Inferring Temporal Information from a Snapshot of a Dynamic Network

Jithin K. Sreedharan[1,+], Abram Magner[1,+], Ananth Grama[1], and Wojciech Szpankowski[1,*]

[1]Center for Science of Information, Dept. of Computer Science, Purdue University, USA
[*]To whom correspondence should be addressed. E-mail: szpan@purdue.edu
[+]These authors contributed equally to this work

ABSTRACT

The problem of reverse-engineering the evolution of a dynamic network, known broadly as network archaeology, is one of profound importance in diverse application domains. In analysis of infection spread, it reveals the spatial and temporal processes underlying infection. In analysis of biomolecular interaction networks (e.g., protein interaction networks), it reveals early molecules that are known to be differentially implicated in diseases. In economic networks, it reveals flow of capital and associated actors. Beyond these recognized applications, it provides analytical substrates for novel studies – for instance, on the structural and functional evolution of the human brain connectome. In this paper, we model, formulate, and rigorously analyze the problem of inferring the arrival order of nodes in a dynamic network from a single snapshot. We derive limits on solutions to the problem, present methods that approach this limit, and demonstrate the methods on a range of applications, from inferring the evolution of the human brain connectome to conventional citation and social networks, where ground truth is known.

Introduction

Complex systems are comprised of interacting entities; e.g., cellular processes are comprised of interacting genes, proteins, and other biomolecules; social systems, of individuals and organizations; and economic systems, of financial entities. These systems are modeled as networks, with nodes representing entities and edges representing their interactions. Typical systems continually evolve to optimize various criteria, including function (e.g., flow of information in social networks, evolution of brain connectomes to specialize function), structure (e.g., evolution of social network structures to minimize sociological stress while maximizing information flow), and survivability (e.g., redundant pathways in genic interactions as evidenced by synthetic lethality screens). Recent results have also demonstrated advantages of dynamic networks in achieving quicker controllability[1]. Effective analysis of dynamic networks provides strong insights into the structure, function, and processes driving system evolution.

The problem of inferring the evolution of a dynamic network is of considerable significance: in a network of financial transactions, the arrival order of nodes tracks the flow of capital. In mapping spread of infectious diseases, node arrival order allows one to identify early patients, yielding clues to genetic origin, evolution, and mechanisms of transmission. In networks of biochemical interactions (e.g., protein interaction networks[2]) one can identify early biomolecules that are known to be differentially implicated in diseases[3]. Recently, strategic seeding and spread of (mis)information in online social networks like Twitter and Facebook has been hypothesized to create strong biases in opinions, even skewing electoral outcomes. Identifying sources and mechanisms of information transfer enables us to quarantine sources in a timely manner and to control spread.

Our contributions. We model and formulate the problem of recovering the temporal order of nodes in general graph models. Focusing on preferential attachment graphs and on deriving fundamental limits on inference of temporal order, we show that there exists no estimator for recovering temporal arrival order with high probability, owing to inherent symmetries in networks. Motivated by this negative result, we relax the formulation to admit a *partial order* on nodes. In doing so, we allow the estimator to make fewer vertex pair order inferences, in exchange for higher precision (e.g., by grouping nodes and finding the order only between groups but not within groups). We refer to the fraction of all node pairs that is comparable by a partial order (in terms of the arrival order) as the partial order density. We cast the partial order inference problem as a rational linear integer program, which allows us to present detailed analytical results on the achievable limits in terms of the tradeoff between expected precision and partial order density. To solve the optimization problem, we need to count the number of linear extensions of the partial order, which is known to be #P-complete[4]. There exists a fully-polynomial-time approximation algorithm that approximates the optimal solution to arbitrarily small relative error. However, in view of its significant computational cost, we propose a Markov chain Monte Carlo technique that achieves faster convergence in practice. We introduce and analyze, both theoretically and empirically, efficient estimators: our first estimator is optimal in the sense that it yields *perfect* precision. It infers all vertex order relations that hold with probability one. However, we find such relations to be asymptotically small compared to the total number of correct pairs. This motivates our investigation of other algorithms (the PEELING and PEELING+ algorithms), which sacrifice some precision in order to achieve higher density.

Experimental evaluation, on both synthetic and real-world datasets (network data of citations (ArXiv), collaborations (DBLP), hyperlinks (Wikipedia) and social connections (Facebook and SMS)), demonstrates the robustness of our methods to variations from the preferential attachment model. We also present a novel application of our method to the analysis of the human brain connectome to identify regions of "early" and "late" development. Our results reveal novel insights into the structural and functional evolution of the brain.

Prior works. The problem of inferring the sequence of node arrivals from a given network snapshot is highly complex, both analytically and methodologically, and has been little studied in prior literature. The works of Navlakha and Kingsford[5] and Young et al.[6] are the ones closest to ours. Navlakha and Kingsford[5] formulates the problem as a maximum *a posteriori* estimation problem and develops a greedy algorithm for different graph models. Such a study can be translated to our maximum likelihood approach and we prove later that this leads to very large number of equiprobable solutions in the case of preferential attachment graphs. Young et al.[6] studies the phase transition of recoverability via numerical experiments in the case of a non-linear preferential attachment graph in the Bayesian framework, with respect to the non-linear exponent of degree. Such a phase transition can be formally justified with the theoretical results in the Supplementary Material of this paper. Some prior results focus on variants of the problem of finding the oldest node in a graph[7,8]. The results of Bubeck et al.[7] are only applicable to trees, thus severely limiting their application scope. Our proposed methods target general graphs and seek node orders beyond identifying the oldest node. Frieze et al.[8] study the problem of identifying the oldest node in preferential attachment graphs using a local exploratory process with the assumption that the time index of a node can be retrieved once it is sampled. A related problem of detecting information sources in epidemic networks has been studied by Shah et al.[9] and Zhu et al.[10] for the Susceptible-Infected model. We first formulated the node arrival order inference problem and presented some preliminary results in Ref 11.

Results

Let G be a graph of n vertices corresponding to a snapshot of a growing network, generated by a dynamic graph model. Without loss of generality, we count time in units of vertex additions. Since G has n vertices, we say that this is the snapshot at time n. We label vertices in their arrival order, $[n] = \{1,\ldots,n\}$, where node j is the jth node to arrive. Note that these vertex labels are not known to us. Instead, the vertices are randomly relabeled according to a permutation π drawn uniformly at random from the set of permutations on n letters S_n, and we are given the graph $H := \pi(G)$. Our goal is to infer the arrival order of vertices in graph G from observed graph H, i.e., to find the inverse permutation π^{-1}, which reveals the true arrival order. See Figure 1 for an illustration of our approach and an application on inferring the evolutionary order of prominent human brain regions. We provide further analyses of this result later in the paper.

We consider a general scenario in which we do not restrict our analysis to inference of a total order. Rather, we consider an estimator ϕ that outputs *partial orders* on the set of vertices (see Figure 1A for an example). For a partial order σ, a relation $u <_\sigma v$ defined on vertices u and v means that vertex u's label is less than that of vertex v in the partial order σ. We say that an ordered pair of vertices (u,v) in $\pi(G)$ satisfying $u <_\sigma v$ forms a *correct pair* if $\pi^{-1}(u) < \pi^{-1}(v)$; i.e., vertex u precedes vertex v in the true arrival order. Given a partial order σ, we can always algorithmically find a total order consistent with σ (i.e., a linear extension of σ).

We formally define measures for quantifying the performance of any estimator. For a partial order σ, let $K(\sigma)$ denote the number of pairs $\{u,v\}$ that are comparable under σ: i.e., $K(\sigma) = |\{(u,v) \,:\, u <_\sigma v\}|$, where $|K(\sigma)| \leq \binom{n}{2}$.

Density: This is the number of comparable pairs in σ normalized by the total number of pairs, that is, $\delta(\sigma) = \frac{K(\sigma)}{\binom{n}{2}}$. The density of a partial order estimator ϕ is thus $\delta(\phi) = \min_{H \in \mathfrak{G}_n}[\delta(\phi(H))]$, where $\mathfrak{G}_n$ is the set of all graphs of size n.

Precision: This measures the expected fraction of *correct* pairs out of all pairs dictated by the partial order. That is,

$$\theta(\sigma) = \mathbb{E}\left[\frac{1}{K(\sigma)}|\{u,v \in [n] \,:\, u <_\sigma v, \pi^{-1}(u) < \pi^{-1}(v)\}|\right].$$

For an estimator ϕ, we denote by $\theta(\phi)$ the quantity $\mathbb{E}[\theta(\phi(\pi(G)))]$.

Recall: This measures the expected fraction of correct pairs (out of the total number of pairs) output by an algorithm inferring a

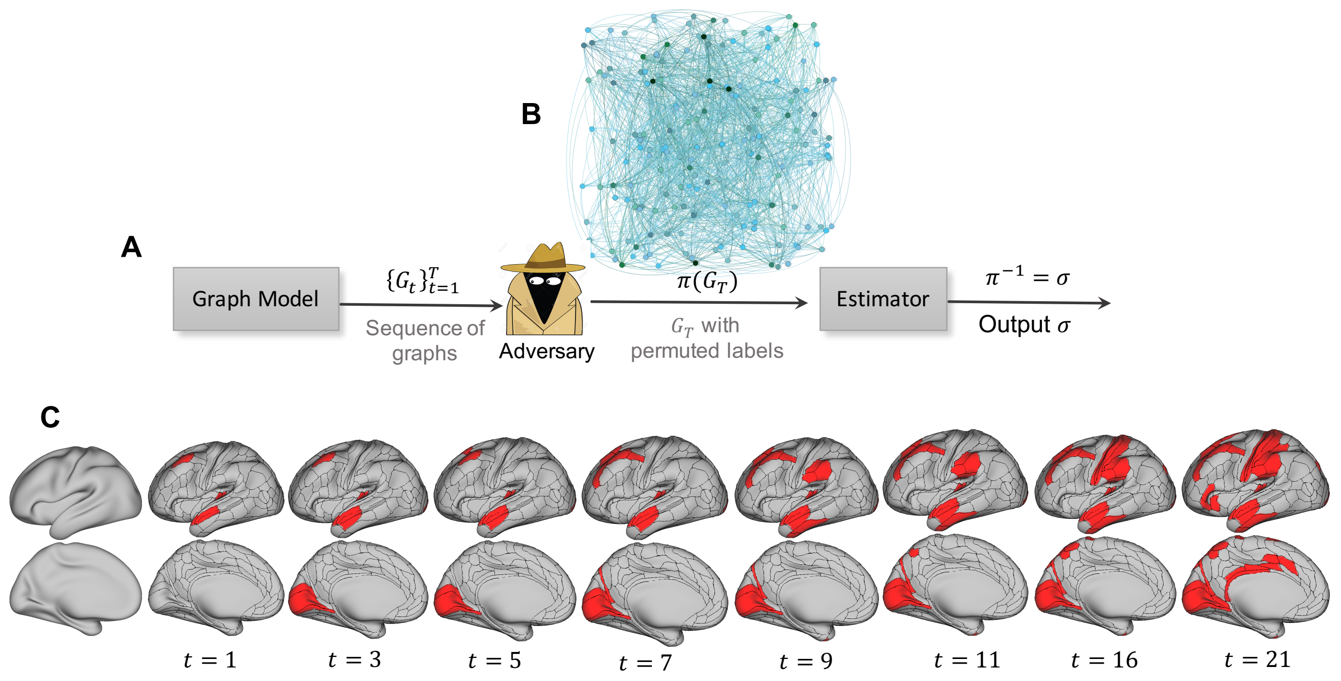


Figure 1. A. Block diagram of our formulation. **B.** A network of human brain formed from Human Connectome Project (HCP) data. This network is shown as an example of $\pi(G_T)$. Since the estimator should not dependent on the permutation, applying an unknown adversary permutation on node labels is equivalent to making the graph unlabeled. **C. Human brain evolution deduced by our method:** Starting from network data in B, we apply our techniques and make an inference on how brain regions evolve in the left hemisphere of a human brain. The time instant t in the figure represents an instant of change in the evolution of brain regions. The data and code are available at[12]. See Figure 5 and the Methods for more details.

partial order σ, that is,

$$\rho(\sigma) = \mathbb{E}\left[\frac{1}{\binom{n}{2}}|\{u,v \in [n] \;:\; u <_\sigma v, \pi^{-1}(u) < \pi^{-1}(v)\}|\right].$$

Figure 2A presents a sample graph, the permuted vertex labels, a candidate partial order, and measures of density, precision, and recall for the partial order.

We present our key results and illustrate them in the context of dynamic networks generated by Barabási-Albert preferential attachment model. Many dynamic networks arising in a variety of applications are hypothesized to follow this model of "rich-gets-richer" mechanism[13–19]. We denote a dynamic graph generated by the preferential attachment model as $\mathscr{PA}(n,m)$[13], where n is the number of nodes and m is the number of connections a new node makes to existing nodes when it is added to the network. At $t = 1$ a single vertex (labeled 1) is created with m self loops. To construct graph G_t at time $1 < t \leq n$, vertex t joins the network and makes m independent connections to the existing nodes in graph G_{t-1} with probability $\Pr[t \text{ connects to } k | G_{t-1}] = \frac{\deg_{t-1}(k)}{2m(t-1)}$, where $\deg_{t-1}(k)$ is the degree of node k at time $t-1$. Let $\mathrm{DAG}(G)$ be the directed acyclic version of G with the direction of edges marked in accordance with the graph evolution (leading from younger nodes to older nodes). For $\pi(G)$, edge directions are captured in its directed version $\pi(\mathrm{DAG}(G))$. Note that $\mathrm{DAG}(G)$ and $\pi(\mathrm{DAG}(G))$ have the same structure. This is illustrated in Figure 2B.

If we restrict the estimator to output a *total order*, i.e., $\delta(\sigma) = 1$, we show in SM Section 3 that no algorithm can solve the problem with error probability asymptotically bounded away from 1. As a specific instance of a solution procedure, one may also frame the problem in terms of maximum likelihood estimation as follows:

$$\mathscr{C}_{\mathrm{ML}}(H) = \arg\max_{\sigma \in S_n} \Pr[\pi^{-1} = \sigma | \pi(G) = H].$$

We show that the set $\mathscr{C}_{\mathrm{ML}}$ yields a large number of equiprobable solutions, $|\mathscr{C}_{\mathrm{ML}}| = e^{n\log n - O(n\log\log n)}$ with high probability, and therefore the maximum likelihood formulation is unsuitable. ($f(x) = O(g(x))$ indicates that there exist $\delta > 0$ and $M > 0$ such that $|f(x)| \leq M|g(x)|$ for $|x - a| < \delta$.)

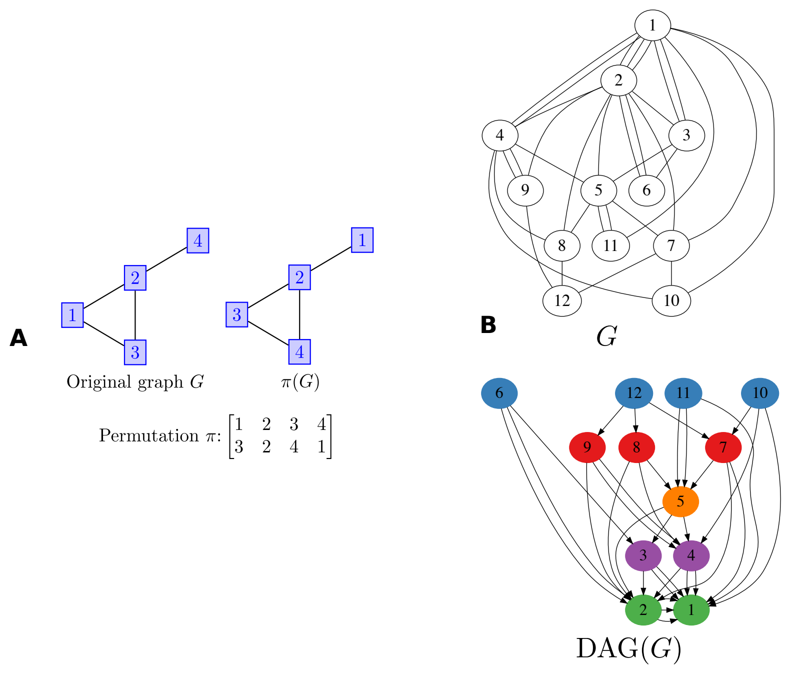

A
Original graph G
$\pi(G)$
Permutation π: $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 4 & 1 \end{bmatrix}$
B
G
$\pi(G)$
DAG(G)
π(DAG(G))

Figure 2. A. An example scenario. The estimator sees only $\pi(G)$ and must infer π^{-1}. E.g., it may output the order $\sigma = \{4 \prec 1 \prec 2\}$ The relation $4 \prec 1$ is correct, since $\pi^{-1}(4) = 3 < \pi^{-1}(1) = 4$, but the relations $4 \prec 2$ and $1 \prec 2$ are incorrect, since $\pi^{-1}(4) = 3 > \pi^{-1}(2) = 2$ and $\pi^{-1}(1) = 4 > \pi^{-1}(2) = 2$. The density is $\delta(\sigma) = 3/\binom{4}{2} = 3/6 = 1/2$, the precision is $\theta(\sigma) = 1/K(\sigma) = 1/3$, and the recall is $\rho(\sigma) = \theta(\sigma)\delta(\sigma) = 1/6$. **B**. The original graph (left) and the observed graph (right) for an instance of π: the same bin nodes in the DAGs have the same colors. Note that DAG(G) and π(DAG(G)) have exactly the same structure. The π(DAG(G)), generated by PEELING algorithm recovers all the probability one order information of G.

Formulation and solution of the underlying optimization problem. In view of this negative result, we consider estimators outputting a *partial order* on nodes. Here, an estimator may make fewer vertex pair order inferences, in exchange for higher precision (e.g., by grouping nodes and inferring the order only across groups, but not within groups). We then seek an optimal estimator in the following sense: for an input parameter $\varepsilon \in [0,1]$, we seek an estimator ϕ with density $\delta(\phi) \geq \varepsilon$ and maximum possible precision $\theta(\phi)$. This yields an *optimal curve* $\theta_*(\varepsilon)$, that characterizes the tradeoff between precision and density. We derive computable bounds on this curve and present efficient heuristic estimators that approach the bounds.

Given a graph H, define the function $J_\varepsilon(\phi)$ as the fraction of correctly inferred vertex orderings from among all allowable orderings by a given partial order. That is,

$$J_\varepsilon(\phi) = \frac{\mathbb{E}\left[|\{u,v \in [n] : u <_{\phi(H)} v, \pi^{-1}(u) < \pi^{-1}(v)\}| \middle| \pi(G) = H\right]}{K(\phi(H))},$$

and the conditional expectation is with respect to the randomness in π and G. To exhibit an optimal estimator, it is sufficient to choose, for each H, a value for $\phi(H)$ (i.e., a partial order) that maximizes the expression $J_\varepsilon(\phi)$ subject to the density constraint, $K(\phi(H)) \geq \varepsilon\binom{n}{2}$. We can then write the precision of estimator ϕ as:

$$\theta(\phi) = \sum_H \Pr[\pi(G) = H] J_\varepsilon(\phi).$$

To construct an optimal estimator, for each ordered pair (u,v) of vertices of H, we associate a binary variable $x_{u,v}$, where setting $x_{u,v} = 1$ indicates that $u <_{\phi(H)} v$. We can then rewrite $J_\varepsilon(\phi)$ as:

$$J_\varepsilon(\phi) = \frac{\sum_{1 \leq u < v \leq n} p_{u,v}(H) x_{u,v}}{\sum_{1 \leq u \neq v \leq n} x_{u,v}}, \tag{1}$$

where $p_{u,v}(H) = \Pr[\pi^{-1}(u) < \pi^{-1}(v) | \pi(G) = H]$ is the probability that u arrived before v given the permuted graph H, with the following constraints coming from the partial order and from our constraint on a given minimum density:

1. Antisymmetry: $x_{u,v} + x_{v,u} \leq 1$.

2. Transitivity: $x_{u,w} \geq x_{u,v} + x_{v,w} - 1$ for all $u, v, w \in [n]$.
3. Minimum density: $\sum_{1 \leq u \neq v \leq n} x_{u,v} \geq \varepsilon \binom{n}{2}$.
4. Domain restriction: $x_{u,v} \in \{0, 1\}$ for all $u, v \in [n]$.

We efficiently upper bound the optimal precision for any given density constraint ε as follows: on a randomly generated input graph $H = \pi(G)$, we recover its edge directions (i.e., $\pi(\mathrm{DAG}(G))$) and use them to approximate the coefficients $p_{u,v}(H)$ up to some relative error. The resulting rational linear integer program with approximated $p_{u,v}(H)$ can be converted into an equivalent linear integer program using a standard renormalization transformation, and we consider its natural linear programming relaxation with $x_{u,v}(H) \in [0, 1]$ for all u, v. This can be solved in polynomial time using standard algorithmic tools. We show the nature of this bound in Figure 3.

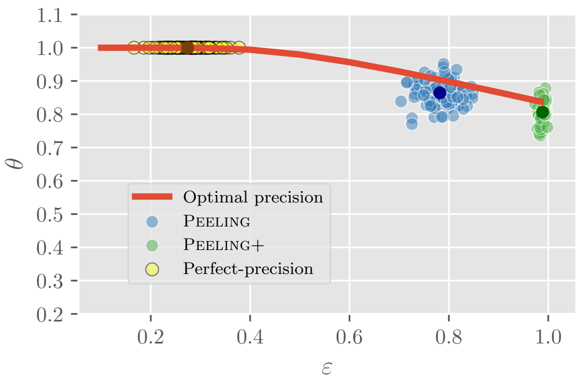


Figure 3. LP relaxation to the optimal precision curve $\theta_*(\varepsilon)$ and estimators for $G \sim \mathscr{PA}(n = 50, m = 3)$. The bold points indicate averaged value. The proposed three estimators serve different purposes. The perfect-precision estimator outputs pairs with full accuracy, but only a few. The PEELING+ gives a total order, but with less accuracy (which is much better than random guessing, and close to the optimal algorithm). The PEELING stands in the middle with better accuracy than PEELING+, and yet recovers a constant fraction of number of pairs.

To characterize the probability $p_{u,v}(H)$ and thus to solve the optimization, we prove that for all $u, v \in [n]$ and graphs H

$$p_{u,v}(H) := \Pr[\pi^{-1}(u) < \pi^{-1}(v) | \pi(G) = H] = \frac{|\{\sigma \,:\, \sigma^{-1} \in \Gamma(H), \sigma^{-1}(u) < \sigma^{-1}(v)\}|}{|\Gamma(H)|},$$

where the subset $\Gamma(H) \subset S_n$ consists of permutations σ such that $\sigma(H)$ has positive probability under the distribution $\mathscr{PA}(n, m)$ (see SM Lemma 4.1). Thus the estimation of $p_{u,v}(H)$ can be reduced to counting linear extensions of the partial order given by $\pi(\mathrm{DAG}(G))$, which is known to be #P-complete (ruling out an efficient exact algorithm). However, we propose a Markov chain Monte Carlo algorithm that achieves sufficiently fast convergence in practice.

Exact recovery of edge directions. Given access to $H = \pi(G)$, the following algorithm, which we call the **PEELING** technique, efficiently recovers $\pi(\mathrm{DAG}(G))$ (thus the edge directions) for a graph G (see Figure 1B). The algorithm starts by identifying the lowest-degree nodes (in our model, the nodes of degree exactly m), which are grouped into a bin. Then, it removes all of these nodes and their edges from the graph. The process proceeds recursively until there are no more nodes. To construct $\pi(\mathrm{DAG}(G))$ during this process, we note that all of the edges of a given degree-m node in a given step of the peeling process must be to older nodes; hence their orientations can be recovered. In the SM Section 6.1, we show that $\pi(\mathrm{DAG}(H))$ captures all the probability-1 information about vertex orderings in H and PEELING exactly recovers $\pi(\mathrm{DAG}(H))$

Estimators. Due to the high polynomial time complexity involved in solving the optimal scheme (estimating the upper bound requires $O(n^5 \log^3 n)$ calculations), we now provide efficient estimators whose performance is close to the optimal curve (see SM Section 6.2 for detailed analysis). In fact, the linear program itself does not yield an optimal scheme (one has to do a rounding step, which only yields an approximation) or an estimator, but only an upper bound on the optimal precision. Moreover, converting it to an optimal estimator is potentially computationally difficult, and thus efficient heuristics are needed.

1. *Maximum-density precision* 1 *estimator*: The estimator itself takes as input a graph $\pi(G)$ and outputs the partial order as $\pi(DAG(G))$ (all connected node pairs with order as the direction of the connection) as recovered by the PEELING algorithm. This estimator gives the maximum density among all estimators that have precision one; however, as shown in Theorem 6.2 of SM, we only can recover $o(n^2)$ correct pairs.
2. PEELING - *A linear binning estimator via peeling*: When the term PEELING is used as an estimator, we mean the serial binning estimator from the bins (groups of nodes) given by the PEELING technique. In particular, the sequence of subsets of vertices removed during each step naturally gives a partial order: each such subset forms a bin, and bins that are removed

earlier are considered to contain younger vertices (see Figure 2B). The PEELING estimator, which returns the bins, outputs strictly more vertex pair order guesses than the optimal precision-one estimator. In particular $\Theta(n^2)$ pairs, but some are not guessed correctly, and thus sacrifices some precision for increased density

3. PEELING+, *Peeling with deduction of same bin pairs*: This estimator runs on top of the PEELING estimator and attempts to order nodes within bins/ groups. For each node, we find the averaged value of its neighbors' bin numbers (levels), which we call the node's *average neighbor level*. A high value of average neighbor level indicates youth of the node. For each pair of nodes inside each bin, we infer the order between them based on the the averaged neighbor level of the respective nodes.

Figure 3 compares these estimators with the optimal one based on the integer programming formulation above. These estimators are observed to have performance close to optimal, at different points on the optimal curve. *Furthermore, the time complexity of these estimators is dominated by the DAG construction, and is $O(n \log n)$.*

Experiments. In what follows, $\sigma_{\text{perf}}, \sigma_{\text{peel}}, \sigma_{\text{peel+}}$ denote the partial orders produced by the Perfect-precision, PEELING, and PEELING+ estimators.

Robustness of the PEELING *algorithm*: Table 1 demonstrates robustness of our PEELING algorithm for various generalizations of the model: preferential attachment model with variable m (denoted by M and $\sim \text{unif}\{a,b\}$ denote discrete uniform distribution), uniform attachment model (denoted by $\mathscr{UA}$), and the more general Cooper-Frieze model[20, 21]. In our instance of the Cooper-Frieze model, the number of new edges m is drawn from $\sim \text{unif}\{5,50\}$, the model allows either addition of a new node (with probability 0.75) or addition of edges between existing nodes, and the endpoints of new edges can be selected either preferentially (with probability 0.5) or uniformly among existing nodes. These results suggest that the proposed DAG-based

Technique	$\theta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel}})$	$\delta(\sigma_{\text{peel}})$
$\mathscr{PA}(n, m=25)$	0.958	0.936	0.977
$\mathscr{PA}(n,M)$, $M \sim \text{unif}\{5,50\}$	0.691	0.683	0.988
$\mathscr{UA}(n, m=25)$	0.977	0.967	0.99
$\mathscr{UA}(n,M)$, $M \sim \text{unif}\{5,50\}$	0.827	0.823	0.995
Cooper-Frieze (Web graph) model	0.828	0.822	0.993

Table 1. A general comparison: $n = 5000$

methods can simultaneously achieve high precision and recall/density.

Real-world networks: We now discuss the performance of our estimators on several real-world networks, presented in Table 2 We first consider the ArXiv network as a directed network, with nodes corresponding to publications and edges from each publication to those that it cites. We also analyze the Simple English Wikipedia network – a directed graph showing the hyperlinks between articles of the Simple English Wikipedia. The DBLP computer science bibliography data is then modeled as an undirected network; an edge between two authors represents a joint publication. Finally, we study an SMS network and an online social network of Facebook focused on the New Orleans region, USA, with an edge (u,v,t) representing that user u posted on user v's wall at time t. Results are presented in Table 2. For all of the networks tested, the methods described here yield excellent precision and density.

Dataset	# Nodes	# Edges	Genre	$\theta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel}})$	$\delta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel+}})$
ArXiv High Energy Physics	7.46K	116K	Citation	0.708	0.681	0.961	0.707
Simple English Wikipedia	100K	1.62M	Hyperlink	0.624	0.548	0.878	0.609
DBLP CS bibliography	1.13M	5.02M	Coauthorship	0.785	0.728	0.927	0.764
Facebook Wall post	43.9K	271K	Social	0.698	0.657	0.941	0.687
SMS network	30.2K	447K	Social	0.669	0.610	0.912	0.621

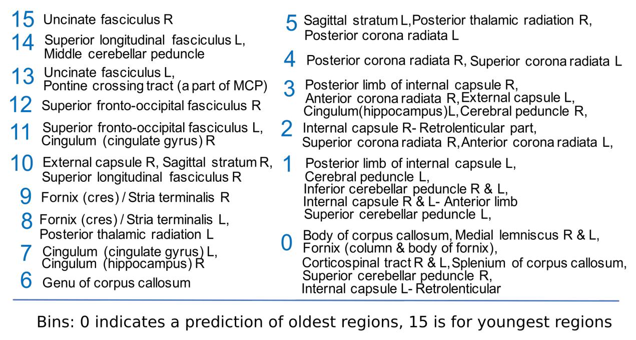
Table 2. Results for real-world networks: A detailed description of the datasets are given in SM. $\theta(\sigma_{\text{peel+}}) \approx \rho(\sigma_{\text{peel+}})$ and $\delta(\sigma_{\text{peel+}}) \approx 1$. When the density of the recovered partial order by PEELING algorithm is low, the recall can be improved via PEELING+ with a slight loss in precision (see the Wikipedia result).

.

Figures 4 and 5 presents results of our analysis on **human brain networks**. The purpose here is to recover the evolutionary order among the important regions inside the brain. We note that there is no available ground truth (in terms of the network) for such ranking. Therefore our ranking provides important insight for further application studies. However, our ranking is "supported" by prior application studies[14, 22], indicating that the brain network is well approximated by the preferential attachment mechanism and its variations. We study two independent sets of brain networks of resting state fMRI images. The first one is derived from the Cambridge-Buckner dataset with 56 labeled brain regions. The PEELING estimator provides a ranking of the brain regions; we analyze this ranking with respect to relatively sparse state of the art in our biological understanding of evolution of human brain. The corpus callosum, which joins the two hemispheres of the brain is believed to have developed in the earliest stages of brain evolution, and the uncinate fasciculus, the white matter tract to have evolved late in the human brain. These observations are consistent with the rankings returned by our PEELING estimator. Our rankings represent a first step towards determining the complete evolutionary trajectory of various regions of the brains.

The second network is extracted from the Human Connectome Project (HCP). We consider rankings of regions of 400 networks from 100 individuals (2 session and 2 scans per subject), with more detailed 300 labeled regions in cortex. We plot the histogram of arrival rank of prominent auditory, visual, and somatosensory regions. These histograms show a concentration of rankings, indicating that our rankings are consistent among 100 people in the regions considered. Moreover, we observe that most of the regions that serve as prime functionaries among auditory, visual, and somatosensory regions have consistently low arrival order in the 400 brain networks we analyzed. This is consistent with the widely accepted notion of early arrivals of these regions in the human brain evolution. Figure 5 illustrates the bins of brain regions of left hemisphere deduced with the PEELING estimator on a network generated from HCP data.

A



15	Uncinate fasciculus R
14	Superior longitudinal fasciculus L, Middle cerebellar peduncle
13	Uncinate fasciculus L, Pontine crossing tract (a part of MCP)
12	Superior fronto-occipital fasciculus R
11	Superior fronto-occipital fasciculus L, Cingulum (cingulate gyrus) R
10	External capsule R, Sagittal stratum R, Superior longitudinal fasciculus R
9	Fornix (cres) / Stria terminalis R
8	Fornix (cres) / Stria terminalis L, Posterior thalamic radiation L
7	Cingulum (cingulate gyrus) L, Cingulum (hippocampus) R
6	Genu of corpus callosum
5	Sagittal stratum L, Posterior thalamic radiation R, Posterior corona radiata L
4	Posterior corona radiata R, Superior corona radiata L
3	Posterior limb of internal capsule R, Anterior corona radiata R, External capsule L, Cingulum(hippocampus)L, Cerebral peduncle R,
2	Internal capsule R- Retrolenticular part, Superior corona radiata R, Anterior corona radiata L,
1	Posterior limb of internal capsule L, Cerebral peduncle L, Inferior cerebellar peduncle R & L, Internal capsule R & L- Anterior limb Superior cerebellar peduncle L,
0	Body of corpus callosum, Medial lemniscus R & L, Fornix (column & body of fornix), Corticospinal tract R & L, Splenium of corpus callosum, Superior cerebellar peduncle R, Internal capsule L- Retrolenticular

Bins: 0 indicates a prediction of oldest regions, 15 is for youngest regions

B

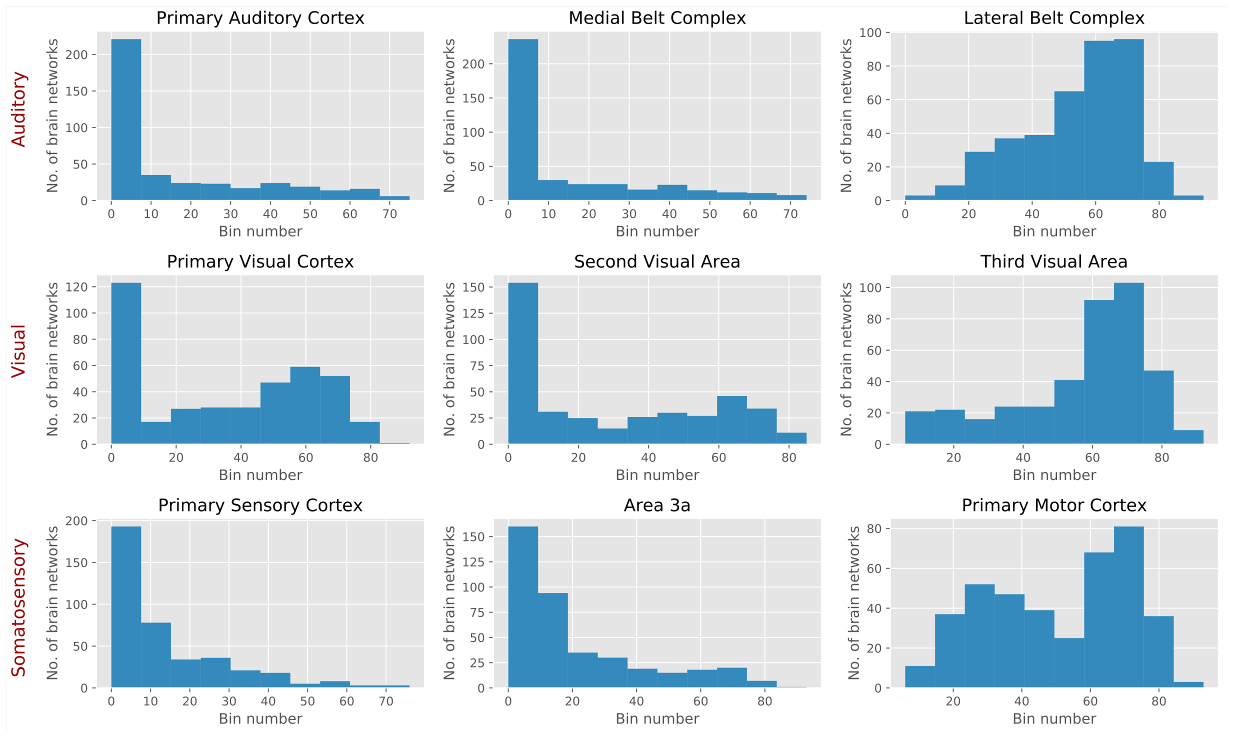


Figure 4. (**A**) PEELING bins of regions in Human brain network - Cambridge-Buckner. (**B**) Human Connectome brain data: histogram of selected regions in cortex.

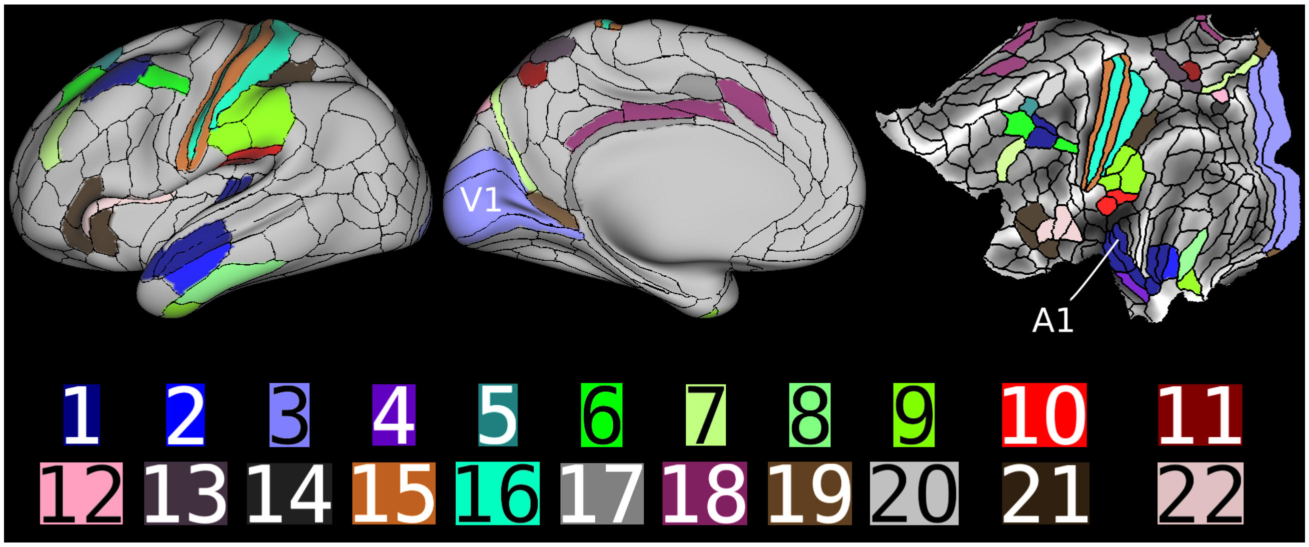


Figure 5. Illustration of the bins of prominent brain regions according to their arrival order in brain evolution: Inflated and flattened representation of left hemisphere of human brain. The network is formed from a correlation matrix of fMRI image of the data from Human Connectome Project (See Methods for details). The borders and regions are defined according to the multi-modal parcellation technique proposed in[23], and the figure is generated with Human Connectome Project's `workbench` tools. The node (brain region) ranking given by the PEELING algorithm are used to create batches of arrivals. The number indicated in the figure represents the first 22 bins, with bin 1 corresponding to the bin of oldest nodes. Note that the primary visual cortex (V1) and primary auditory cortex (A1) are classified into older bins. Code and data available at[12]

Discussion

We focus here on node arrival inference from a single snapshot of a dynamic network. Our models, analyses framework, and methods are applicable to a broad class of dynamic network generation models. Our infeasibility results (details in the SM) of total order recovery are useful in understanding fundamental limitations owing to different types of symmetries in networks. The general optimization problem we pose, which includes solutions of total and partial orders, provides an overarching framework within which disparate algorithms can be evaluated. We use this framework to argue near-optimal solutions from our estimators.

In a broader perspective, network archeology is not limited to the recovery of node arrival order, it generalizes to inference of higher order structures like triads, motifs, and communities. Our work provides the foundation for a rich class of problems in the area, both from analytical and applications' points of view. An alternate perspective of network archeology is in finding the course of (mis)information spreading. Recently, strategic information dissemination in online social networks like Twitter and Facebook has been alleged to create biases in opinions, even to the point of skewing electoral outcomes. Often, it is not one single source controlling the spread, rather, a group of nodes working in collusion. Our solutions provide powerful tools in identifying and quarantining these malicious nodes rapidly.

Methods

Code and data availability. We make our code available at `https://github.com/jithin-k-sreedharan/times`. The code supports random graph model (variants of preferential attachment model) and real-world networks. It also includes a script for generating brain networks from fMRI correlation data. The brain networks data is shared in the above link and the other networks used in this work are publicly available online. We solve the linear programming optimization using the Python interface of a commercial optimizer Gurobi, and the script is available in the same link.

Constructing the brain networks. The data of Human Connectome project is the resting state fMRI data from Human Connectome project focusing on the cortex area. The Human Connectome project provides a clean and refined data, which gives consistent results in many published studies. We process and form brain networks out of 100 healthy young adults. First, the cortex brain data corresponding to 100 subjects (2 sessions per subject, and 2 scans per session) is parcellated into 180 regions per hemisphere using a procedure described by Glasser et al.[23] Then the correlation matrices are formed from the time series of the parcellated data. Finally binary, undirected networks are constructed from the correlation matrices as follows: a spanning tree is created first from the complete network of the correlation matrix, and later k-nearest neighbors (higher correlation values) of each node are added into this network, where k is chosen as 10 in our case. Each network has 300 nodes, which are regions or clusters formed from group-Independent Component Analysis. The data is in correlation matrix format, with each element as the Gaussianized version of the Pearson correlation coefficient (Fisher Z transform with AR(1) process estimation). In order to form a binary adjacency matrix, we use a threshold just high enough to make the resulting graph connected. Such a graph is sparse.

Estimating $p_{u,v}$ using Markov chain Monte Carlo. We describe the procedure to estimate the integer programming coefficients $p_{u,v} = p_{u,v}(H)$ ($\Pr[\pi^{-1}(u) < \pi^{-1}(v) | \pi(G) = H]$). Solving the original optimization requires knowledge of $P = [p_{u,v}]$, which can be estimated via MCMC (Markov chain Monte Carlo) techniques. The order of convergence of one important MCMC technique (which we will call the Karzanov-Khachiyan algorithm and denote by K-K) for sampling uniformly from the set of linear extensions of a partial order, reported by[24], is $O(n^6 \log(n) \log(1/\varepsilon))$ transitions to achieve ε-bounded error between the distribution of a sampled linear extension and the uniform distribution. Estimation of certain functions of the set of linear extensions in general requires more transitions. For instance, Brightwell and Winkler[4] proved that estimating the total number of linear extensions based on K-K chain requires $O(n^9 \log^6(n))$ transitions. From a practical, computational perspective, this time complexity is untenable. Thus, we propose a different random walk (RW)-based algorithm.

First, a linear extension graph is formed as $G_{\mathrm{LE}} = (V_{\mathrm{LE}}, E_{\mathrm{LE}})$, where vertex set V_{LE} consists of linear extensions $\Gamma(H)$ consistent with the partial order (DAG) given by the DAG. The extensions λ and μ are adjacent in the graph if and only if λ can be obtained from μ by an adjacent transposition. We describe a RW process below, which does not require the graph to be known beforehand, instead the graph will be explored locally as neighbors of the nodes sampled by the random walk.

For instance, let $\{v_1, v_2, v_3, v_4\}$ be the nodes of the underlying PA graph. Let the partial order given by the DAG be $v_2 < v_3$ and $v_4 < v_1$. Then a node in the linear extension graph (which is a linear extension with the given partial order) is $v_4 < v_1 < v_2 < v_3$. Among the three possible adjacent transpositions of this total order, only one is a linear extension, which is $v_4 < v_2 < v_1 < v_3$. Thus the degree of this total order is 1. Figure 6 shows the graphs $\mathrm{DAG}(G)$ and G_{LE} for this example.

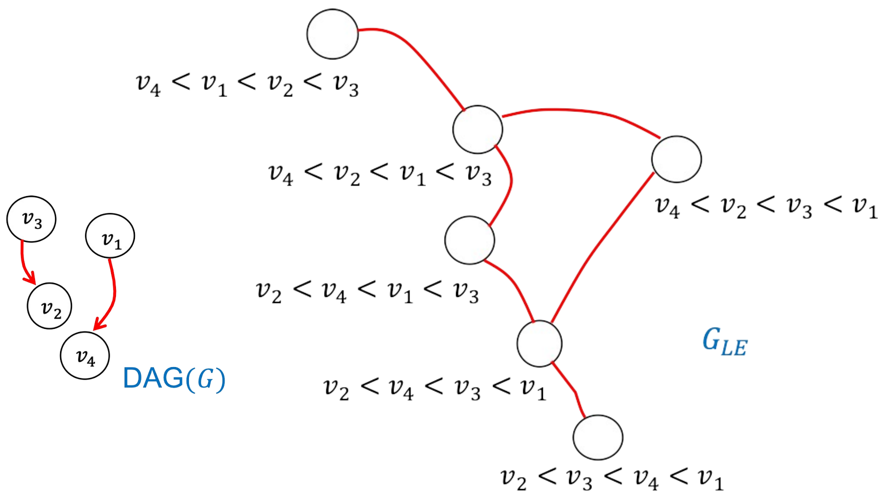


Figure 6. An example $\mathrm{DAG}(G)$ and its linear extension graph G_{LE}

The algorithm is as follows:

1. We sample a node λ in G_{LE}, which is a linear extension, using the SEQUENTIAL algorithm (The SEQUENTIAL algorithm works similar to the PEELING technique, but instead of peeling away all the m-degree nodes at each step, it removes only a randomly selected node among the m-degree nodes present at any step and all other nodes stay for the next removal.). Let it be the initial node.

2. The neighbor set of λ can be obtained as follows. If any adjacent elements in λ form a perfect pair, they are not allowed to swap positions. All other adjacent pairs are allowed to transpose, and each neighbor of λ corresponds to a linear extension differed from λ with one transposed pair.

3. We run a simple random walk on graph G_{LE} with the random walk choosing the next node in the walk uniformly among the neighbors of the present node.

4. Such a RW has a stationary distribution $d(\lambda)/\sum_{\mu\in\Gamma(H)} d(\mu)$, where $d(\lambda)$ is the degree of linear extension λ in G_{LE}. We form the following ratio form estimator for $P_{u,v}$ without directly unbiasing such a non-uniform distribution, which is

impossible without the knowledge of $\sum_{\mu \in \Gamma(H)} d(\mu)$.

$$\hat{p}_{u,v}^{(k)} := \frac{\sum_{t=1}^{k} \mathbf{1}\{X_t(u) < X_t(v)\}/d(X_t)}{\sum_{t=1}^{k} 1/d(X_t)} \xrightarrow{k\to\infty} \frac{|\mu \in \Gamma(H) : \mu(u) < \mu(v)|}{|\Gamma(H)|} \quad \text{a.s.}$$

Here X_t indicate the t-th sample of the RW, which is a linear extension of the underlying DAG.

5. Stop the RW when a convergence criteria is met.

Note that, unlike in K-K method, we do not need to make Markov chain aperiodic as the intention is not to sample from the unique stationary distribution, but only to estimate an average function of the nodes. The K-K method first forms a Markov chain similar to the construction in the above algorithm, but makes it aperidoc by adding self transitions with probability $1 - d(\lambda)/(2n-2)$. Given the discussion in SM Section 6.2.1, we expect $d(\lambda)$ should be very small, and hence the Markov chain in the K-K method spends a large amount of time in self loops, thus making the mixing slower. Our method avoids artificial self loops, and achieves faster convergence in practice.

References

1. Li, A., Cornelius, S. P., Liu, Y.-Y., Wang, L. & Barabási, A.-L. The fundamental advantages of temporal networks. *Science* **358**, 1042–1046 (2017).

2. Pinney, J. W., Amoutzias, G. D., Rattray, M. & Robertson, D. L. Reconstruction of ancestral protein interaction networks for the bzip transcription factors. *Proc. Natl. Acad. Sci.* **104**, 20449–20453 (2007).

3. Srivastava, M. *et al.* The amphimedon queenslandica genome and the evolution of animal complexity. *Nature* **466**, 720–726 (2010).

4. Brightwell, G. & Winkler, P. Counting linear extensions. *Order* **8**, 225–242 (1991).

5. Navlakha, S. & Kingsford, C. Network archaeology: Uncovering ancient networks from present-day interactions. *PLOS Comput. Biol.* **7**, 1–16 (2011).

6. Young, J.-G. *et al.* Network archaeology: phase transition in the recoverability of network history. *arXiv preprint arXiv:1803.09191* (2018).

7. Bubeck, S., Devroye, L. & Lugosi, G. Finding adam in random growing trees. *Random Struct. & Algorithms* (2016).

8. Frieze, A., Pegden, W. *et al.* Looking for vertex number one. *The Annals Appl. Probab.* **27**, 582–630 (2017).

9. Shah, D. & Zaman, T. Rumors in a network: Who's the culprit? *IEEE Transactions on information theory* **57**, 5163–5181 (2011).

10. Zhu, K. & Ying, L. Information source detection in the sir model: A sample-path-based approach. *IEEE/ACM Transactions on Netw.* **24**, 408–421 (2016).

11. Magner, A., Sreedharan, J. K., Grama, A. Y. & Szpankowski, W. Times: Temporal information maximally extracted from structures. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, 389–398, DOI: 10.1145/3178876.3186105 (2018).

12. Code & data of this submission. Available at https://github.com/jithin-k-sreedharan/times.

13. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).

14. Klimm, F., Bassett, D. S., Carlson, J. M. & Mucha, P. J. Resolving structural variability in network models and the brain. *PLoS computational biology* **10**, e1003491 (2014).

15. Watts, D. J. The "new" science of networks. *Annu. Rev. Sociol.* **30**, 243–270 (2004).

16. Perc, M. Evolution of the most common english words and phrases over the centuries. *J. The Royal Soc. Interface* rsif20120491 (2012).

17. Kunegis, J., Blattner, M. & Moser, C. Preferential attachment in online networks: Measurement and explanations. In *Proceedings of the 5th Annual ACM Web Science Conference*, 205–214 (ACM, 2013).

18. Barabási, A.-L. *et al.* Evolution of the social network of scientific collaborations. *Phys. A: Stat. mechanics its applications* **311**, 590–614 (2002).

19. Barabási, A.-L. Network science: Luck or reason. *Nature* **489**, 507 (2012).

20. Cooper, C. Distribution of vertex degree in web-graphs. *Comb. Probab. Comput.* **15**, 637–661 (2006).

21. Cooper, C. & Frieze, A. A general model of web graphs. *Random Struct. & Algorithms* **22**, 311–335 (2003).

22. Vértes, P. E. *et al.* Simple models of human brain functional networks. *Proc. Natl. Acad. Sci.* **109**, 5868–5873 (2012).

23. Glasser, M. F. *et al.* A multi-modal parcellation of human cerebral cortex. *Nature* **536**, 171–178 (2016).

24. Karzanov, A. & Khachiyan, L. On the conductance of order markov chains. *Order* **8**, 7–15 (1991).

Acknowledgements

This work was supported by NSF Center for Science of Information (CSoI) Grant CCF-0939370, NSF Grants CCF-1524312 and CSR-1422338, and by NIH Grant 1U01CA198941-01. We thank Vikram Ravindra (Purdue) for the help in collecting the brain data.

Author contributions statement

All authors designed and performed the research, and analyzed the results. J.S. implemented the methods and ran the experiments. A.M performed the analytical calculations, and J.S. and W.S. assisted him. J.S. drafted the initial manuscript, and A.M., A.G. and W.S. edited and refined the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Supplementary Material: Inferring Temporal Information from a Snapshot of a Dynamic Network

Jithin Sreedharan[†], Abram Magner[†], Ananth Grama, and Wojciech Szpankowski[*]

Center for Science of Information, Purdue University, USA
[*]To whom correspondence should be addressed
E-mail: szpan@purdue.edu
[†]Authors contributed equally.

February 3, 2019

Contents

1 Introduction

Throughout the supplementary information, we will generally follow certain notational conventions (unless otherwise stated), which we describe here. An arbitrary random graph model on n vertices is denoted by $\mathcal{G}_n$, and a sample from that model by G. The random permutation generated by an adversary is given by π, and we use H to denote the observed graph $\pi(G)$ (a random variable) or one of its possible values. Fixed graphs will generally be denoted by capital letters.

1.1 Overview of prior work

Mahantesh et al. [15] empirically studied inference of arrival order via a binning method. In this context, we provide a rigorous formulation and analysis, a simpler solution without having to generate samples from an estimated random graph model. We also provide a precise characterization of the performance of our methods. To the best of our knowledge, our work presents the first feasible and rigorous approach to the problem of inferring partial orders in preferential attachment graphs. We first formulated the present problem and presented some preliminary results in [14, 13].

For preferential attachment graphs, Frieze et al.[6] study the problem of identifying the oldest node. However, their setting and goal are rather different: they assume that the arrival order of nodes is known, and their goal is to arrive at the oldest node by a process that starts at a different node and only uses local information to advance.

2 Model and Definitions

The given formulation is equivalent to the following intuitive scenario: we are given a graph G with unique vertex ids, and there exists a hidden bijection ψ from the set of ids to the integers $\{1, ..., n\}$ giving the order of arrival of the vertices, identified by their names. The task is to use the structure of G to recover ψ. Formally, then, the problem that we study is entirely specified by the following data: a distribution on labeled graphs (for the bulk of this paper, the preferential attachment distribution $\mathcal{PA}(n, m)$, but in general denoted by $\mathcal{G}_n$), the form that a solution takes, and a way of evaluating the performance of a given solution method.

To present succinctly our results we need three quantities associated with permutations of a graph model:

(a) *Set of feasible permutations of a graph H*: the subset $\Gamma(H) \subseteq S_n$, which consists of permutations σ, such that $\sigma(H)$ has positive probability under the distribution $\mathcal{PA}(n, m)$. An example of a permutation that is *not* feasible for a graph G generated by preferential attachment is $\pi = (1, n)$, which swaps the first and last vertices. This is because the degree of the vertex labeled n in the resulting graph $\pi(G)$ is $> m$, which happens with probability zero. We know that for this model $\mathbb{E}[\log |\Gamma(G)|] = n \log n - O(n \log \log n)$ [11].

(b) *Set of admissible graphs of H*: $\mathrm{Adm}(H) = \{\sigma(H) : \sigma \in \Gamma(H)\}$. These are graphs obtained by applying $\Gamma(H)$ to H.

(c) *Automorphism group*: The *automorphism group* of a graph G, denoted by $\mathrm{Aut}(G)$, is the set of permutations π of vertices of G that preserve edge relations (i.e., the number of edges between vertices u and v is equal to the number of edges between vertices $\pi(u)$ and $\pi(v)$ in G). Note that $|\mathrm{Adm}(H)| = |\Gamma(H)|/|\mathrm{Aut}(H)|$ [11].

3 Infeasibility of Total Order Recovery

Here we restrict the estimator to output a total order (i.e., a permutation) for which $\delta(\sigma) = 1$. An estimator function takes the form $\phi : \mathfrak{G}_n \to S_n$ and $\mathfrak{G}_n$ is the set of graphs on n vertices.

Let the minimax risk for a random graph model $\mathcal{G}_n$ be denoted by

$$R_*(\mathcal{G}_n, d) = \min_{\phi} \max_{\mathfrak{A}_n} \mathbb{E}[d(\phi(\pi(G)), \pi^{-1})],$$

where $\mathfrak{A}_n$ is the the set of all adversaries, and d is some distortion measure on permutations. We focus on two natural choices for d, namely, error probability $d_e(\sigma_1, \sigma_2) = I[\sigma_1 \neq \sigma_2]$ for exact recovery, and Kendall τ distance $\tau(\sigma_1, \sigma_2) = \sum_{1 \leq i < j \leq n} I[\sigma_2\sigma_1^{-1}(i) > \sigma_2\sigma_1^{-1}(j)]$ for approximate recovery.

The next result says that, at least for the two distortion measures described above, the worst-case adversary is the uniform distribution.

Theorem 3.1. *In the case of error probability and Kendall τ distortion measures, we have, for any adversary $\mathfrak{A}_n$, that the minimum risk over any estimator is greater than or equal to that for the adversary that chooses a permutation uniformly at random.*

Proof. Consider an arbitrary adversary that outputs a permutation π on input G. We will construct a randomized estimator as follows: we draw uniformly at random a $\sigma \sim S_n$ and compute $H' = \sigma(\pi(G))$. Note that $\sigma \circ \pi$ is uniformly distributed on S_n. The estimator will then apply an optimal estimator for the uniform adversary to H', yielding a permutation λ, which is meant to be an approximation of $(\sigma \circ \pi)^{-1} = \pi^{-1} \circ \sigma^{-1}$. Finally, the estimator outputs $\lambda\sigma$ as an approximation of π^{-1}.

To analyze the precision of this estimator, we note the following chain of equalities holds, by elementary properties of the Kendall τ distance:

$$\tau(\lambda\sigma, \pi^{-1}) = \tau(\lambda, \pi^{-1}\sigma^{-1}) = \tau(\lambda, (\sigma\pi)^{-1}). \tag{1}$$

Since $(\sigma\pi)^{-1}$ is uniform on S_n and λ was generated by an optimal estimator for the uniform adversary, the expectation of the final expression is exactly equal to the minimum possible risk for the uniform adversary.

Finally, standard results yield that there exists a deterministic estimator with at least as large risk, which concludes the proof. The derivation for the error probability distortion measure is similar, and we omit it. $\square$

The following theorem gives a general upper bound on the recall.

Theorem 3.2. *Consider a random graph model $\mathcal{G}_n$ for which any two positive-probability isomorphic graphs are equiprobable. Then the minimax risk is*

$$\begin{aligned} R_*(\mathcal{G}_n, d_e) &\geq \frac{\mathbb{E}[\log|\mathrm{Aut}(G)|] + \mathbb{E}[\log|\mathrm{Adm}(G)|] - 1}{\log n!} \\ &= \frac{\mathbb{E}[\log|\Gamma(G)|] - 1}{\log n!}. \end{aligned}$$

Proof. We consider a particular adversary, one that chooses π uniformly at random from S_n. Applying Fano's inequality, we have that the probability of error is

$$\begin{aligned} \Pr[\phi(\pi(G)) \neq \pi^{-1}] &\geq \frac{H(\pi^{-1}|\phi(\pi(G))) - 1}{\log|S_n|} \\ &= \frac{H(\pi^{-1}|\phi(\pi(G))) - 1}{\log n!} \end{aligned}$$

Now, by the data processing inequality and chain rule,

$$\begin{aligned} H(\pi^{-1}|\phi(\pi(G))) &\geq H(\pi^{-1}|\pi(G)) \\ &= H(G|\pi(G)) + H(\pi^{-1}|G, \pi(G)). \end{aligned}$$

We note here that $H(\pi^{-1}|G, \pi(G)) = \mathbb{E}[\log |\mathrm{Aut}(G)|]$ as there are exactly $|\mathrm{Aut}(G)|$ automorphisms between G and $\pi(G)$, and $H(G|\pi(G)) = \mathbb{E}[\log |\mathrm{Adm}(G)|]$ since all isomorphic graphs with positive probability have the same probability. □

For the preferential attachment model, the following lemma proves the assumption in Theorem 3.2. Strictly speaking, we prove the lemma for the version of the model in which vertex degrees are updated with every connection choice. An approximate version holds under tweaks of the model (essentially because multiple edges are rare, and the results that rely on the lemma only require a very approximate version). In the proof and elsewhere throughout the supplementary information, we will use some common terms: a *DAG* is a directed, acyclic graph. The *structure* or *unlabeled* version of a graph or a DAG M (denoted by $S(M)$) is the isomorphism class of the object in question (i.e., the set of all graphs or DAGs that are equivalent to the given one, up to permutations of its labels).

Lemma 3.1. *Consider two isomorphic preferential attachment graphs $G_1, G_2 = \sigma(G_1)$, $\sigma \in \Gamma(G_1)$. If $\Pr[G = G_1] > 0$, then*

$$\Pr[G = G_1] = \Pr[G = G_2]. \tag{2}$$

Proof. First, we will show that for a positive-probability graph K, $\Pr[G = K]$ depends only on the unlabeled DAG of K (see Definition 5.1 for the definition of the labeled version – namely, an edge from u to v exists in the (labeled) DAG of K if and only if $u > v$ and there is an edge between u and v in K). Later in the supplementary information, we will state and prove Lemma 6.1, which gives an algorithm for recovering the unlabeled DAG of K from the structure $S(K)$, implying that the former is *uniquely determined* by the latter. This will complete the proof.

To show that $\Pr[G = K]$ depends only on the unlabeled DAG of K, we derive an explicit expression as follows: for a given vertex v of K, denote the set of neighbors of v that chose v by $N_K(v)$. We call this set the *parents* of v. Furthermore, let $d_1(v) \leq \cdots \leq d_{N_K(v)}(v)$ denote the number of edges from each of the parents of v to v (note that all of this information is given by the unlabeled DAG of K). Then we can write an expression for $\Pr[G = K]$ via the following considerations: this probability is a ratio, the denominator of which is the product $\prod_{j=1}^{mn-1}(2mj)$. This comes from the fact that for the jth connection choice made in the graph, the probability of connecting to *any* vertex contributes a denominator of $2mj$. To determine an expression for the numerator of $\Pr[G = K]$, we consider the contribution of vertex v, for arbitrary v with degree, say, $d > m$. When v joined the graph, it was given degree m. When a parent of v chooses to connect to v, bringing its degree from some d' to $d' + 1$, this contributes a factor of d', leading to a factor of $m(m+1)\cdots d = \frac{d!}{(m-1)!}$. However, more is needed: a given parent that chooses v, say, $d_i(v)$ times can do so in $\binom{m}{d_i(v)}$ ways. More precisely, if a given vertex chooses to connect to ℓ vertices $a_1, \cdots, a_\ell$ times, respectively, where $a_1 + \cdots + a_\ell = m$, then there are $\binom{m}{a_1, \cdots, a_\ell}$ ways to do this. In the expression for the numerator of $\Pr[G = K]$, we will factor out the $m!$ and associate the factor

$\frac{1}{a_i!}$ with its corresponding vertex (so $\frac{1}{d_i!}$ becomes associated with the vertex v). All of this leads to the following expression:

$$\Pr[G = K] = \prod_{j=1}^{mn-1} (2mj)^{-1} \times (m!)^{n-1} \times$$
$$\prod_{d>m} \prod_{v\ :\ \deg_K(v)=d} \frac{d!}{(m-1)!} \prod_{i=1}^{N_K(v)} \frac{1}{d_i!}$$

Here, $\deg_K(v)$ denotes the degree of vertex v in the graph K. The product indexed by d is over all degrees $> m$ present in the graph K, the product indexed by v is over all vertices having the given degree, and the one indexed by i is over all parents of v. Since all parts of this expression are dependent only on the structure of DAG(K), the lemma is proven. □

In particular for preferential attachment graphs, we have the following inapproximability result.

Theorem 3.3. *Let $\mathcal{G}_n$ denote $\mathcal{PA}(n, m)$, with $m \geq 3$. Then we have $R_*(\mathcal{G}_n, d_e) = 1 - o(1)$. Furthermore, for approximate recovery, $R_*(\mathcal{G}_n, d_a) = \Theta(n^2)$. In other words, the recall is at most* $1 - \Theta(1)$*, and thus no total order estimation algorithm can solve the problem without making a large number of inversion errors.*

Proof. Recall that $\tau(\cdot, \cdot)$ denotes the Kendall τ distance between two permutations.

We will consider preferential attachment graphs, and will prove the bound for approximate recovery, from which all other statements will follow. It is sufficient to exhibit an adversary distribution for which the statement is true. We consider the set of degree-m nodes (denote this set by $D_m(G)$), which, with high probability, has size $|D_m(G)| = \Theta(n)$. Denote by $S_{D_m(G)}$ the set of permutations in S_n which fix all vertices with degree not equal to m. Within this set, there is a subset $R(G)$, with size tending to ∞ with n and such that any two permutations within $R(G)$ have Kendall τ distance at least $\delta = cn^2$.

We consider an adversary that chooses π uniformly at random from $R(G)$.

Consider an arbitrary total order estimator ϕ. The intuition will be as follows: there are many possible graph/permutation pairs $(K, \hat{\pi})$ such that $\hat{\pi}(K) = \pi(G)$, and these $\hat{\pi}^{-1}$ are well-separated. So they must also be well-separated from $\phi(\pi(G))$. This will provide the lower bound on the expected number of errors.

We start by conditioning on the value of $\pi(G)$ and of π. Here, the outer sum is over all graphs K with at least $c'n$ degree-m nodes, for some appropriate fixed $c' > 0$.

$$\mathbb{E}[\tau(\phi(\pi(G)), \pi^{-1})] \geq \sum_K \Pr[\pi(G) = K] \times$$
$$\sum_{\hat{\pi} \in R(K)} \mathbb{E}[\tau(\phi(K), \pi^{-1}) I[\pi = \hat{\pi}] \mid \pi(G) = K].$$

The inner sum becomes

$$\sum_{\hat{\pi} \in R(K)} \tau(\phi(K), \hat{\pi}^{-1}) \Pr[\pi = \hat{\pi} \mid \pi(G) = K]. \qquad (3)$$

Now, the probability in this expression is simply $1/|R(K)|$, which is constant with respect to $\hat{\pi}$.

Thus, the inner sum further simplifies to

$$\frac{1}{|R(K)|} \sum_{\hat{\pi} \in R(K)} \tau(\phi(K), \hat{\pi}^{-1}). \tag{4}$$

To lower bound $\tau(\phi(K), \hat{\pi}^{-1})$ in this sum, denote by δ_{min} the quantity

$$\delta_{min} = \min_{\sigma \in R(K)} \tau(\phi(h), \sigma^{-1}), \tag{5}$$

and let σ_{min} denote the optimizing permutation. Now, using the triangle inequality, we have

$$\begin{aligned} \delta \le \tau(\hat{\pi}^{-1}, \sigma_{min}^{-1}) &\le \tau(\hat{\pi}^{-1}, \phi(K)) + \tau(\phi(K), \sigma_{min}^{-1}) \\ &= \tau(\hat{\pi}^{-1}, \phi(K)) + \delta_{min}. \end{aligned}$$

We thus have, for all $\hat{\pi} \in R(K)$ not equal to σ_{min},

$$\tau(\hat{\pi}^{-1}, \phi(K)) \ge \max\{\delta - \delta_{min}, \delta_{min}\}. \tag{6}$$

Then we have the following lower bound on (4).

$$\frac{1}{|R(K)|} \sum_{\hat{\pi} \in R(K)} \tau(\phi(K), \hat{\pi}^{-1}) \ge \frac{|R(K)| - 1}{|R(K)|} \max\{\delta - \delta_{min}, \delta_{min}\}.$$

Now, we plug this in as a lower bound on the inner sum of (3), which gives

$$\begin{aligned} &\mathbb{E}[\tau(\phi(\pi(G)), \pi^{-1})] \\ &\qquad \ge \sum_{K} \Pr[\pi(G) = K] \frac{|R(K)| - 1}{|R(K)|} \max\{\delta - \delta_{min}, \delta_{min}\}. \end{aligned}$$

The outer sum is again over the same set of graphs K as in (3).

Now, in order to simplify the final expression, we recall that $|R(K)| \to \infty$ as $n \to \infty$, and $\max\{\delta - \delta_{min}, \delta_{min}\} = \Theta(n^2)$. All of this gives

$$\mathbb{E}[\tau(\phi(\pi(G)), \pi^{-1})] \ge (1 - o(1)) \cdot \Theta(1) \cdot n^2 = \Theta(n^2). \tag{7}$$

This immediately implies the desired upper bound on the precision/recall for the uniform distribution adversary.

We can prove the result for Erdős-Rényi by directly considering the uniform distribution adversary. The details are quite similar, so we omit them.

□

3.1 Maximum likelihood estimation

For the preferential attachment model, a natural way to approach the total order estimation problem is to frame it in terms of maximum likelihood estimation as follows: $\mathcal{C}_{\mathrm{ML}}(H) = \arg\max_{\sigma \in S_n} \Pr[\pi^{-1} = \sigma | \pi(G) = H]$. The following proposition proves that the optimal solution set $\mathcal{C}_{\mathrm{ML}}$ gives a large number of equiprobable solutions and the maximum likelihood formulation is not a useful approach for the problem.

Theorem 3.4. *The maximum likelihood estimation solution set $\mathcal{C}_{\mathrm{ML}} = \mathcal{C}_{\mathrm{ML}}(\pi(G))$ satisfies $|\mathcal{C}_{\mathrm{ML}}| = e^{n \log n - O(n \log \log n)}$ with high probability.*

Proof. First, by definition of $\Gamma(H)$, we must have that $\mathcal{C}_{ML}(H) \subseteq \Gamma(H)$. We will show, in fact, that the likelihoods given to all elements of $\Gamma(H)$ are equal. This will then imply that $\mathcal{C}_{ML}(H) = \Gamma(H)$. Next, from a result of [11], we have that with high probability $|\Gamma(\pi(G))| = e^{n \log n - O(n \log \log n)}$, which will complete the proof.

So it is sufficient to show that, for each $\sigma \in \Gamma(H)$,

$$\Pr[G = \sigma(H) | \pi(G) = H]$$

depends only on the structure $S(H)$. To do this, note that by definition of $\mathrm{Adm}(H)$ and $\Gamma(H)$, we must have $\sigma(H) \in \mathrm{Adm}(H)$. So it is enough to show that for any two positive-probability isomorphic graphs G_1 and G_2, we have $\Pr[G = G_1] = \Pr[G = G_2]$. This is the content of Lemma 3.1. This completes the proof of the proposition. $\square$

4 Formulation and Solution of an Optimization Problem

Remark 4.1. We could have defined a tradeoff curve between precision and recall, but this would have the undesirable feature that the optimal precision is not necessarily defined for every value of recall in $[0, 1]$, since some values for recall are not achievable [12].

4.1 Reduction of rational integer program to linear program

It is a general result that a rational linear program such as ours may be solved by converting to an equivalent truly linear program. We define a new variable s, which will capture the rational part of the objective function (i.e., we make the substitution $s = 1/\sum_{1 \leq u \neq v \leq n} x_{u,v}$, and $y_{u,v} = s x_{u,v}$). The objective function is rewritten as a linear function of the normalized variables. This gives us the LP relaxation

$$\max_{y,s} \sum_{1 \leq u < v \leq n} p_{u,v}(H) y_{u,v}, \tag{8}$$

subject to the constraints

1. $y_{u,v} \in [0, s], \forall u, v \in [n]$
2. $0 \leq s \leq \dfrac{1}{\varepsilon \binom{n}{2}}$
3. $y_{u,v} + y_{v,u} \leq s, \forall u, v \in [n]$
4. $y_{u,v} + y_{v,w} - y_{u,w} \leq s, \forall u, v, w \in [n]$
5. $\displaystyle\sum_{1 \leq u < v \leq n} y_{u,v} = 1.$

4.2 Coefficients of optimal precision

The integer program in the Main Text is rather explicit, except for the coefficient $p_{u,v}(H)$. The next lemma gives a combinatorial formula for the probability $p_{u,v}(H)$.

Lemma 4.1 (Coefficients of the optimal precision)**.** *For all $v, w \in [n]$ and graphs H,* $\Pr[\pi^{-1}(v) < \pi^{-1}(w)|\pi(G) = H] = \frac{|\{\sigma\ :\ \sigma^{-1} \in \Gamma(H), \sigma^{-1}(v) < \sigma^{-1}(w)\}|}{|\Gamma(H)|}$.

Proof. We can express the conditional probability in question as a sum, as follows:

$$\begin{aligned}&\Pr[\pi^{-1}(v) < \pi^{-1}(w)|\pi(G) = H]\\ &= \sum_{\substack{\sigma\ :\ \sigma^{-1} \in \Gamma(H)\\ \sigma^{-1}(v) < \sigma^{-1}(w)}} \Pr[\pi = \sigma|\pi(G) = H]\end{aligned}$$

Now, recall that $\pi^{-1} \in \Gamma(H)$ under this conditioning, since $\pi(G) = H$ and G is admissible. Moreover, it is uniformly distributed on $\mathrm{Iso}(G, H)$ (the set of isomorphisms from G to H), so we have

$$\begin{aligned}\Pr[\pi = \sigma|\pi(G) = H] &= \frac{\Pr[G = \sigma^{-1}(H)|\pi(G) = H]}{|\mathrm{Aut}(H)|}\\ &= \frac{1}{|\mathrm{Aut}(H)||\mathrm{Adm}(H)|}.\end{aligned}$$

Taking the sum, this becomes

$$\frac{|\{\sigma\ :\ \sigma^{-1} \in \Gamma(H), \sigma^{-1}(v) < \sigma^{-1}(w)\}|}{|\mathrm{Aut}(H)||\mathrm{Adm}(H)|}.$$

Finally, recall that $|\mathrm{Adm}(H)| = |\Gamma(H)|/|\mathrm{Aut}(H)|$ [12]. □

5 Optimal Solution for Preferential Attachment Graph Model

From now on, we focus on the preferential attachment graph model, and solve the optimization problem explained in the previous section. In particular we describe how estimate $p_{u,v}(H)$. Later in the section, we provide some efficient estimators that provide good approximations for the optimal solution.

We remark that the existence of self-loops on the initial vertex in preferential attachment graphs allows for clean proofs; our theoretical and empirical results extend without significant changes to models in which these self-loops are not present.

5.1 Recovering edge directions

To efficiently calculate $p_{u,v}(H)$, we need to further characterize the set $\Gamma(H)$ (note that its cardinality is invariant under relabeling). At a high level, given $H = \pi(G)$, we can recover a natural directed, acyclic version $\mathrm{Dir}(H)$, which induces a partial order on its vertices. We can then show that $\Gamma(H)$ is precisely the set of *linear extensions* [3] of this partial order. We can then use algorithms [2, 10] developed for approximate counting of linear extensions to estimate $p_{u,v}(H)$.

We first formalize the DAG in question:

Definition 5.1 (DAG of G)**.** For G distributed according to the preferential attachment distribution (for any m), we define $\mathrm{DAG}(G)$ to be the directed acyclic graph defined on the same vertex set as G, such that there is an edge from u to $v < u$ if and only if there is an edge between u and v in G. This is just G with the directions of edges marked in accordance with the graph evolution (leading from younger nodes to older nodes).

We note here that $\mathrm{DAG}(G)$ and $\pi(\mathrm{DAG}(G))$ are exactly the same in structure and relabeling will not affect the directions, and we denote $\pi(\mathrm{DAG}(G))$ by $\mathrm{Dir}(H)$.

5.2 Integer program coefficients via $\mathrm{Dir}(H)$

The discussion in the previous subsection particularly implies that $\Gamma(H)$ is precisely the set of linear extensions of the partial order defined by $\mathrm{Dir}(H)$.

Coming back to computing the coefficients $p_{u,v}(H)$ in the integer program, we see that this task can be reduced to the problem of counting linear extensions of $\mathrm{Dir}(H)$ and of $\mathrm{Dir}(H)$ with the additional relation that $u \prec v$.

In full generality, the problem of counting linear extensions of an arbitrary partial order is classically known to be #P-complete [4]. However, there exist fully polynomial-time approximation schemes for the problem, which allow us to approximate the coefficients up to an arbitrarily small relative error [8, 9].

Proposition 5.1. *There exists a randomized algorithm which, on input H and positive number λ, outputs a sequence $\hat{p}_{u,v}(H)$ satisfying $|\hat{p}_{u,v}(H) - p_{u,v}(H)| \leq \lambda p_{u,v}(H)$ for all $u, v \in [n]$ with probability $1 - o(1)$, in time*

$$O(n^5 (\log n)^3 \lambda^{-2} \log(1/\lambda)).$$

The time complexity given in the above proposition is based on the worst case running times of the fastest known schemes [2, 10].

Given that we can approximate the coefficients $p_{u,v}(H)$ by $\hat{p}_{u,v}(H) = (1 \pm \lambda) p_{u,v}(H)$ uniformly for arbitrarily small $\lambda > 0$, the next lemma bounds the effect of this approximation on the optimal value of the integer program.

Lemma 5.1. *Consider the integer program whose objective function is given by $\hat{J}_{\varepsilon,\lambda}(\phi) = \frac{\sum_{1 \leq u < v \leq n} \hat{p}_{u,v}(H) x_{u,v}}{\sum_{1 \leq u \neq v \leq n} x_{u,v}}$, with the same constraints as in the original IP. Let ϕ_* and $\hat{\phi}_*$ denote optimal points for the original and modified integer programs, respectively. Then $|\hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*) - J_\varepsilon(\phi_*)| \leq 3\lambda$, for arbitrary $\lambda > 0$.*

Proof. Our goal is to upper bound

$$|J_\varepsilon(\phi_*) - \hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*)|.$$

We can rewrite this as

$$\begin{aligned}
&|J_\varepsilon(\phi_*) - \hat{J}_{\varepsilon,\lambda}(\phi_*) + \hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\hat{\phi}_*) + J_\varepsilon(\hat{\phi}_*) - \hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*)| \\
&\leq |\hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*) - J_\varepsilon(\hat{\phi}_*)| + |\hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\phi_*)| \\
&+ |\hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\hat{\phi}_*)|
\end{aligned}$$

Now, the first and second differences on the right-hand side are at most λ, since

$$\begin{aligned}
&|\hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\phi_*)| \\
&\qquad \le \frac{\sum_{1\le u\ne v\le n} \phi_{*u,v} |p_{u,v}(H) - (1 \pm \lambda) p_{u,v}(H)|}{\sum_{1\le u\ne v\le n} \phi_{*u,v}} \\
&\qquad \le \lambda \frac{\sum_{1\le u\ne v\le n} \phi_{*u,v}}{\sum_{1\le u\ne v\le n} \phi_{*u,v}} = \lambda.
\end{aligned}$$

The remaining difference can be estimated as follows:

$$\begin{aligned}
&|\hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\hat{\phi}_*)| \\
&\qquad \le \max\{|\hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*) - J_\varepsilon(\hat{\phi}_*)|, |\hat{J}_{\varepsilon,\lambda}(\phi_*) - J_\varepsilon(\phi_*)|\} \\
&\qquad \le \lambda.
\end{aligned}$$

This inequality is a result of the fact that ϕ_* and $\hat{\phi}_*$ are optimal points for their respective objective functions. So we took the larger of $\hat{J}_{\varepsilon,\lambda}(\phi_*)$ and $J_\varepsilon(\hat{\phi}_*)$ and increased it, leading to the upper bound.

This shows that

$$|J_\varepsilon(\phi_*) - \hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*)| \le 3\lambda,$$

so we only incur a small additive error in the optimal precision by estimating the coefficients. □

Given these results, we can efficiently upper bound the optimal precision for any given density constraint as follows: on a randomly generated input graph H, we recover its edge directions (as explained in the next section) and use them to approximate the coefficients $p_{u,v}(H)$ up to some relative error λ. Now, at this point, we have a rational linear integer program with our approximation for $pu, v(H)$. We can convert this to an equivalent truly linear integer program using a standard renormalization transformation and consider the natural linear programming relaxation, obtained by replacing the binary constraint with $x_{u,v}(H) \in [0, 1]$ for all u, v. This can be solved in polynomial time using standard tools.

6 Algorithms and Estimators

6.1 Exact recovery of $\pi(\mathrm{DAG}(G))$

Next, we show that we can efficiently recover $\pi(\mathrm{DAG}(G))$ for a preferential attachment graph G, given access to $H = \pi(G)$, via a method that we call the Peeling technique. Recovering the directions of edges from a younger node to an older node is the first step toward our goal. The algorithm starts by identifying the lowest-degree nodes (in our model, the nodes of degree exactly m) and we group them in a bin (group). Then, it removes all of these nodes and their edges from the graph. Importantly, note that, in this step, *all* of the lowest-degree nodes are identified before any of them are removed, and all of the identified nodes are removed together. The process proceeds recursively, removing the lowest-degree nodes and placing them in a new bin, until there are no more nodes to remove from the graph, as summarized in Algorithm 1 below. To construct $\mathrm{Dir}(H) = \pi(\mathrm{DAG}(G))$ during this process, we simply note that all of the edges of a given degree-m node in a given step of the peeling process must be to older nodes; hence, their orientations can be recovered (see Lemma 6.1).

Algorithm 1 Peeling Technique

```
1: procedure PEELING($H$)
2:     $t \leftarrow 1$
3:     while NoNodes($H$) $> 0$ do                          ▷ Finds no. of nodes
4:         MinDeg $\leftarrow$ FindMinDeg($H$)               ▷ Finds minimum degree
5:         $\mathcal{B}_t \leftarrow \{u : \deg(u) = \text{MinDeg}\}$     ▷ Nodes with minimum degree
6:         $H \leftarrow$ RemoveNodes($H, \mathcal{B}_t$)    ▷ Removes minimum degree nodes from $H$
7:         $t \leftarrow t + 1$
8:     end while
9:     ReverseOrder($\{\mathcal{B}_t\}$)    ▷ Reverse the indices of $\mathcal{B}_t$'s, e.g, 1 to $t$, $t$ to 1
10:    return $\{\mathcal{B}_t\}$                           ▷ Returns set of bins
11: end procedure
```

Lemma 6.1. *The* PEELING *algorithm exactly recovers* $\pi(\text{DAG}(G))$ *from* $\pi(G)$.

Proof. The PEELING algorithm maintains the following invariant at the beginning of each step: every degree-m node connects only to vertices older than itself in the remaining graph. This is clear in the initial step, since a node can only have degree exactly m in the full graph if its neighbors are all older than it is. In subsequent steps (assuming by induction that the invariant holds for all previous ones), if some edge incident with a degree-m node u is incident with a younger node v in the current graph, then this implies that some node w older than u and adjacent to u in a previous step has already been removed. This, in turn, implies that in that previous step, w had degree m and was connected to u, a younger vertex. This yields a contradiction, completing the proof. $\square$

The directed, acyclic graph $\pi(\text{DAG}(G))$ conveniently encodes the set of all vertex pairs whose true order relationship can be inferred with *complete certainty.* To make this precise, for a graph H or the resulting DAG $\text{Dir}(H)$, we define a *vertex pair order event* (u, v), for vertices $u, v \in H$, to be simply an ordered pair of distinct vertices. We interpret this as a claim that u came before v according to π^{-1}.

Definition 6.1. A vertex pair order event (u, v) is *perfect* for a graph H if, for all $\sigma \in \Gamma(H)$, we have $\sigma(u) < \sigma(v)$. Equivalently, for any random permutation σ, $\Pr[\sigma^{-1}(u) < \sigma^{-1}(v) | \sigma(G) = H] = 1$.

The following result formalizes our intuition that $\text{Dir}(H)$ captures all probability-1 information about vertex ordering in H:

Theorem 6.1 ($\text{Dir}(H)$ captures perfect vertex pair information)**.** *Consider a graph H on the vertex set $[n]$ satisfying $\Gamma(H) \neq \emptyset$. Let its DAG $\text{Dir}(H)$ be denoted by K. For any $u, v \in [n]$, the pair (u, v) is perfect for H if and only if there exists a directed path in K from v to u (denoted by $v \rightsquigarrow u$).*

Proof. First, we will show that if $v \rightsquigarrow u$, then (u, v) is perfect. This follows by showing the simpler claim that if there is an edge from v to u (denoted by $v \to u$), then (u, v) is perfect.

Let $\sigma \in \Gamma(K)$. This means that $\sigma(K)$ is a positive-probability DAG under the preferential attachment model. Note also that $\sigma(v) \to \sigma(u)$ in $\sigma(K)$, which implies that we certainly must have $\sigma(v) > \sigma(u)$ (vertices only choose to connect to older vertices). Since σ was arbitrary, we have that (u, v) is perfect for H, as desired.

Now we show the converse claim: if (u, v) is perfect for K, then $v \rightsquigarrow u$ in K. We will do this by proving the contrapositive: assume that there is no directed path from v to u. Then we will construct a permutation σ satisfying i) $\sigma \in \Gamma(K)$, and ii) $\sigma(u) > \sigma(v)$. This is equivalent to producing a feasible *schedule* of the vertices of K (i.e., a sequence of distinct vertices $v_1, v_2, ..., v_n$ of K, such that, for each $j \in [n]$, all m of the the descendants of v_j in K are contained in the set $\{v_1, ..., v_{j-1}\}$). We will require that $v_u > v_v$ in the schedule. Such a schedule gives a permutation satisfying the properties above as follows: for each j, $\sigma(j) = v_j$.

To do this, we start by considering the sub-DAG K_v, consisting of v and all of its descendants. Now, we set v_1 to be the bottom vertex of K (which is also the bottom vertex of K_v). We will add subsequent vertices to our schedule as follows: at each time step $t > 1$, $[n]$ is partitioned into three parts: $S_{p,t}$ (the vertices already in the schedule), $S_{a,t}$ (the *active* vertices; i.e., those vertices not in $S_{p,t}$, all of whose children in K are contained in $S_{p,t}$), and $S_{d,t}$ (the *dormant* vertices; i.e., those vertices that are not active or already processed). So $S_{p,1} = \{v_1\}$, $S_{a,1}$ is the set of neighbors of v_1, and $S_{d,1}$ consists of the rest of the vertices.

We observe that $S_{a,t}$ is nonempty unless $t = n$: otherwise, there are less than n vertices in $S_{p,t}$ (in fact, precisely t of them), and the rest are in $S_{d,t}$. In this case, consider the *lowest* vertex ℓ in $S_{d,t}$; ℓ cannot have any children in $S_{d,t}$, since it is the lowest, so all of its children must be in $S_{p,t}$. But this means precisely that $\ell \in S_{a,t}$. Thus, $S_{a,t}$ must be nonempty.

Note that it is clear that at any time t, we can designate any active vertex as the next one in our schedule; we would then move it to the processed set, potentially resulting in some vertices in $S_{d,t}$ becoming active.

Now, observe that at time $t = 1$, some vertex from K_v must be active (by the same reasoning that established that the active set must be nonempty). In fact, until all vertices of K_v have been processed, there remains at least one such vertex that is active. We thus choose active vertices K_v until it is entirely processed (note that we do not process the vertex $u \notin K_v$ yet, since there is no directed path from v to u). Then we process active vertices until a complete schedule has been generated. By construction, v comes earlier in the schedule than u, which completes the proof. $\square$

6.2 Estimators and their properties

Due to the high polynomial time complexity involved in solving the optimal scheme (Proposition 5.1 and complexity of linear programming), we now provide efficient estimators whose performance is close to the optimal curve. In fact, the linear program itself does not yield an optimal scheme (one has to do a rounding step, which only yields an approximation) or an estimator, but only an upper bound on the optimal precision. Moreover, converting it to an optimal estimator is computationally difficult and thus proposing efficient heuristics is warranted.

The time complexity of the estimators is dominated by the DAG construction, and is $O(n \log n)$.

Note that one must take care in designing an estimator to ensure that the resulting set of vertex pair order relations satisfies transitivity and antisymmetry.

6.2.1 Maximum-density precision 1 estimator

The estimator itself takes as input a graph H and outputs the partial order as $\mathrm{Dir}(H)$ as recovered by the Peeling algorithm. This estimator gives the maximum density among all estimators that have precision one.

In this subsection, we show that the point (density, precision) $= (0, 1)$ is a point on the optimal precision curve: namely, we prove that the perfect-precision estimator has precision one, but asymptotically negligible density, and this estimator has the maximum density among all precision one estimators. From Theorem 6.1, we know that this achieves precision 1. So, in order to prove the remainder of our claim, we need to analyze the density of this estimator; that is, we analyze the typical number of perfect pairs, culminating in the following theorem.

Theorem 6.2 (Typical number of perfect pairs)**.** *With high probability, for arbitrary fixed $m \geq 1$, the number of perfect pairs associated with G is $\Omega(n \log n)$ and $o(n^2)$ (uniformly in m). When $m = 1$, we have the matching upper bound of $O(n \log n)$, where the hidden constant in the asymptotic notation can be explicitly calculated.*

Proof. **Upper bound:** Let X_t denote the number of perfect pairs in the graph immediately after time t. We will prove the claimed upper bound by upper bounding X_t in expectation, then using Markov's inequality. To bound $\mathbb{E}[X_t]$, we will show that it is sufficient to upper bound $X(v)$ in expectation for each $v < t$, where $X(v)$ denotes the number of descendants of v in the DAG.

Using Proposition 1 of [11] with $\ell = \Theta(\log^{4/5} n \log(\log n))$, we can show that the total number of descendants $X(u)$, for all $u \leq n$, is at most $O(n/\log^{1/5} n)$, with high probability.

Now, we translate this to an upper bound on X_t as follows: we have $\mathbb{E}[X_t] \leq \mathbb{E}[X_{t-1}] + m + mO(t/\log^{1/5} t)$. This upper bound is from the following facts: at time t, all perfect vertex pairs from time $t-1$ are still perfect, contributing the X_{t-1} term. Next, vertex t makes at most m choices, creating at most m new perfect pairs (which explains the second term). Finally, the third term comes from the fact that if t chooses v, and u is a descendant of v (so that (u, v) is a perfect pair), then (u, t) is also perfect.

Solving this recurrence, we find that $\mathbb{E}[X_t] = o(t^2)$, as desired, and the proof is completed using Markov's inequality.

In the case where $m = 1$, we have a much better upper bound on $X(v)$, for arbitrary v: with high probability, at time t, $X(v) = O(\log t)$, as a result of the height of the tree being $O(\log t)$. This gives the improved bound of $\mathbb{E}[X_t] = O(t \log t)$.

Lower bound: To prove the lower bound of $\Omega(t \log t)$ perfect pairs, we write the total number of perfect pairs in such a graph as the sum, over all vertices $v \leq t$, of the number of descendants $X(v)$: $X_t = \sum_{v \leq t} X(v)$.

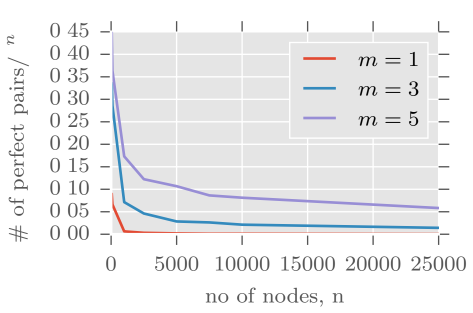
In the $m = 1$ case (where G is a tree), this implies that X_t is the *total path length* of the tree. It is known that, with high probability, this parameter is $\Theta(t \log t)$ (since it can be written as an additive parameter, the results of [16] apply).

Now, to prove it for general m, we recall that to form a sample G from $\mathcal{PA}(n, m)$, we first produce a sample T from $\mathcal{PA}(mn, 1)$, then collapse sequences of m consecutive vertices. The number of perfect pairs in G is at least the same quantity in T, divided by m^2, since there can be at most m^2 edges from one sequence of m consecutive vertices to an older sequence in T. Using the result for $m = 1$, we have that with high probability, the number of perfect pairs in T is $\Theta(mt \log(mt)) = \Theta(t \log t)$, which implies that the number of perfect pairs in G is $\Omega(t \log t)$. □

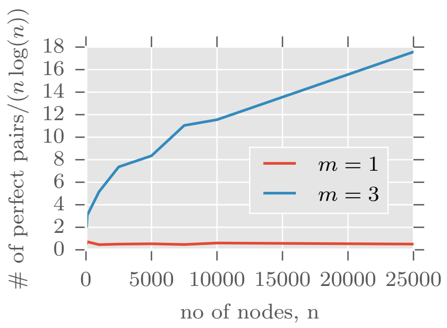
The above theorem implies that the density of the perfect pair estimator is asymptotically 0, for arbitrary m. This is validated by simulation as shown in Figure 1. [1]

Further scrutiny of the numerical evidence leads to a conjecture about the more precise behavior of the number of perfect pairs as a function of m: we conjecture that for $m > 1$, the number of perfect pairs is $O(n^{1+\delta(m)})$, for some function $1 > \delta(m) > 0$ (see again Figure 1).

[1]All simulation results take empirical expected values with respect to several samples from the graph distribution in question.



(a) Upper bound



(b) Lower bound

Figure 1: Normalizations of the number of perfect pairs in synthetic preferential attachment graphs.

This conjecture arises from the intuition from the (rigorous) proof of Theorem 6.2 as follows: denote by $X(u)$ the number of descendants of any vertex u in $\mathrm{DAG}(G)$, and by X_t the number of perfect pairs in the graph up to time t. If we can upper bound the number of descendants of any given vertex at time t by $O(t^c)$, for some constant $c = c(m) < 1$ dependent only on m (which is supported by empirical evidence), then we have $\mathbb{E}[X_t] \le \mathbb{E}[X_{t-1}] + m + O(t^c) = \mathbb{E}[X_{t-1}] + O(t^c)$. This upper bound arises as follows: the pairs that are perfect at time $t - 1$ remain perfect at time t, yielding the first term of the upper bound. The second and third terms arise from the fact that vertex t connects to at most m other vertices (resulting in m additional perfect pairs), and each perfect pair (v, w) involving two descendants of a node chosen by vertex t yields another perfect pair (v, t).

Iterating this recurrence and using Euler-Maclaurin summation, we find that $\mathbb{E}[X_t] = O(t^{1+c(m)})$, as desired. We give empirical evidence for the upper bound on $X(t)$ (and hence for the improved upper bound on the number of perfect pairs) in Figure 2, which indicates that the maximum number of descendants of any node at a given time t is $O(t^{c(m)})$.

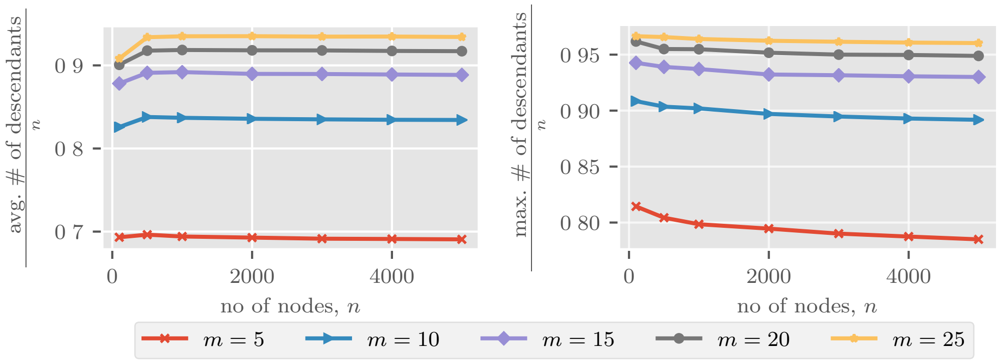


Figure 2: Normalized average and maximum number of descendants of degree-m nodes in $\mathrm{DAG}(G)$.

Theorem 6.2 gives us a single point on the optimal precision curve, and it is very simple to show that the curve is decreasing as ε increases. In the next section, we show Peeling estimator achieves nontrivial density and precision with high probability, which gives a lower bound on the optimal precision for another value of the density. We will use this to give an empirical indication of the tightness of our upper bound.

6.2.2 Peeling: A linear binning estimator via peeling

When the term Peeling is used as an estimator, we mean the linear binning estimator[2] from the bins (groups of nodes) given by the Peeling Algorithm 1. In particular, the sequence of subsets of vertices removed during each step naturally gives a partial order: each such subset forms a bin, and bins that are removed earlier are considered to contain younger vertices.

We can start by showing a crude estimate of the precision and density of the Peeling estimator.

Theorem 6.3 (Non-negligible precision and density of the peeling estimator)**.** *For each $m \geq 1$ the* Peeling *estimator has precision and density* $\Theta(1)$.

Proof. The precision claim follows by showing that, with high probability, there exist $\Theta(n)$ vertices in the last $\Theta(n)$ timesteps that are never chosen (so that they are removed in the first bin and, thus, correctly declared to be younger than $\Theta(n)$ other vertices). This also proves the density claim.

The required result follows easily from the fact that with high probability there are $\Theta(n)$ vertices of degree m in the graph of size n. Namely, suppose that there are J degree-m vertices. We consider the $J/2$th such vertex (say, vertex v). Then, clearly, since $J/2 = \Theta(n)$, we have that $n - J/2$ is also $\Theta(n)$, so that the result follows. □

We now discuss further analysis of the procedure. In the case $m = 1$, the original graph G is a tree, and we are able to precisely characterize certain parameters of the Peeling procedure, owing to the large amount of literature surrounding the preferential attachment distribution for this case (the structure is (essentially) identically distributed to a random *plane-oriented recursive tree*): for instance, the asymptotic size of the jth youngest bin, for each fixed j, may be precisely determined by writing it as an additive parameter and using results from Wagner et al. [16]. This gives us a complicated but explicitly computable convergent series expression for the typical asymptotic density of the estimator in this case. Precise theoretical results for $m > 1$ (and for the precision in the $m = 1$ case) are much more elusive, owing to the fact that the precision and the DAG structure of the graph fail to lead to clean recursive formulas for parameters of interest (though a Pólya urn approach allows us to derive very complicated expressions involving arbitrarily high moments and covariances of quantities of interest); however, we can show that the peeling process recovers all perfect pairs, in addition to many imperfect ones.

7 Experiments

In this section, we evaluate our methods on synthetic and real-world networks. In what follows, $\sigma_{\text{perf}}, \sigma_{\text{peel}}, \sigma_{\text{peel}+}$ denote the partial orders produced by the Perfect-precision, Peeling, and Peeling+ algorithms.

7.1 Synthetic graphs

We derive significant insight by studying the Peeling method empirically. Table 1 and Figure 3

[2]A *Linear binning* is a special type of partial order, defined as follows: a partial order $\prec$ is a linear binning if its elements may be partitioned into subsets $\mathcal{B}_1, ..., \mathcal{B}_j$, for some j, such that $u \prec v$ if and only if there is a pair $i < j$ such that $u \in \mathcal{B}_i$ and $v \in \mathcal{B}_j$. This is equivalent to a graded partial order.

m	$\theta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel}})$	$\delta(\sigma_{\text{peel}})$
5	0.954	0.758	0.794
10	0.971	0.858	0.884
25	0.986	0.936	0.949
50	0.993	0.967	0.974

Table 1: Comparison for $\mathcal{PA}(n = 5000, m)$

shows simulation results for samples from the Barabási-Albert model. The performance improves greatly with even small increases in m. This can be explained intuitively as follows: for small m, as the graph evolves, each new vertex is likely to connect to high-degree nodes, which are already in older bins in DAG(G). Thus, bins tend to be large, resulting in low precision and recall. For larger m, each new node chooses a degree-m node (say, v) with higher probability, since a constant fraction of all nodes have degree m and the number of choices that each new vertex makes increases with m. A new vertex choosing v results in shifting the bin indices of all descendants of v. Thus, larger m leads to a DAG with a more equitable distribution of vertices across bins, which yields higher precision and recall.

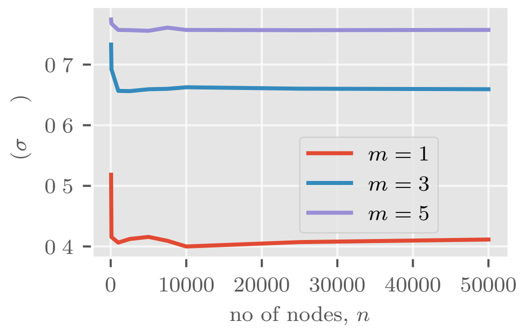


Figure 3: Recall of the Peeling algorithm remains constant when the number of nodes varies

7.2 Real-world networks

All datasets used in our study, except the human brain networks, are publicly available. We explain later how the brain network is formed. The ArXiv, Wikipedia and DBLP are collected from [5]. The first three networks have partial temporal information available, and the original ranking of the network is deduced from the time stamps of the edges. For the brain networks, no explicit temporal information is available.

The ArXiv network. Directed network; the nodes are publications and an edge indicates one publication citing another. The original ranking is not a full ranking; it has 1457 bins.

The Simple English Wikipedia dynamic network. Directed network; it shows the evolution of hyperlinks between articles of the Simple English Wikipedia. Nodes represent articles and an edge indicates that a hyperlink was added or removed.

DBLP computer science bibliography:. Undirected network; an edge between two authors represents a common publication.

Human brain network - Cambridge-Buckner. It is an undirected network. The nodes here are the regions in the human brain, and the edges represent communication between two regions while performing an activity. The network is formed as follows. We collect the human brain fMRI data at resting state, from the Cambridge-Buckner dataset shared by NITRC[3]. An initial network is first formed with nodes as voxels, and there are $243,648$ voxels. Each voxel is associated with a time series data that lasted approximately 350 seconds. We compute the Pearson correlation coefficient between the time series of every pair of voxels. If the correlation is greater than 0.8 we form an edge between the voxels. Finally we form a network of 56 regions, which is a contracted network of voxels - all voxels corresponding to a region are merged to form a node, and an edge is created between two regions when there exists at least one edge between member voxels.

Human brain network - Human Connectome project. This is the resting state fMRI data from Human Connectome project focusing on the cortex area. The Human Connectome project provides a clean and refined data, which gives consistent results in many published studies. We processed and formed brain networks out of 100 healthy young adults. First, the cortex brain data corresponding to 100 subjects (2 sessions per subject, and 2 scans per session) is parcellated into 180 regions per hemisphere using a procedure described in [7]. Then the correlation matrices are formed from the time series of the parcellated data. Finally binary, undirected networks are constructed from the correlation matrices as follows: a spanning tree is created first from the complete network of the correlation matrix, and later k-nearest neighbors (higher correlation values) of each node are added into this network, where k is chosen as 10 in our case (see [1] for different ways to generate networks from the brain data).

Each network has 300 nodes, which are regions or clusters formed from group-Independent Component Analysis. The data is in correlation matrix format, with each element as the Gaussianized version of the Pearson correlation coefficient (Fisher Z transform with AR(1) process estimation). In order to form a binary adjacency matrix, we use a threshold just high enough to make the resulting graph connected. Such a graph is sparse.

Facebook Wall network. Recently, strategic information spreading in online social networks like Twitter or Facebook is alleged to create biases in the opinions of people, and skews the political elections results. In most cases it is difficult to track the culprits behind the false information. Our methods to rank the nodes according to their arrival finds applications here. In general, in an online social network the identification of malicious information sources and carriers, be it in the case of an online spam spreading in the Internet or a misinformation or rumor propagating, allows timely quarantine of the epidemic-like spreading to mitigate the damage caused. In particular, we study the network of Facebook wall posts here, where nodes are the users and the edges are wall posts between users on the social network Facebook located in the New Orleans region [18]. Any friend of a given user can see all posts on that user's wall, so communication is public among friends. An edge (u, v, t) means that user u posted on user v's wall at time t. We could limit to the network of wall posts containing a specific hashtag, and thus focus on rumor spreading on a certain topic.

SMS-A. Short messaging service (SMS) is a texting service provided on mobile phones. In this dataset, an edge (u, v, t) means that person u sent an SMS message to person v at time t [19].

[3]https://www.nitrc.org

7.2.1 Results

For all real networks above for which an approximate ground truth is known, our algorithms yield excellent precision and recall results, consistent with our theoretical predictions (see Table 2, and Figures 4 and 5) in the main paper). Note that the precision and recall measures achieved on these datasets using our estimators are somewhat pessimistic, due to the fact that the only ground truths available are sparse partial orders. The table shows the performance of the Peeling+ algorithm, in addition to that of Peeling. When the density of the recovered partial order by Peeling algorithm is low, the recall can be improved via Peeling+ with a slight loss in precision (see the Wikipedia result).

For the brain networks, no ground truth is known, so we cannot give measures of precision or recall for the generated orderings. We propose our orderings for further study. We discuss the results on these networks below.

Human brain network - Cambridge-Buckner. The purpose here is to predict the evolutionary order among the important regions inside the brain. Figure 3A (left) presents the bins of regions we found via Peeling algorithm. Figure 3B (right) shows an image of the human brain with prominent regions annotated with bin number. Regarding plausibility of our proposed order, we first note that prior studies support preferential attachment involvement in brain networks [17]. Moreover, the placement of certain brain regions in the ordering are consistent with known results. E.g., the corpus callosum, which joins the two hemispheres of the brain, and the cerebellum and most of its related structures, which are present in all vertebrates, are supposed to have developed at the earliest stages of brain evolution. Another example is the uncinate fasciculus, the last white matter tract to be evolved in the human brain, which is in compliance with our result. We consider this as a first step towards finding a proper evolutionary order of brain regions, and corroborating evidence (which is currently lacking) must come from alternative methodologies devised by domain experts. This analysis could separate regions with core functionality from those involved in functional specializations, giving information about evolution of various structural and functional elements.

Human brain network - Human Connectome project. We analyzed this dataset in order to demonstrate that our methods are robust across networks coming from different individuals. We plot the histogram of arrival rank of prominent auditory, visual, and somatosensory regions. The histograms show a concentration, indicating consistency in ranking of the respective regions. Moreover, we observe that most of the regions that serve as prime functionaries among auditory, visual and somatosensory regions have consistently early arrival order in the 100 brain networks we took into account. This is in accordance with the typical notion of early arrivals of these regions in the human brain evolution.

References

[1] A. F. Alexander-Bloch, N. Gogtay, D. Meunier, R. Birn, L. Clasen, F. Lalonde, R. Lenroot, J. Giedd, and E. T. Bullmore. Disrupted modularity and local connectivity of brain functional networks in childhood-onset schizophrenia. *Frontiers in systems neuroscience*, 4:147, 2010. 17

[2] J. Banks, S. Garrabrant, M. L. Huber, and A. Perizzolo. Using TPA to count linear extensions. *arXiv preprint arXiv:1010.4981*, 2010. 8, 9

[3] G. Brightwell and P. Winkler. Counting linear extensions. *Order*, 8(3):225–242, 1991. 8

[4] G. Brightwell and P. Winkler. Counting linear extensions is #p-complete. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing (STOC)*, pages 175–181, New York, NY, USA, 1991. ACM. 9

[5] K. T. K. N. Collection. Preferential attachment exponent. Available on `http://konect.uni-koblenz.de/statistics/prefatt` [Accessed: 30 April 2017], 2016. 16

[6] A. Frieze, W. Pegden, et al. Looking for vertex number one. *The Annals of Applied Probability*, 27(1):582–630, 2017. 2

[7] M. F. Glasser, T. S. Coalson, E. C. Robinson, C. D. Hacker, J. Harwell, E. Yacoub, K. Ugurbil, J. Andersson, C. F. Beckmann, M. Jenkinson, et al. A multi-modal parcellation of human cerebral cortex. *Nature*, 536(7615):171–178, 2016. 17

[8] M. Huber. Fast perfect sampling from linear extensions. *Discrete Mathematics*, 306(4):420–428, 2006. 9

[9] M. R. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986. 9

[10] A. Karzanov and L. Khachiyan. On the conductance of order markov chains. *Order*, 8(1):7–15, 1991. 8, 9

[11] T. Łuczak, A. Magner, and W. Szpankowski. Asymmetry and structural information in preferential attachment graphs. *Random Structures & Algorithms (to appear); arXiv preprint arXiv:1607.04102*, 2019. 2, 7, 13

[12] A. Magner, A. Grama, J. Sreedharan, and W. Szpankowski. Recovery of vertex orderings in dynamic graphs. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2017. 7, 8

[13] A. Magner, A. Grama, J. K. Sreedharan, and W. Szpankowski. Recovery of vertex orderings in dynamic graphs. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1563–1567, June 2017. 2

[14] A. Magner, J. K. Sreedharan, A. Y. Grama, and W. Szpankowski. Times: Temporal information maximally extracted from structures. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 389–398, 2018. 2

[15] S. M. V. Mahantesh, S. Iyengar, M. Vijesh, S. R. Nayak, and N. Shenoy. Prediction of arrival of nodes in a scale free network. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 517–521, 2012. 2

[16] D. Ralaivaosaona and S. Wagner. Additive functionals of d-ary increasing trees. In *Proceedings of the 27th International Conference on Probabilistic, Combinatorial, and Asymptotic Methods for the Analysis of Algorithms*, pages 1–12, 2016. 13, 15

[17] P. E. Vértes, A. F. Alexander-Bloch, N. Gogtay, J. N. Giedd, J. L. Rapoport, and E. T. Bullmore. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences*, 109(15):5868–5873, 2012. 18

[18] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 37–42. ACM, 2009. 17

[19] Y. Wu, C. Zhou, J. Xiao, J. Kurths, and H. J. Schellnhuber. Evidence for a bimodal distribution in human communication. *Proceedings of the national academy of sciences*, 107(44):18803–18808, 2010. 17