. Lovász, L.: Random walks on graphs: A survey. Bolyai Society Mathematical Studies 2, 353–397 (1996)
14. Stutzbach, D., Rejaie, R., Duffield, N.G., Sen, S., Willinger, W.: On unbiased sampling for unstructured peer-to-peer networks. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement IMC 2006, pp. 27–40 (2006)

Appendix

Lemma 1 Given $G(c, m, t)$ and a, ϵ and suppose $m > (1/\epsilon)(1/a - 1/\eta)$.

With high probability for all vertices s, such that $\mathbf{E}d(s,t) \geq t^a$, we have that $d(s,t) \geq \left(\frac{t}{s}\right)^{\eta(1-\epsilon)}$. For all $s \geq \log^2 t$, $d(s,t) \leq \left(\frac{t}{s}\right)^{\eta} \log^2 t$.

Proof . The upper bound on $d(s,t)$ is given in [7], as is the following degree distribution. For $m \geq 2$, the distribution of $d(s,t)$ is given by

$$\mathbf{Pr}(d(s,t) = m + \ell \mid d(s,s) = m) \leq C\binom{m+\ell-1}{\ell}\left(\frac{s}{t}\right)^{\eta m}\left(1-\left(\frac{s}{t}\right)^{\eta}\right)^{\ell}.$$

Thus, crudely

$$\mathbf{Pr}(d(s,t) \leq \ell) \leq C\ell^m\left(\frac{s}{t}\right)^{\eta m}.$$

Inserting $\ell = \left(\frac{t}{s}\right)^{\eta(1-\epsilon)}$, and choosing $s = t^{1-a/\eta}$, we find the expected number of vertices $1 \leq v \leq s$ not satisfying the lower bound is of order

$$s\left(\frac{s}{t}\right)^{m\epsilon\eta} = t^{(1-a/\eta)(1+m\epsilon\eta)} = o(1),$$

provided

$$m > \frac{1}{\epsilon}\left(\frac{1}{a} - \frac{1}{\eta}\right).$$

□

Lemma 2. Let $d([s],t)$ denote degree of $[s] = \{1,...,s\}$ at step t. Let $K > 1$. Then

$$\mathbf{Pr}\left(d([s],t) \leq \frac{2ms}{K}\left(\frac{t}{s}\right)^{\eta}\right) = O(s^{-mK}).$$

Proof . We give the proof for $\eta = 1/2$ (preferential attachment), the general proof is similar.

Let $Z_t = d([s],t)$. Then $Z_t = X_t + Z_{t-1}$ where $Z_s = 2ms$ and $X_t \sim Bin(m, Z_{t-1}/(2m(t-1)))$. Also, $\mathbf{E}Z_t \sim 2ms(t/s)^{1/2}$. Given $h, c_t, A > 0$,

$$\mathbf{Pr}(Z_t < A) = \mathbf{Pr}(e^{-hZ_t/c_t} > e^{-hA/c_t}).$$

Let $p = Z_{t-1}/(2m(t-1))$, then

$$\begin{aligned}\mathbf{E}(e^{-hX_t/c_t}) =& (1-p+pe^{-h/c_t})^m \\ \leq& e^{-\frac{h}{c_t}(1-h/c_t)\frac{Z_{t-1}}{2(t-1)}},\end{aligned}$$

by using $e^{-x} \leq 1 - x + x^2$. Let $c_s = 1, c_t = (1 + 1/(2(t-1)))c_{t-1}$ so that $c_t \sim (t/s)^{1/2}$. We will choose $h = o(1)$ (see below). Iterating the expression $Z_t = X_t + Z_{t-1}$, gives

$$\begin{aligned}\mathbf{E}(e^{-hZ_t/c_t}) \leq& e^{-h\frac{Z_{t-1}}{c_{t-1}}\frac{1+(1-h/c_t)/(2(t-1))}{1+1/(2(t-1))}} \\ =& e^{-h'Z_{t-1}/c_{t-1}},\end{aligned}$$

where

$$h(1 - O(h/tc_t)) \leq h' \leq h,$$

and

$$\mathbf{E}(e^{-hZ_s/c_s}) = e^{-h2ms}.$$

All in all,

$$\mathbf{E}(e^{-hZ_t/c_t}) \leq \mathbf{E}\left(e^{-h\frac{Z_s}{c_s}\prod_{j=s}^{t-1}(1-O(h/(jc_j)))}\right) = e^{-h2ms(1-O(h))}.$$

Choosing $A = \mathbf{E}Z_t/K'$ and applying the Markov inequality that $\mathbf{Pr}(Y \geq A) \leq \mathbf{E}(Y)/A$ with $Y = e^{-hZ_t/c_t}$, we have

$$\mathbf{Pr}(Z_t \leq \mathbf{E}Z_t/K') \leq e^{-h2ms(1-1/K'-O(h))} = O(s^{-mK}),$$

on choosing $h = (K \log s)/s = o(1)$. □

Author Index