TIMES: Temporal Information Maximally Extracted from Structures

Abram N. Magner*
anmagner@illinois.edu
UIUC
Urbana, IL, USA

Jithin K. Sreedharan*
jithinks@purdue.edu
Purdue University
West Lafayette, IN, USA

Ananth Y. Grama
ayg@cs.purdue.edu
Purdue University
West Lafayette, IN, USA

Wojciech Szpankowski
spa@cs.purdue.edu
Purdue University
West Lafayette, IN, USA

ABSTRACT

Inferring the node arrival sequence from a snapshot of a dynamic network is an important problem, with applications ranging from identifying sources of contagion to flow of capital in financial transaction networks. Variants of this problem have received significant recent research attention, including results on infeasibility of solution for prior formulations. We present a new formulation of the problem that admits probabilistic solutions for broad classes of dynamic network models. Instantiating our framework for a preferential attachment model, we present effectively computable and practically tight bounds on the tradeoff curve between optimal achievable precision and *density*/recall. We also present efficient algorithms for partial recovery of node arrival orders and derive theoretical and empirical performance bounds on the precision and density/recall of our methods in comparison to the best possible. We validate our methods through experiments on both synthetic and real networks to show that their performance is robust to model changes, and that they yield excellent results in practice. We also demonstrate their utility in the context of a novel application in analysis of the human brain connectome to draw new insights into the functional and structural organization and evolution of the human brain.

Keywords. Preferential attachment graphs, partial order, linear extension, integer programming

ACM Reference Format:
Abram N. Magner, Jithin K. Sreedharan, Ananth Y. Grama, and Wojciech Szpankowski. 2018. TIMES: Temporal Information Maximally Extracted from Structures. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186105

1 INTRODUCTION

The problem of inferring the arrival order of nodes in a dynamic network, from its observed current state, is of considerable significance. In a network of financial transactions, the arrival order tracks the flow of capital. In mapping spread of infectious diseases, arrival order allows one to identify early patients, yielding clues to genetic origin, evolution, and mechanisms of transmission. In social networks, one can map the spread of information, and use it to design personalized channels by prioritizing information of shared interest. In networks of biochemical interactions (e.g., protein interaction networks), one can identify early biomolecules (e.g., proteins in prokaryotic species), which are known to be preferentially implicated in cancers and other diseases [20].

Owing to this significance, and the availability of large datasets in diverse application domains, there has been considerable recent research interest in variants of this problem. Despite the apparent simplicity of the informal problem statement, the node arrival inference problem is highly complex, both from analysis and methodological points of view. This complexity stems, in part, from the symmetries inherent in graphs and the models that generate them, which admit multiple (partial) node orderings with equal likelihood. Characterizing these symmetries and suitably reformulating the problem to establish limits on inference is at the core of our analysis framework. Efficient methods that achieve the prescribed limits of inference characterize the algorithmic challenges associated with the problem.

Dynamic graph models underlying a large class of applications primarily consider node arrivals, along with corresponding edge insertions (without corresponding node departures)[1]. Starting from a snapshot of a dynamic network, the problem of inferring the arrival order inverts the arrival process. Consequently, a specific inference technique is intimately coupled with a corresponding model of the arrival process. In this paper, we assume that networks evolve according to some stochastic model, which provides the basis for node order inference, with associated probabilistic guarantees. We present a general framework for inferring node orders, and a specific instantiation of our framework in the context of a preferential attachment model. We note that our framework is general, and capable of admitting a large class of dynamic graph models. Our specific instantiation of preferential attachment graphs is directly applicable to a large number of applications, since the model has been extensively studied in literature. We will primarily consider the following form of the preferential attachment model:

Barabási-Albert model. Let $\mathcal{PA}(n, m)$ denote the following version of the Barabási-Albert model [2]: the parameter n represents the number of nodes in the network and m is the number of connections a new node makes to existing nodes when it is added to the network. To form graph G_t at time t, for $t = 1$, a single vertex (called 1) is created with m self loops. At time $t > 1$, vertex t joins the network and makes m connections to existing nodes in graph G_{t-1} with probability proportional to their current degrees as follows: each connection choice is conditionally independent of the

*These authors contributed equally to this work.

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN ISBN 978-1-4503-5639-8/18/04.
https://doi.org/10.1145/3178876.3186105

[1] In many applications, for instance, brain coactivity, protein interaction, and citation networks, departures do not play a significant role. In other domains, it is important to consider departures – we do not directly address these domains in our current work.

others and satisfies $\Pr[t \text{ connects to } k | G_{t-1}] = \frac{\deg_{t-1}(k)}{2m(t-1)}$, where $\deg_{t-1}(k)$ is the degree of node k at time $t-1$. This process continues until t equals n^2. This paper primarily focuses on $\mathcal{PA}(n, m)$, unless otherwise stated explicitly. We remark that the existence of self-loops on the initial vertex allows for clean proofs; our theoretical and empirical results extend without significant changes to models in which these self-loops are not present.

In our previous work [16], we present a formulation of the problem of recovering the temporal order of nodes in preferential attachment graphs. Focusing on deriving fundamental limits on inference of temporal order, we show that any estimator that outputs a *total* ordering of vertices, with high probability, makes a large number of mistakes (to be made precise later) – owing to inherent symmetries in the network model. This negative result provides the motivation for this work.

Our Contributions. In this paper, we present the following key results:

- We present a relaxed formulation of the problem that aims to infer a *partial order* on nodes (recall that a partial order satisfies antisymmetry and transitivity), which is meant to approximate the true arrival order. The intuition is that we allow an estimator to make fewer vertex pair order inferences, in exchange for higher precision.
  We present detailed analytical results on the limits achievable in this new formulation in terms of the tradeoff between expected precision and partial order density. In particular, we derive tight, efficiently computable linear programming bounds on the optimal precision achievable when our estimator is required to make a certain minimum number of vertex pair order inferences.
- We show that maximum likelihood estimation of a total order of arrivals leads to an exponentially large set of equiprobable solutions, making it an unfavorable notion of optimality.
- Our computable bounds on the optimal precision curve do not yield efficient partial order estimators. For this reason, we introduce and analyze, both theoretically and empirically, efficient estimators: the first is optimal in the sense that it yields an estimator with *perfect* precision (see Theorem 3.5): that is, given a relabeled preferential attachment graph, it infers all vertex order relations that hold with probability 1. We rigorously analyze the number of such relations and find them to be asymptotically small (Theorem 5.1). This motivates our investigation of other algorithms (the Peeling and Peeling+ algorithms), which sacrifice some precision in order to achieve higher density.
- We present experimental evaluation on both real and synthetic datasets. These experiments demonstrate the robustness of our methods to variations in the preferential attachment model (see Section 4) and provide guidance for future theoretical exploration. 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. This analysis could separate regions with core functionality from those involved in functional specializations, giving information about evolution of various structural and functional elements.

[2]We will drop the subscript n in G_n if it is implied from the context.

Prior Related Work. Several prior results focus on variants of the problem of finding the oldest node in a graph [5, 10]. Bubeck et al. [5] consider uniform attachment and preferential attachment trees, and find a set that contains the root node with a prescribed error probability. They prove that the set size is independent of the number of nodes. However, their technique cannot be readily extended to the case $m > 1$. Our proposed algorithms target a more general problem, and can be used to find such a set for the oldest node. For preferential attachment graphs, Frieze et al.[10] 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.

A related problem of detecting information sources in epidemic networks has been studied by Shah et al. [19] for the Susceptible-Infected model and Zhu et al. [22] for the Susceptible-Infected-Recovered model. The more general problem of inferring node arrival orders has been shown to be particularly challenging by us in a recent work [16], in which we derive fundamental bounds for exact and approximate recovery of vertex ordering in preferential attachment graphs. Recognizing this complexity, in this paper we focus on analysis and algorithmic approaches for partial ordering of vertices. Mahantesh et al. [17] 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.

2 PROBLEM FORMULATION

Let G be a graph drawn from the model $\mathcal{PA}(n, m)$. Note that, by definition, the vertices of G are precisely the integers $[n] = \{1, \ldots, n\}$, where node j was the jth node to arrive. Next, the nodes are subjected to a permutation π drawn uniformly at random from the symmetric group S_n, and we are given the graph $G' := \pi(G)$; that is, the nodes of G are randomly *relabeled*. This paper aims to develop methods for inferring arrival order in G after observing G', i.e., to find π^{-1}. The permutation π^{-1} can be considered as the true arrival order of the nodes of the given graph; that is, $\pi^{-1}(u)$ for a node u in G' gives the arrival time of that node in the original graph process.

In our recent work [16], a solution to this problem takes the form of an estimator function $\phi : \mathfrak{G}_n \to S_n$, where S_n is the symmetric group on n letters and $\mathfrak{G}_n$ is the set of graphs on n vertices. We evaluated estimators using the expected Kendall τ distance between $\phi(\pi(G))$ and π^{-1} as a metric. In particular, they showed that *any* such estimator makes, with high probability, $\Theta(n^2)$ inversion errors (note that the maximum possible number of such errors is $\binom{n}{2} = \Theta(n^2)$), owing to the existence of a superexponentially large number of equiprobable graphs with the same structure as $\pi(G)$.

In view of this negative result, we relax the problem: instead of requiring that an estimator infer a total order (i.e., a permutation), we study estimators that output *partial* orders on the set of vertices. For a partial order σ, a relation $u <_\sigma v$ indicates a guess that

$\pi^{-1}(u) < \pi^{-1}(v)$. The advantage of allowing partial orders is that the ability of an estimator to refrain from guessing the relative order of some vertex pairs allows for the possibility of greater *precision*. An example of our new setting is given in Figure 1.

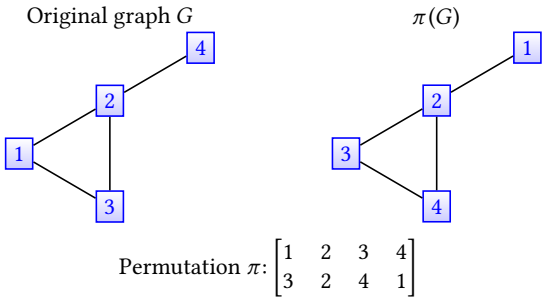


Figure 1: An example scenario. The estimator sees only $\pi(G)$ and must infer π^{-1}. E.g., it may output the order $\sigma = \{4 \prec 1 \prec 2\}$ The relation $4 \prec 1$ is correct, since $\pi^{-1}(4) = 3 < \pi^{-1}(1) = 4$, but the relations $4 \prec 2$ and $1 \prec 2$ are incorrect, since $\pi^{-1}(4) = 3 > \pi^{-1}(2) = 2$ 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 $\rho(\sigma) = \theta(\sigma)\delta(\sigma) = 1/6$. (See below for definitions of δ, θ and ρ.)

In evaluating a partial order σ as a solution to our problem, we say that an ordered pair of vertices (u, w) in $\pi(G)$ satisfying $u <_\sigma w$ forms a *correct pair* if $\pi^{-1}(u) < \pi^{-1}(w)$. A pair of vertices is *unguessed* with respect to σ if they are unrelated in σ. Note that, given a partial order σ, we can always algorithmically find a total order consistent with σ (i.e., a linear extension of σ) and containing at least as many correct pairs (but generally with more incorrect pairs as well).

A partial order may be described by a directed, acyclic graph (DAG) on the set of vertices. We find it useful to describe a partial order by a *binning*, which is simply a more compact representation of a DAG: vertices within a given bin are unrelated in the partial order, while an edge from a bin $\mathcal{B}_1$ to another one, say, $\mathcal{B}_2$, indicates that $v_1 <_\sigma v_2$ for every $v_1 \in \mathcal{B}_1$ and $v_2 \in \mathcal{B}_2$.

Why is maximum likelihood estimation not a good approach? In the case of total order estimators, a natural way to approach the problem is to frame it in terms of maximum likelihood estimation (MLE) as follows: $C_{\mathrm{ML}}(H) = \arg\max_{\sigma \in S_n} \Pr[\pi^{-1} = \sigma | \pi(G) = H]$. The following proposition gives a characterization of the optimal solution set C_{ML}. Its proof sketch can be found in the Section 5.3.

Proposition 2.1 (MLE of total orders). *The maximum likelihood estimation solution set $C_{\mathrm{ML}} = C_{\mathrm{ML}}(\pi(G))$ satisfies $|C_{\mathrm{ML}}| = e^{n \log n - O(n \log\log n)}$ with high probability.*

Thus, C_{ML} gives a large number of equiprobable solutions and the maximum likelihood formulation is not the appropriate approach for the problem.

Measures for evaluating partial order estimators. In order to evaluate estimators that output partial orders, we define a few measures that capture important aspects of their performance: 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 the notation $u <_\sigma v$ means that u is less than v according to the partial order σ.

Density: The density of a partial order σ is simply the number of comparable pairs, normalized by the total possible number, $\binom{n}{2}$. That is, $\delta(\sigma) = \frac{K(\sigma)}{\binom{n}{2}}$. Note that $\delta(\sigma) \in [0, 1]$. Then the density of a partial order estimator ϕ is simply its minimum possible density $\delta(\phi) = \min_H[\delta(\phi(H))]$.

Measure of precision: This measures the expected fraction of *correct* pairs out of all pairs that are guessed by the partial order. It is given by

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

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

The previous two measures capture the salient behaviors of interest in an estimator. For convenience, we will sometimes refer to an additional measure, which differs from the precision in the choice of normalization:

Measure of recall: This measure gives the expected fraction of correct pairs (out of the total number, $\binom{n}{2}$) given by the algorithm that outputs the partial order σ. It can be interpreted as a modified version of the Kendall τ metric for partial orders. It is defined as

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

There is a simple relation among the different measures: for a given partial order σ, we have $\rho(\sigma) = \theta(\sigma) \cdot \delta(\sigma)$.

Our main objective is then to seek an estimator that is optimal in the following sense: for an input parameter $\varepsilon \in [0, 1]$, we wish to find an estimator ϕ which has $\delta(\phi) \geq \varepsilon$ and maximum possible precision $\theta(\phi)$. This then yields an *optimal curve* $\theta_*(\varepsilon)$, that characterizes the tradeoff between precision and density. The main results of this paper (explained in the next section) derives computable bounds on this curve and efficient heuristic estimators that approach it.

Remark 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 [16].*

3 MAIN RESULTS

Our main results concern the approximation of the optimal precision function $\theta_*(\varepsilon)$ defined in the previous section. We start by expressing the precision of a given estimator ϕ as a sum over all graphs H:

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

where the conditional expectation is with respect to the randomness in π and G. Note, however, that, under the conditioning, once π is fixed, then so is G. Note that we were able to take $\frac{1}{K(\phi(H))}$ outside the expectation because it is constant with respect to π.

This equation shows that, 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

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

subject to the density constraint that $K(\phi(H)) \geq \varepsilon\binom{n}{2}$.

To do this, we derive a useful representation for this optimization problem as an integer program. There are various ways to do this, but our method allows for simple upper bounds via linear programming relaxations.

3.1 Integer program formulation

To 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 $u <_{\phi(H)} v$. Then, we can rewrite the objective function (1) in terms of these variables by expressing the cardinalities in the numerator and denominator as sums of these indicators. Using linearity of expectation, we set:

$$J_\varepsilon(\phi) = \frac{\sum_{1\leq u<v\leq n} p_{u,v}(H)x_{u,v}}{\sum_{1\leq u\neq v\leq n} x_{u,v}}, \quad (2)$$

where $p_{u,v}(H) = \Pr[\pi^{-1}(u) < \pi^{-1}(v)|\pi(G) = H]$. We have the following constraints, coming from the fact that the outputs of our estimators are partial orders 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 $u,v \in [n]$.

The above integer program is rather explicit, except for the coefficient $p_{u,v}(H)$. The next lemma gives a combinatorial formula for the probability $p_{u,v}(H)$. Associated with permutations of the random graph model, we define the following terms:

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.

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

Lemma 3.1 (Expression for coefficients of the optimal precision integer program). *For all $v,w \in [n]$ and graphs H,* $\Pr[\pi^{-1}(v) < \pi^{-1}(w)|\pi(G) = H] = \frac{|\{\sigma\ :\ \sigma^{-1}\in\Gamma(H),\sigma^{-1}(v)<\sigma^{-1}(w)\}|}{|\Gamma(H)|}$.

The proof is given in Section 3.1.

We thus end up with an integer program written in terms of coefficients that are (in principle) explicitly computable, and the expression in Lemma 3.1 can be used to guide intuition in the design of more efficient heuristic methods. In the next subsection, we define the directed version of a sampled graph $\pi(G)$ and explain how to efficiently recover it from $\pi(G)$. This will be essential in the calculation of the coefficients $p_{u,v}(H)$, and it will yield efficient heuristic partial order estimators.

Remark 2. *The integer program formulation and Lemma 3.1 hold for any random graph model in which two positive-probability graphs that are isomorphic are equiprobable.*

3.2 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* of this partial order. We can then use algorithms developed for approximate counting of linear extensions to estimate $p_{u,v}(H)$.

We first formalize the DAG in question:

Definition 3.2 (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 (incident on an existing node from a new node). [3]. For an arbitrary permutation σ with $H = \sigma(G)$, we denote by $\mathrm{Dir}(H) = \sigma(\mathrm{DAG}(G))$.

3.2.1 **Exact recovery of $\pi(\mathrm{DAG}(G))$.** Next, we show that we can efficiently recover $\pi(\mathrm{DAG}(G))$, given access to $H = \pi(G)$, via a method that we call the Peeling technique. *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.* Hence 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 places them in the youngest bin. Then, it removes all of these nodes and their edges from the graph. The process proceeds recursively, removing the lowest-degree nodes and placing them in the next-youngest bin, until there are no more nodes to remove from the graph. 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 3.3 and Section 5.5 for its proof. An example graph and its DAG with the indication of bins are shown in Figure 2.

Lemma 3.3 (Reconstruction of $\pi(\mathrm{DAG}(G))$). *The Peeling algorithm exactly recovers $\pi(\mathrm{DAG}(G))$ from $\pi(G)$.*

The directed, acyclic graph $\pi(\mathrm{DAG}(G))$ 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 $\mathrm{Dir}(H)$, 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 3.4 (Perfect vertex pair order event). 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, recalling that π was the uniformly random permutation used to relabel the vertices of G, $\Pr[\pi^{-1}(u) < \pi^{-1}(v)|\pi(G) = H] = 1$.

[3] In reference to the DAG of a preferential attachment graph, we ignore self-loops, without loss of generality.

Algorithm 1 Peeling Technique

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

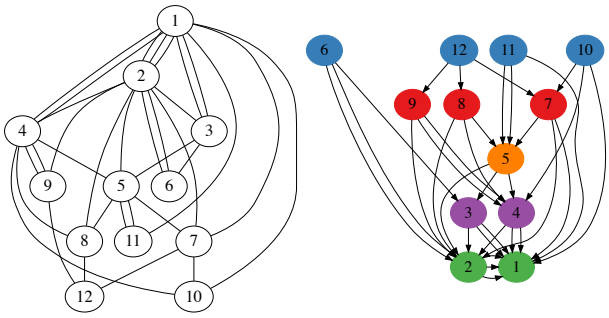


Figure 2: Graph (left) and its DAG (right) where the same bin nodes have the same colors.

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

THEOREM 3.5 (Dir(H) CAPTURES PERFECT VERTEX PAIR INFORMATION). *Consider a graph H on the vertex set $[n]$ satisfying $\Gamma(H) \neq \emptyset$. Let its DAG* Dir(H) *be denoted by K. For any $u, v \in [n]$, (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 $v \rightarrow u$), 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 $\sigma(v) \rightarrow \sigma(u)$ 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 $v_1, v_2, \ldots, v_n$ 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 $\{v_1, \ldots, v_{j-1}\}$). 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. □

3.3 Integer program coefficients via 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 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 Dir(H) and of 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.

PROPOSITION 3.6 (FPTAS FOR APPROXIMATING $p_{u,v}(H)$). *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 [1, 13].

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

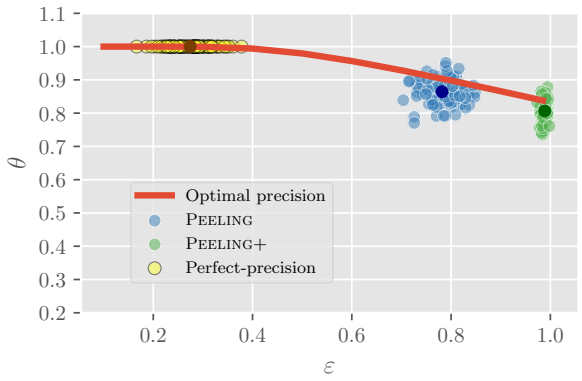


Figure 3: LP relaxation to the optimal precision curve $\theta_*(\varepsilon)$ and estimators for $G \sim \mathcal{PA}(n = 50, m = 3)$. The bold points indicate averaged value.

lemma bounds the effect of this approximation on the optimal value of the integer program.

Lemma 3.7 (Effect of perturbation of $p_{u,v}(H)$). *Consider the integer program whose objective function is given by* $\hat{J}_{\varepsilon,\lambda}(\phi) = \frac{\sum_{1\le u<v\le n} \hat{p}_{u,v}(H) x_{u,v}}{\sum_{1\le u\ne v\le n} x_{u,v}}$, *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 we have* $|\hat{J}_{\varepsilon,\lambda}(\hat{\phi}_*) - J_\varepsilon(\phi_*)| \le 2.5\lambda$, *for arbitrary* $\lambda > 0$.

This particularly implies that we can approximate the optimal objective function value to an arbitrarily small relative error, provided that the true value is bounded away from 0.

Given these results, we can efficiently approximately upper bound the optimal precision for any given density constraint as follows: on a randomly generated input graph H, we recover its edge directions and use them to approximate the coefficients $p_{u,v}(H)$ up to some relative error λ (using Proposition 3.6). Now, at this point, we have a rational linear integer program with estimates for the coefficients. We can convert this to an equivalent truly linear integer program using a standard renormalization transformation (we will not explicitly indicate this transformation in what follows).

We then consider the natural linear programming relaxation, obtained by replacing the binarity constraint with $x_{u,v}(H) \in [0, 1]$ for all u, v. This can be solved in polynomial time using standard tools. The result is depicted in Figure 3. We thus have an effectively computable approximate upper bound on the precision-density curve, which we can use as a baseline for comparison with efficient heuristics.

Due to the high polynomial time complexity involved in solving the optimal scheme (Proposition 3.6 and complexity of linear programming), we now provide efficient estimators whose performance is close to the optimal curve. They stand as efficiently computable lower bounds on the optimal precision for particular densities, which empirically show the tightness of our upper bound.

3.4 Estimators

We propose three estimators here. A detailed analysis of the estimators is given in Section 5. Figure 3 shows a comparison of the following estimators with the optimal estimator. All these estimators are based on the Peeling procedure. *The time complexity of the estimators is based only on the formation of DAG, 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.

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.

Peeling: A linear binning estimator via peeling. Another natural estimator arises from the bins given by the Peeling procedure. 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 (recall the definition of a binning, given near the beginning of Section 2). This forms a *linear binning*, which 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 $\mathcal{B}_1, \ldots, \mathcal{B}_j$, 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. With a slight abuse of notaion, when the term Peeling is used as an estimator, we mean the above linear binning estimator.

Peeling+: Peeling with deduction of same bin pairs. This estimator first runs the Peeling algorithm, and classifies the vertices into bins. For each node except those in bin $\mathcal{B}_1$, we find the averaged value of its neighbors' bin numbers (levels), which we call in what follows the node's *average neighbor level*. A high value of average neighbor level indicates youngness 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. Moreover, its performance is very close to the optimal curve at around $\varepsilon = 1$. (The density may not be exactly 1 because there may still exist some pairs inside bins that have the same average neighbor level and thus makes them incomparable. Moreover we cannot order the nodes inside bin 0).

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. We will primarily focus on Peeling estimator.

4 EXPERIMENTS

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

4.1 Synthetic graphs

We derive significant insight by studying the Peeling method empirically. Table 1 and Figure 4 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

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 $\mathcal{PA}(n = 5000, m)$

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.

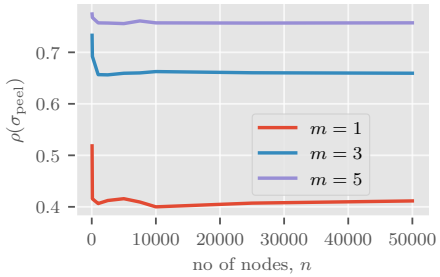


Figure 4: Recall of the Peeling algorithm remains constant when the number of nodes varies

Robustness of the Peeling algorithm. Table 2 demonstrates robustness of our Peeling algorithm for various generalizations of the model: preferential attachment model with variable m (denoted by M and $\sim$ unif$\{a, b\}$ denote discrete uniform distribution), uniform attachment model (denoted by $\mathcal{UA}$), and the very general Cooper-Frieze model [8, 9]. In our instance of Cooper-Frieze model, at each time instant, number of new edges at a time step m is drawn from $\sim$ unif$\{5, 50\}$, the model allows either addition of a new node (with probability 0.75) or addition of edges between exiting nodes, and the endpoints of new edges can be selected either preferentially (with probability 0.5) or uniformly among existing nodes.

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

Table 2: A general comparison: $n = 5000$

These results suggest that the proposed DAG-based methods can simultaneously achieve high precision and recall/density.

4.2 Real-world networks

Datasets. We use the following datasets. All the datasets except human brain network are publicly available. We explain later how the brain network is formed. The ArXiv, Wikipedia and DBLP are collected from [7], and human protein network is available from BioGRID (https://thebiogrid.org). The first three networks have

Dataset	# Nodes	# Edges	Genre
ArXiv High Energy Physics	7,464	116,268	Citation
Simple English Wikipedia	100,312	1,627,472	Hyperlink
DBLP CS bibliography	1,137,114	5,018,065	Coauthorship
Human protein interaction	14,867	126,593	Biological
Human brain	56	1,164	Biological

Table 3: Network statistics

partial temporal information available, and he original ranking of the network is deduced from the time stamp of the edges. For the protein network, only a prediction is presented in previous works, and for the brain network no explicit temporal information is available.

The ArXiv network: Directed network; the nodes are the publications and the edges are formed when a publication cite 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; an edge between two authors represents a common publication.

Human protein interaction network: Directed network; the nodes are the proteins, and the links indicate interaction between two proteins in gene formation. The original ranking is taken as the approximate ranking provided by ProteinHistorian [6], and has only 16 bins. Note that the ranking provided in [6] is only an approximation, and not the ground truth.

Human brain network: It is an undirected network. The nodes here are the regions in human brain, and the edges represent communication between two regions while performing an activity. The network is formed as follows. We gathered the human brain fMRI data at resting state from the Cambridge-Buckner dataset [3, 11]. 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 the between member voxels.

Results. The results for all the datasets except human brain network are presented in Table 4. For all real networks above, with the exception of the protein interaction network, our algorithms yield excellent precision and recall results, consistent with our theoretical predictions. Note that these results would have improved if we have complete temporal information instead of partial original ranking with bins. The above table shows the use of Peeling+ algorithm. 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).

Human protein interaction network: In the case of protein interaction networks, our definitions of precision and recall are overly pessimistic for a few reasons, the first of which is that the "ground truth" that we use is a very sparse binning. A pair whose order is

	$\theta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel}})$	$\delta(\sigma_{\text{peel}})$	$\rho(\sigma_{\text{peel+}})$
ArXiv	0.708	0.681	0.961	0.707
Wikipedia	0.624	0.548	0.878	0.609
DBLP	0.785	0.728	0.927	0.764
Protein interaction	0.5526	0.511	0.925	0.54

Table 4: Results for real-world networks: $\theta(\sigma_{\text{peel+}}) \approx \rho(\sigma_{\text{peel+}})$ and $\delta(\sigma_{\text{peel+}}) \approx 1$.

inferred by our algorithm and not inferred in this ground truth is counted as an error. Thus, we extend the ground truth ordering to a total order by randomly guessing the order of incomparable pairs, resulting in the numbers given. Moreover, importantly, the protein interaction network is much noisier than the others that we analyzed: while we have reasonable confidence that all edges are accounted for in the citation and Wikipedia networks, there is no such guarantee that this holds for the protein interaction network, as the presence or absence of each edge is costly to determine.

Human brain network: The purpose here is to predict the evolutionary order among the important regions inside the brain. Figure 5a presents the bins of regions we found via Peeling algorithm. Figure 5b shows an image of the human brain with prominent regions annotated with bin number. We note here that there is no available ground truth of the ranking, and we propose our order for further study in this area. Our ranking can be backed up with the results we derived in this paper and with the prior studies that show preferential attachment involvement in brain networks [21]. Moreover a quick consistency check can be done on some of the brain regions. E.g., the corpus callosum which joins the two hemispheres of the brain is supposed to develop 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 proper evolutionary order of brain regions.

Finally, in general, the more preferential attachment plays a role in the formation of a real network, the closer the performance will be to our theoretical and synthetic results. In some of the networks that we analyzed, previous studies report preferential attachment involvement, for instance see [14]. Vértes et al. proposed in [21] a widely used variation of preferential attachment model for brain functional networks.

5 ANALYSIS OF ALGORITHMS, PROOFS

Here we derive certain properties of the estimators and provide proof sketches for the results in the previous sections.

5.1 Maximum-density precision 1 estimator

In this subsection, we show that the point (density, precision) = $(0, 1)$ is a point on the optimal precision curve: namely, we prove perfect-precision estimator estimator has precision one, but asymptotically negligible density, and this estimator has the maximum density among all precision one estimators. From Theorem 3.5, 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 5.1 (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 (sketch). **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 [15] 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: We only sketch the lower bound proof. In a nutshell, for $m = 1$, we can show that X_t is the *total path length* of the tree. We then use the results of [18] to finish the proof. For general m, we can reduce to the $m = 1$ case using the correspondence between $\mathcal{PA}(n, m)$ and $\mathcal{PA}(nm, 1)$. □

The above theorem implies that the density of the perfect pair estimator is asymptotically 0, for arbitrary m.

A 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 $1 > \delta(m) > 0$. For lack of space, we omit the explanation of the conjecture.

Theorem 5.1 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.

5.2 Linear binning estimator Peeling estimator

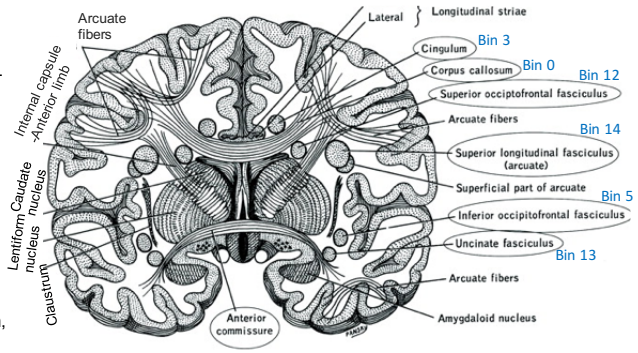
We can start by showing a crude estimate of the precision and density of the Peeling estimator.

Theorem 5.2 (Non-negligible precision and density of the peeling estimator). *For each $m \geq 1$ the Peeling estimator has precision and density $\Theta(1)$.*

Proof (sketch). The precision claim follows by showing that, with high probability, there exist $\Theta(n)$ vertices in the last $\Theta(n)$

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

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



(b) Prominent regions annotated with bin number[4]

Figure 5: PEELING bins of regions in human brain network

timesteps that are never chosen (so that they are removed in the first layer and, thus, correctly declared to be younger than $\Theta(n)$ other vertices). This also proves the density claim. □

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. [18]. 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.

5.3 Proof sketch of Proposition 2.1

First, by definition of $\Gamma(H)$ (see Section 3.1 for definitions), we must have that $C_{ML}(H) \subseteq \Gamma(H)$. We will show, in fact, that the likelihoods given to all elements of $\Gamma(H)$ are equal. This will then imply that $C_{ML}(H) = \Gamma(H)$. Next, from a result of [15], 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 $\Pr[G = G_1] = \Pr[G = G_2]$. We omit the proof of this fact for the sake of brevity.

5.4 Proof of Lemma 3.1

We can express the conditional probability in question as a sum, as follows: $\Pr[\pi^{-1}(v) < \pi^{-1}(w) | \pi(G) = H]$
$= \sum_{\sigma\,:\,\sigma^{-1} \in \Gamma(H), \sigma^{-1}(v) < \sigma^{-1}(w)} \Pr[\pi = \sigma | \pi(G) = H]$.

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 $\Pr[\pi = \sigma | \pi(G) = H] = \frac{\Pr[G = \sigma^{-1}(H) | \pi(G) = H]}{|\mathrm{Aut}(H)|} = \frac{1}{|\mathrm{Aut}(H)||\mathrm{Adm}(H)|}$. 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 $|\mathrm{Adm}(H)| = |\Gamma(H)|/|\mathrm{Aut}(H)|$ [16]. □

5.5 Proof of Lemma 3.3

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. □

6 CONCLUSIONS

We considered approaches to the problem of estimating the temporal ordering of nodes in a dynamically grown network, based on modeling by the preferential attachment distribution. To overcome the limitations imposed by the structure of such models, we dealt with partial ordering schemes. The proposed algorithms showed good performance both in analysis and simulations, that is close to the optimal estimator.

We expect that this work will open up many interesting questions on the inference of node ages in growing networks.

ACKNOWLEDGMENTS

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 for providing network correlation matrix from the brain data.

[4]The original figure without bin number is taken from [12].

REFERENCES

[1] Jacqueline Banks, Scott Garrabrant, Mark L Huber, and Anne Perizzolo. 2010. Using TPA to count linear extensions. *arXiv preprint arXiv:1010.4981* (2010).

[2] Albert-László Barabási and Réka Albert. 1999. Emergence of Scaling in Random Networks. *Science* 286, 5439 (1999), 509–512.

[3] Bharat B Biswal, Maarten Mennes, Xi-Nian Zuo, Suril Gohel, Clare Kelly, Steve M Smith, Christian F Beckmann, Jonathan S Adelstein, Randy L Buckner, Stan Colcombe, et al. 2010. Toward discovery science of human brain function. *Proceedings of the National Academy of Sciences* 107, 10 (2010), 4734–4739.

[4] Graham Brightwell and Peter Winkler. 1991. Counting Linear Extensions is #P-complete. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing (STOC)*. ACM, New York, NY, USA, 175–181.

[5] Sébastien Bubeck, Luc Devroye, and Gábor Lugosi. 2016. Finding Adam in random growing trees. *Random Structures & Algorithms* (2016).

[6] John A. Capra, Alexander G. Williams, and Katherine S. Pollard. 2012. ProteinHistorian: Tools for the Comparative Analysis of Eukaryote Protein Origin. *PLOS Computational Biology* 8 (06 2012), 1–9.

[7] KONECT The Koblenz Network Collection. 2016. Preferential attachment exponent. Available on http://konect.uni-koblenz.de/statistics/prefatt [Accessed: 30 April 2017]. (2016).

[8] Colin Cooper. 2006. Distribution of vertex degree in web-graphs. *Combinatorics, Probability and Computing* 15, 05 (2006), 637–661.

[9] Colin Cooper and Alan Frieze. 2003. A general model of web graphs. *Random Structures & Algorithms* 22, 3 (2003), 311–335.

[10] Alan Frieze, Wesley Pegden, et al. 2017. Looking for vertex number one. *The Annals of Applied Probability* 27, 1 (2017), 582–630.

[11] International Neuroimaging Data-Sharing Initiative. 2017. The 1000 Functional Connectomes Project. http://fcon_1000.projects.nitrc.org/index.html. (2017). [Online; accessed 11-February-2018].

[12] Brian J Jellison, Aaron S Field, Joshua Medow, Mariana Lazar, M Shariar Salamat, and Andrew L Alexander. 2004. Diffusion tensor imaging of cerebral white matter: a pictorial review of physics, fiber tract anatomy, and tumor imaging patterns. *American Journal of Neuroradiology* 25, 3 (2004), 356–369.

[13] Alexander Karzanov and Leonid Khachiyan. 1991. On the conductance of order Markov chains. *Order* 8, 1 (1991), 7–15.

[14] Jérôme Kunegis, Marcel Blattner, and Christine Moser. 2013. Preferential attachment in online networks: Measurement and explanations. In *Proceedings of the 5th Annual ACM Web Science Conference*. ACM, 205–214.

[15] Tomasz Łuczak, Abram Magner, and Wojciech Szpankowski. 2017. Asymmetry and structural information in preferential attachment graphs. *arXiv preprint arXiv:1607.04102* (2017).

[16] Abram Magner, Ananth Grama, Jithin Sreedharan, and Wojciech Szpankowski. 2017. Recovery of Vertex Orderings in Dynamic Graphs. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*.

[17] S. M. Vijay Mahantesh, Sudarshan Iyengar, M. Vijesh, Shruthi R. Nayak, and Nikitha Shenoy. 2012. 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)*. 517–521.

[18] Dimbinaina Ralaivaosaona and Stephan Wagner. 2016. 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*. 1–12.

[19] Devavrat Shah and Tauhid Zaman. 2011. Rumors in a network: Who's the culprit? *IEEE Transactions on information theory* 57, 8 (2011), 5163–5181.

[20] Mansi Srivastava, Oleg Simakov, Jarrod Chapman, Bryony Fahey, Marie EA Gauthier, Therese Mitros, Gemma S Richards, Cecilia Conaco, Michael Dacre, Uffe Hellsten, et al. 2010. The Amphimedon queenslandica genome and the evolution of animal complexity. *Nature* 466, 7307 (2010), 720–726.

[21] Petra E Vértes, Aaron F Alexander-Bloch, Nitin Gogtay, Jay N Giedd, Judith L Rapoport, and Edward T Bullmore. 2012. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences* 109, 15 (2012), 5868–5873.

[22] Kai Zhu and Lei Ying. 2016. Information source detection in the SIR model: A sample-path-based approach. *IEEE/ACM Transactions on Networking* 24, 1 (2016), 408–421.