Recovering the first vertices in a Preferential Attachment model

Math 199H - Honors Thesis
Department of Mathematics,
University of California, San Diego

Spring 2020

Thu Nguyen
Advisor: Professor Ery Arias-Castro

May 5, 2020

Abstract

We study the problem of recovering the first vertices in a graph grown under the *preferential attachment* rules. We focus on tree graphs. We introduce the familiar algorithm of looking at the largest connected component after removing any one vertex.

We first consider recovering only vertex 1. We provide a detailed proof of the accuracy of the algorithm. We then test the algorithm with simulations over a wide range of graph sizes. Given the computation cost, we propose a two-step procedure which combine that with using the vertex degree.

We also consider recovering more than just vertex 1. We propose a sequential algorithm where at each stage we do a local search conditioned on what already know. We conjecture on the expectation of that algorithm and then test the algorithm on simulations.

We find that we can get significantly accurate when recovering vertex 1. However, the problem starts becoming much harder once we want to recover more vertices. In particular, once we look for the fifth vertex and beyond, the accuracy drops considerably. Nonetheless, it appears that we can increase the accuracy if we are willing to trade that with being precise.

Acknowledgement

I would like to express the most sincere thanks to Professor Arias-Castro, who offered me the opportunity to work on this project, provided resources, and guided me through how to tackle such open problems. It has been an eye-opening experience for me.

Contents

1 Introduction

Preferential attachment model was first introduced in the 1970s by Price in [9] and has picked up in popularity since the 1990s after the paper by Barabási and Albert in [1]. The model has been shown to represent many of the networks observed in real-life, ranging from biology and transportation networks, and the classic example of citation networks.

One reason for why *preferential attachment* model is commonly observed relies on the way *preferential attachment* model grows, which mimics the phenomena of "the rich get richer": when graphs evolve over time, newly added vertices will have tendency to connect to already well-connected vertices.

We introduce a (somewhat) standard version of a *preferential attachment* mode. This takes three parameters, namely:

1. $G^{(0)}$: the seed graph at time $t = 0$
2. m : the number of new edges added at each iteration;
3. α : the parameter which controls the growth of the graph.

We can now define the graph iteratively as follows. Let $G^{(0)}$ be given. Let $d_t(u)$ be the degree of u in $G^{(m,\alpha)}(t)$. We construct $G^{(m,\alpha)}(t+1)$ as:

1: Add a new vertex $t+1$
2: Choose m existing vertices $u \in V(T^{(\alpha)}(t))$ such that

$$\mathbb{P}(u \text{ is chosen}) \ \propto \ \big(d_t(u)\big)^{\alpha}$$

3: Add new edges between $t+1$ and u

The specification above leaves us a lot of freedom to tweak the model. Some choices we can make include:

1. m : the larger the m, the denser the graph, but also more variation between different realizations of such a model;
2. α : the larger the α, the faster the already well-connected vertices will be even more connected, the behavior of which closely mimics the observed real-life phenomena of *the rich get richer*; common choices of α are 0 and 1;
3. if $m > 1$, we can either choose m vertices sequentially or simultaneously, the former of which further enhances the *the rich get richer* phenomena;

4. we can introduce another parameter, say $p \in [0, 1]$: a *Bernoulli* random variable which decides if at the current iteration we want to add a new vertex and then new edges or we can just add new edges between the vertices.

A defining characteristic of the *preferential attachment* model is its degree profile, which closely follows a *power-law*. This characteristic has been studied rigorously and in great detail, such as by Dorogovtsev et al in [5] and by Hofstad in [7]. In particular, given a strict *preferential attachment* model (with $m = 1$ and $\alpha = 1$), it has been shown, for example by Chung in [4], that in the limit as k gets large,

$$\mathbb{P}(X = k) \ \propto \ k^{-3}$$

where $X = k$ is the number of vertices of degree k. We observe that the older a vertex, the larger the expected degree of that vertex.

2 Overview

The motivation for our study is to recover the seed graph of a graph assumed to have grown under a *preferential attachment* mechanism.

Suppose we have a graph G, which we believe grew according to our *preferential attachment* model specified in 3.1.1. G gives us a snapshot of our graph at some point in time. It is natural to ask ourselves if we could trace back in time and recover graphs at previous past. Of particular interest is the seed graph which gave rise to G. In other words, we believe that G started with a single vertex, say 1, we now want to identify that vertex.

We first consider the special case of recovering vertex 1, and then generalize that to recover first L vertices. To introduce a bit of structure to the graphs, we work with *preferential attachment* model on tree, which will be defined later on. Section 3 addresses the problem of recovering vertex 1, and section 4 that of L vertices.

Within each problem, we first define the set-up and the objectives, together with other considerations, such as the computation cost or the need to be accurate versus concise. We next introduce or propose algorithms to tackle the problems. For each algorithm, we introduce the motivation and the theoretical results or supporting arguments. We then test the algorithms over a wide range of graph sizes and note the results.

Furthermore, to keep track of the performance on vertex recovery, we introduce two performance metrics:

$$\begin{aligned} Accuracy &= \frac{\text{number of times all } L \text{ vertices are recovered}}{\text{number of simulations}} \\ Precision &= \frac{L}{K} \end{aligned}$$

Ideally, we look for an algorithm that could be both highly accurate and highly precise. It should be noted that as introduced by Lugosi and Pereira in [8], in the problem of recovering the first L veritces, there are two notions of *accuracy* of an algorithm: suppose the algorithm returns a set of vertices, say H, then:

- *acuracy of first kind:* $H \subseteq L$
- *acuracy of second kind:* $L \subseteq H$

In this study, we are going to look only at the accuracy of *second kind*, ie. we want to fully recover the first L vertices.

3 Recovering Vertex 1

3.1 Set-up and Objective

We first define some notions on graphs.

Definition 3.1. ***Simple graph:*** *A graph G is simple if there is at most one edge between any pair of vertices. We write $G = (V, E)$ to denote that V and E are the sets of vertices and edges in G respectively.*

Definition 3.2. ***Connected graph:*** *A simple graph G is connected if for any pair of vertices u and v in G, there always exists a path between u and v.*

Definition 3.3. ***Tree:*** *A simple graph G is a tree if it does not contain a cycle: if u and v are two vertices in a connected tree, there is exactly one path between between them. We denote a tree rooted at a vertex u by (G, u).*

Definition 3.4. ***Vertex neighbor:*** *Let $u \in V(G)$, a vertex v is said to be a neighbor of u, denoted by $v \in N(u)$, if $\{u, v\} \in E(G)$, ie. there is an edge between u and v.*

Definition 3.5. ***Vertex degree:*** *Let $u \in V(G)$ be a vertex in a simple graph G, the degree of u, denoted by $d_G(u)$ is the number vertices neighboring u.*

Notation 3.1. *Let G be a fully labeled graph, then G^o denote the isomorphism class of G, ie. G^o is an unlabeled copy of G.*

Notation 3.2. *Let $L \in \mathbb{N}$, we denote $[L] = \{1, 2, \ldots, L\}$.*

3.1.1 Set-up

We will start by specifying the exact settings and the objective. We consider a special case of a *preferential attachment* model introduced earlier, one in which $m = 1$ and $\alpha \in \{0, 1\}$. These choices of parameters give a *uniform attachment* and *preferential attachment* on trees respectively.

Let $T^{(0)}$ be a singleton (a single vertex with no edges). Let $T^{(\alpha)}(t)$ be the (tree) graph at time t. Let $d_t(u)$ be the degree of u in $T^{(\alpha)}(t)$. We construct $T^{(\alpha)}(t+1)$ as:

1: Add a new vertex $t + 1$

2: Choose an existing vertex $u \in V(T^{(\alpha)}(t))$ such that

$$\mathbb{P}(u \text{ is chosen}) \ \propto \ (d_t(u))^\alpha$$

3: Add a new edge between $t + 1$ and u

In particular,

(a) when $\alpha = 0$, we have a *uniform attachment* model:

$$\mathbb{P}(u \text{ is chosen}) \;=\; \frac{1}{|V\big(T^{(0)}(t)\big)|} \;=\; \frac{1}{t}$$

(b) when $\alpha = 1$, we have a *preferential attachment* model:

$$\mathbb{P}(u \text{ is chosen}) \;=\; \frac{d_t(u)}{\displaystyle\sum_{v \in V(T^{(1)}(t))} d_t(v)} \;=\; \frac{d_t(u)}{2(t-1)}$$

Figure 1 illustrates an instance of the revolution of a *preferential attachment* model on tree.

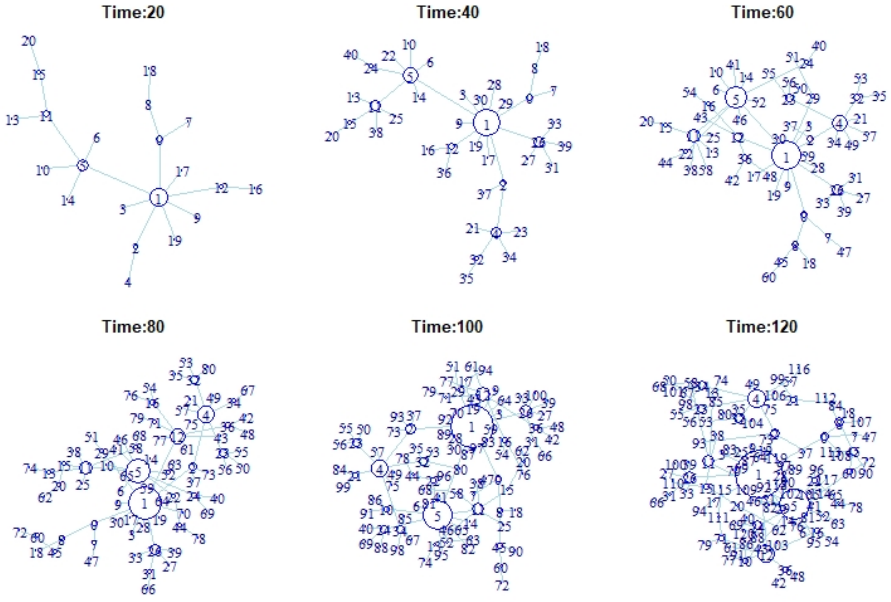

Figure 1: Evolution of a *preferential attachment* model on tree.

3.1.2 Objectives

Suppose we have a graph G, which we believe grew according to our *preferential attachment* model specified in 3.1.1. G gives us a snapshot of our graph at some point in time. It is natural to ask ourselves if we could trace back in time and recover graphs at previous past. Of particular interest is the seed graph which gave rise to G. In other words, we believe that G started with a single vertex, say 1, we now want to identify that vertex.

Not only do we want to identify vertex 1, we want to be as *accurate* and *precise* as possible, which were defined in section 2. We now establish certain criteria that we would like the algorithms to achieve. The first is on the accuracy:

> Fix some $\varepsilon > 0$. We want an algorithm that takes in an unlabeled graph G and return K vertices such that:
>
> $$\mathbb{P}(1 \in K) \ \geq \ 1 - \varepsilon$$

The second is on the precision:

> In returning K vertices, for a graph G of n vertices, K should not depend on n. Instead, K should depend on ε only.

In particular, as n goes to infinity, we want the algorithm to return only finitely many vertices without compromising on the accuracy.

We can now formalize the problem. Given a labelled graph G, let G^0 be the isomorphism class of G, i.e. G^0 is an unlabelled copy of G. Let $\varepsilon > 0$, we look for a map H which takes G^o as input and return a subset of vertices of G^o such that:

$$\mathbb{P}\Big(1 \in H\big(\big(G^{(\alpha)}(n)\big)^o\big)\Big) \ \geq \ 1 - \varepsilon$$

3.2 Algorithm

Arguably one of the most defining characteristic of a *preferential attachment* model is its *power-law* distribution of the vertex degrees, as seen in Barabási and Albert in [1]. In particular, given a strict *preferential attachment* model (with $m = 1$ and $\alpha = 1$), it has been shown, for example by Bollobás in [2] and by Chung in [4], that in the limit as k gets large,

$$\mathbb{P}(X = k) \ \propto \ k^{-3}$$

where $X = k$ is the number of vertices of degree k. We observe that the older a vertex, the larger the expected degree of that vertex.

Given the observations, it is natural to think of using the vertex degree as the criteria in returning those K vertices. However, it is claimed by Bubeck et al in [3] that to guarantee the precision, we would need to return $\mathcal{O}(\ln(n))$ vertices, where n is the number of vertices in the graph G. In particular, when n goes to infinity, we would need to return infinitely many vertices, which does not satisfy the conciseness criteria.

Also in [3], Bubeck et al introduced a different algorithm. Instead of looking at the degree of a vertex u, we look at the size of the maximum subgraph starting at a neighbor of u and not containing u.

We first introduce a notation.

Notation 3.3. *$(T, u)_{v\downarrow}$ is the subtree rooted at v, a child of u, and not containing u.*

Figures 2 and 3 illustrates examples of those notations.

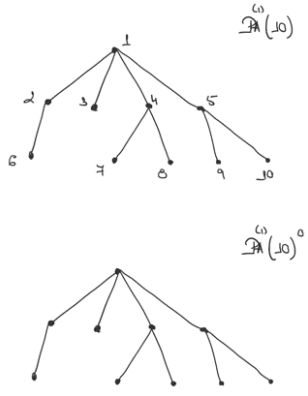

Figure 2: Unlabeled copy G^0 of G.

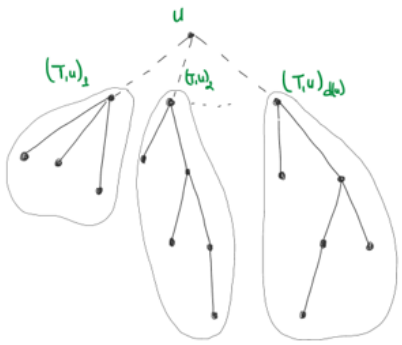

Figure 3: $(T, u)_{v\downarrow}$

We are now ready to describe the algorithm.

Algorithm 1 Recovering vertex 1

Input: an unlabeled tree graph $T^o(n)$
for each $u \in V(T^o(n))$ **do**
  for each $v \in N(u)$, a neighbor of u **do**
    Compute $|(T,u)_{v\downarrow}|$
  end for
  Return $\varphi(u) = \max_{v\in N(u)} \left|(T,u)_{v\downarrow}\right|$
end for
Output: K vertices with the smallest φ values

We have results on the asymptotic accuracy of the algorithm on the *preferential attachment* and *uniform attachment* models on tree.

Theorem 3.1: Uniform attachment model on tree, [3]

Consider a *uniform attachment* model on tree with n vertices, denoted by $UA(n)$. Let $K \geq 2.5\frac{\log(1/\varepsilon)}{\varepsilon}$, then

$$\liminf_{n\to+\infty} \Big(1 \in H_\varphi\big(UA(n)^0\big)\Big) \;\geq\; 1 - \frac{4\varepsilon}{1-\varepsilon}$$

We first give a sketch of the proof here.

1. Let $T_{i,k}$ $(i \leq k)$ be the tree containing vertex i by removing from $UA(n)$ all edges between vertices $\{1,2,\ldots,k\}$. Show that:
$$\left(\frac{|T_{1,k}|}{n}, \frac{|T_{2,k}|}{n}, \ldots, \frac{|T_{k,k}|}{n}\right) \xrightarrow[n\to\infty]{\mathbb{P}} Dir(1,1,\ldots,1)$$
2. Show that: $\mathbb{P}(1 \notin H_\varphi) \;\leq\; \mathbb{P}\Big(\varphi(1) \geq (1-\epsilon)n\Big) \;+\; \mathbb{P}\Big(\exists i > K : \varphi(i) \leq (1-\epsilon)n\Big)$
3. Show that: $\limsup_{n\to+\infty} \mathbb{P}\Big(\varphi(1) \geq (1-\epsilon)n\Big) \leq 2\epsilon$
4. Show that: $\limsup_{n\to+\infty} \mathbb{P}\Big(\exists i > K : \varphi(i) \leq (1-\epsilon)n\Big) \;\leq\; K(1-\epsilon)^{K-1}$
5. Show that: if $K \geq 2.5\,\frac{\log(1/\epsilon)}{\epsilon}$, then the result follows.

Remark: contrary to the procedure described in the algorithm above, where we were working with an unlabelled graph, here, to analyze the accuracy, we work with a chronically labelled graph, i.e. we know the ground truth: the exact order of which vertices appeared.

Proof: We now give a complete proof.

1. Let $T_{i,k}$ $(i \leq k)$ be the tree containing vertex i by removing from $T(n)$ all edges between vertices $\{1, 2, \ldots, k\}$. Given the labelled graph (in particular, a tree) $T(n)$, in which labels are chronical, consider the first k elements. Note that if we remove all edges between them, this subgraph now becomes a collection of singletons, i.e. k vertices and 0 edges.

   Now, for every new vertex t, since t is equally likely connected to each existing vertices, from the perspective of growing subtrees $T_{i,k}$, the probability of t getting added to a particular subtree $T_{i,k}$ is proportional to the current size of $T_{i,k}$. Hence, growing $T_{i,k}$ is equivalent to the classic Polya Urn Model with k bins starting with 1 ball in all bins.

   Furthermore, since we started with k singletons, the initial sizes of each bin are all 1. Note that $\frac{|T_{i,k}|}{n}$ is the proportion of the size of bin i relative to all k bins. Hence, as n goes to infinity, we have:

$$\left(\frac{|T_{1,k}|}{n},\ \frac{|T_{2,k}|}{n}, \ldots, \frac{|T_{k,k}|}{n}\right) \xrightarrow[n\to\infty]{\mathbb{P}} Dir(1, 1, \ldots, 1)$$

2. We first note that the event of vertex 1 not recovered implies that there must be at least one vertex $i > K$ such that $\varphi(i) \leq \varphi(1)$, which gives:

$$\begin{aligned} 1 \notin H_\varphi \ &\subseteq\ \exists i > K : \varphi(i) \leq \varphi(1) \\ \implies \mathbb{P}(1 \notin H_\varphi) \ &\leq\ \mathbb{P}(\exists i > K : \varphi(i) \leq \varphi(1)) \\ \implies \mathbb{P}(1 \notin H_\varphi) \ &\leq\ \mathbb{P}(\varphi(1) \geq (1-\varepsilon)n) + \mathbb{P}(\exists i > K : \varphi(i) \leq (1-\varepsilon)n) \end{aligned}$$

3. By construction, $\varphi(1)$ is the size of the largest subtree starting at a child of 1.

   Now, recall the definition from step 1, $T_{1,2}$ and $T_{2,2}$ are the two subtrees each starting at 1 and 2 after removing the edge $\{1, 2\}$. Since $T_{1,2}$ always includes vertex 1, when computing any subtrees starting from children of 1 given $T_{1,2}$, we always have $\varphi(1)' < T_{1,2}$. Hence:

$$\varphi(1) \ \leq\ \max\left\{|T_{1,2}|,\ |T_{2,2}|\right\}$$

   Note that the equality sign happens when 1 is a leaf.

   Furthermore, note that after time $t = 2$, in which case we have 2 vertices, it follows that vertex 3 has equal and identical chances of connecting to either vertex. By symmetry:

$$i \in \{1, 2\} : \quad \frac{|T_{i,2}|}{n} \xrightarrow[n\to\infty]{\mathbb{P}} Unif[0, 1]$$

In particular, they are identically distributed. This gives:

$$\limsup_{n\to+\infty} \mathbb{P}\Big(\varphi(1) \geq (1-\epsilon)n\Big) \;\leq\; \limsup_{n\to+\infty} \mathbb{P}\Big(\frac{|T_{i,2}|}{n} \geq (1-\epsilon) \;\Big|\; i \in \{1,2\}\Big) \;=\; 2\epsilon$$

4. Let $i > K$. Let j be a vertex between i and the K vertices (we note that j can be one of the K vertices). We have:

$$\varphi(i) \;\geq\; |(T,i)_{j\downarrow}| \;\geq\; \min_{1\leq k\leq K} \sum_{j=1, j\neq k}^{K} |T_{j,K}|$$

by removing the subtree containing vertex i itself.

Recall that the vector $\frac{1}{n}(|T_{1,K}|, \ldots, |T_{K,K}|)$ converges in distribution to a Dirichlet distribution with parameters $(1, \ldots, 1)$, it follows that:

$$\frac{1}{n} \sum_{j=1, j\neq k}^{K} |T_{j,K}| \;\xrightarrow[n\to\infty]{\mathbb{P}}\; Beta(K-1, 1) \tag{1}$$

This gives:

$$\begin{aligned}
&\limsup_{n\to+\infty} \mathbb{P}\Big(\exists i > K : \varphi(i) \leq (1-\epsilon)n\Big) \\
&\leq \limsup_{n\to+\infty} \mathbb{P}\Big(\exists k \in [K] : \min_{1\leq k\leq K} \sum_{j=1, j\neq k}^{K} |T_{j,K}| \leq (1-\epsilon)n\Big) \\
&\leq K(1-\epsilon)^{K-1}
\end{aligned}$$

5. Let $K \geq 2.5\frac{\log(1/\varepsilon)}{\varepsilon}$. From steps $2, 3, 4$ above, we have:

$$\begin{aligned}
\mathbb{P}(1 \notin H_\varphi) \;&\leq\; \mathbb{P}\Big(\varphi(1) \geq (1-\epsilon)n\Big) \;+\; \mathbb{P}\Big(\exists i > K : \varphi(i) \leq (1-\epsilon)n\Big) \\
&\leq\; 2\epsilon + K(1-\epsilon)^{K-1}
\end{aligned}$$

and the result follows. ■

Theorem 3.2: Preferential attachment model on tree, [3]

Consider a *preferential attachment* model on tree with n vertices, denoted by $PA(n)$. Let $K \geq C\frac{\log^2(1/\varepsilon)}{\varepsilon^4}$ for some constant $C > 0$, then

$$\liminf_{n\to+\infty} \Big(1 \in H_\varphi\big(PA(n)^0\big)\Big) \;\geq\; 1-\varepsilon$$

3.3 Simulations

We would like to study the algorithm via simulations. In particular, the focus is on how fast the accuracy converges, if any. As a motivation, table 1 shows the number of vertices we need to return to achieve certain levels of accuracy.

Desired accuracy	UA on tree	PA on tree
90%	≥ 381	$\geq C(53,019)$
95%	≥ 890	$\geq C(1,435,906)$

Table 1: The number of vertices to be returned in order to achieve the desired accuracy.

A quick look gives an impression that to ensure the accuracy, we would need to output a very large number of vertices. We will test the algorithms on a variety of graph sizes.

3.3.1 Simulation Data

We will build graphs of sizes over the range of:

$$n \in \{5000k : \ k \in [15]\}$$

(from 5000 to 75,000, in increments of 5000). In each graph size, we will build 1000 simulations. We will then run the algorithms over those 1000 simulations, and calculate the number of times 1 is recovered. We will do that over different choices of K:

$$K \in [10] \cup \{5k : \ k \in [20]\}$$

We will repeat the same procedure for two sets of graphs: one generated by a *uniform attachment* model on tree, and one by a *preferential attachment* model on tree.

Disclaimer: Due to the significant cost of implementing algorithm 1, we will only compute the φ value of the first 5000 vertices in all the simulations. All the other vertices will be assigned $n-1$ by default where n is the size of the graph (i.e. they are assumed to be leaves).

3.3.2 Simulation Results

We first note that in the analysis, we use the terms *accuracy* and *recovery rate* interchangably. Now we look ***preferential attachment*** model on tree.

Figure 4 shows the recovery rate across different value for K on *preferential attachment* model on trees of size $50,000$, via the method of minimum maximum subgraph. We have some observations:

1. The accuracy converges very quickly.
2. We only need to return 20 vertices to achieve an accuracy of at least 90%.
3. If we only take the vertex with the minimum φ value, we would be correct more than one third of the time, which increases to over two thirds with only three such vertices.

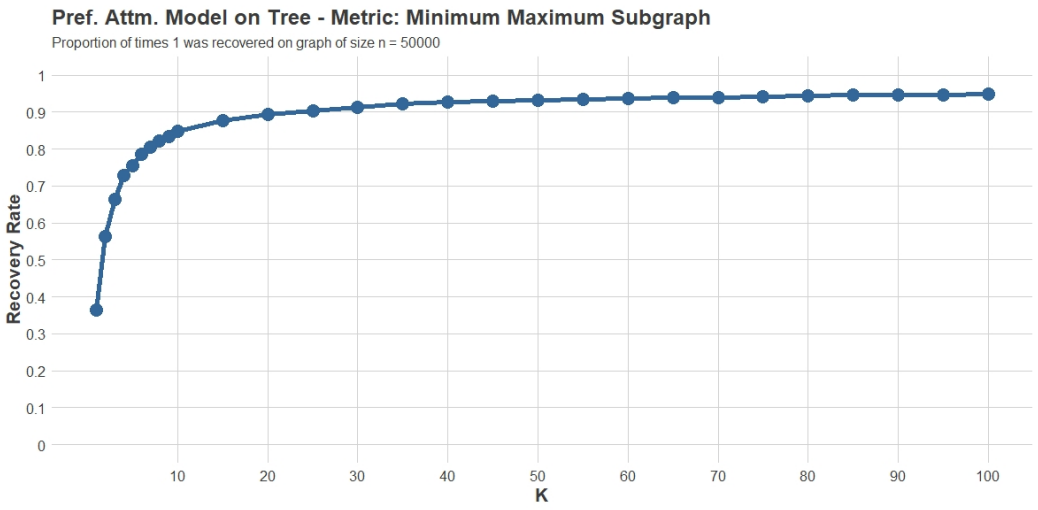

Figure 4: Recovery rates of vertex 1 on *preferential attachment* model on tree across different K on graphs of size $50,000$.

Figure 5 shows the long-run behavior of the recovery rate when the graph size increases while returning only 20 vertices. We have some observations:

1. Except from the rugged segment between sizes of $5,000$ to $15,000$, the recovery rate changed minimally even after the size increased by over 10 times.

2. The accuracy consistently stayed above 90% regardless of the graph sizes.

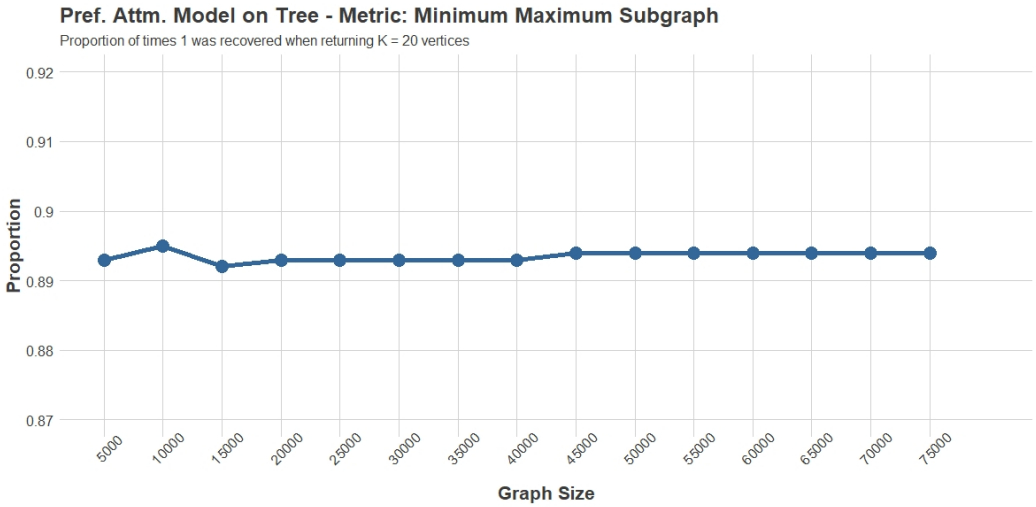

Figure 5: Recovery rates of vertex 1 on *preferential attachment* model on tree across different graph sizes when returning $K = 20$ vertices.

Figures 4 and 5 give us an impression that the algorithm is very robust: it can be precise and concise very quickly. Figures 6 and 7 show the recovery rates when we only return only vertices with the smallest φ values. One observation significantly stands out:

1. The recovery rates remained unchanged across a wide range of graph sizes.

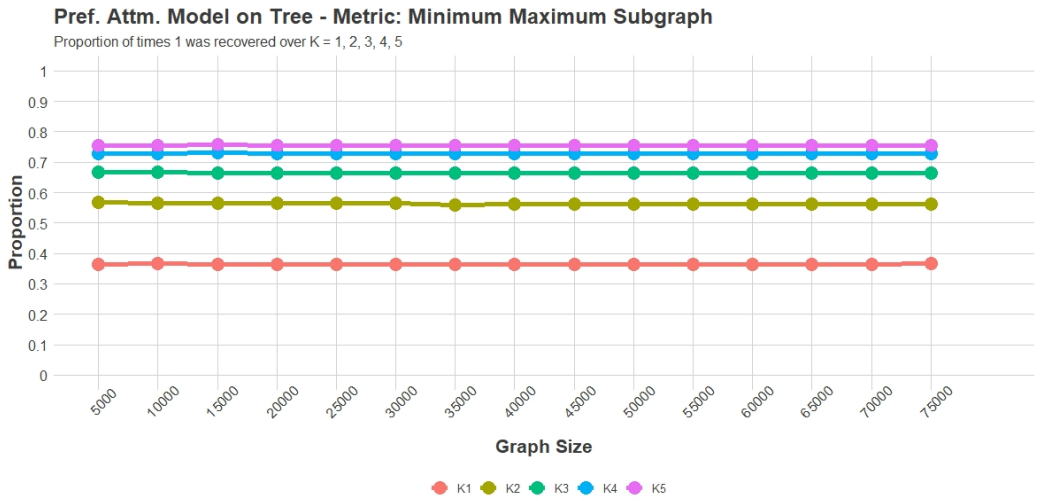

Figure 6: Recovery rates of vertex 1 on *preferential attachment* model on tree across different graph sizes when returning only $1, 2, 3, 4$ or 5 vertices.

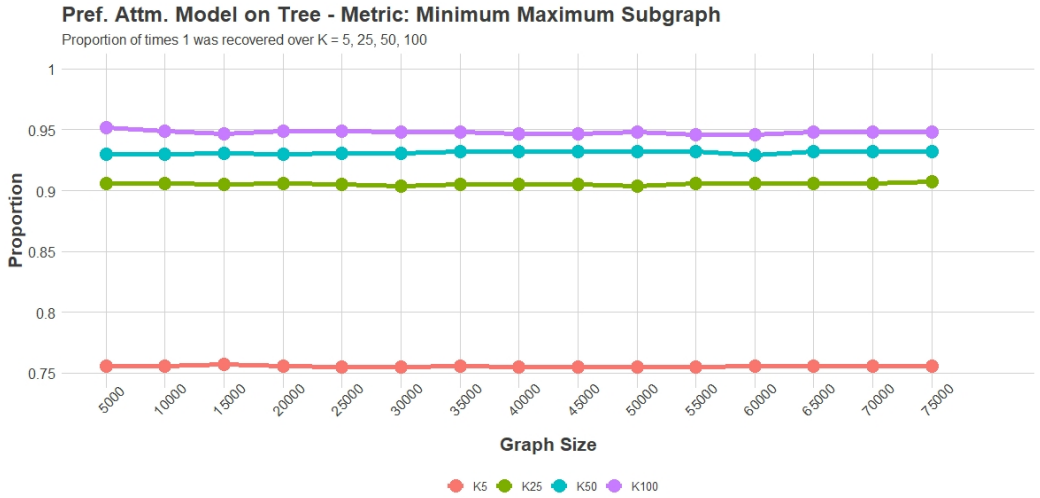

Figure 7: Recovery rates of vertex 1 on *preferential attachment* model on tree across different graph sizes when returning $5, 25, 50$ or 100 vertices.

We would like to now compare the two methods:

$$\underbrace{\text{minimum maximum connected component}}_{\text{method } (A)} \quad vs. \quad \underbrace{\text{maximum degree}}_{\text{method } (B)}$$

Figure 8 gives a look at how the two methods fare. We have some observations:

1. The naive approach of returning vertices of maximum degree yielded recovery rates that closely resembled that of the proposed method.

2. Across all sizes, the accuracy from method (A) was strictly better than that of method (B).

3. Although less accurate, the accuracy gaps between the two methods were relatively small, at most 15% when we wanted to return less than 5 vertices, but reduced to less than 5% as soon as we were willing to return 50 vertices or more, as seen in figures 9 and 10.

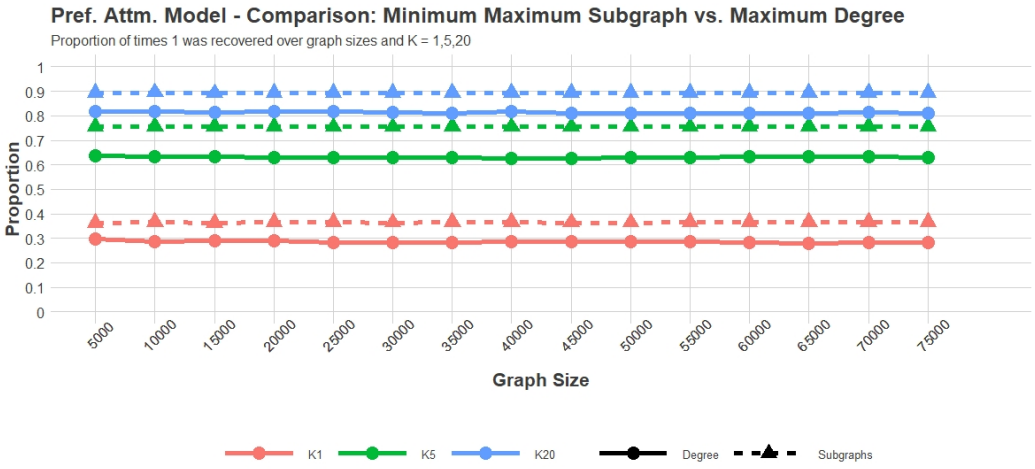

Figure 8: Recovery rates of vertex 1 on *preferential attachment* model on tree across different graph sizes when returning $1, 5$, or 20 vertices over the two methods.

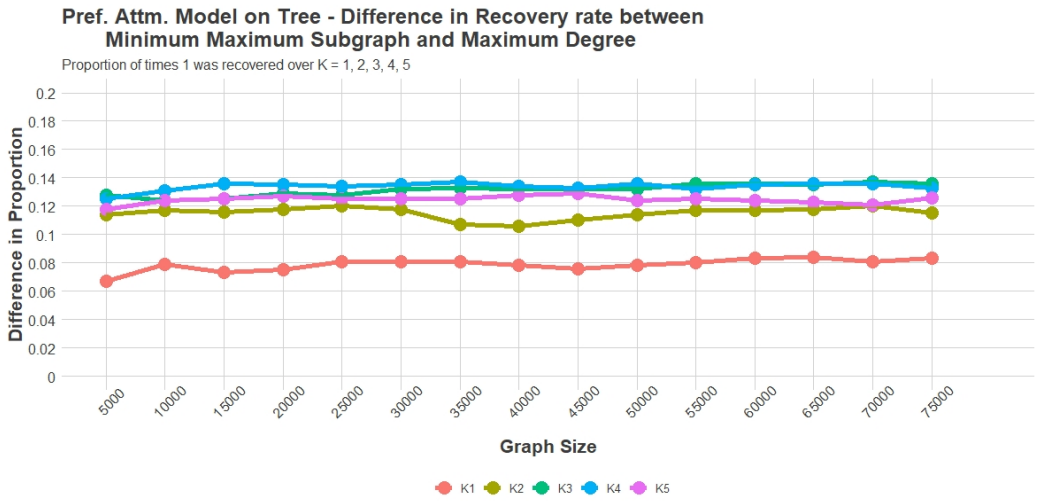

Figure 9: Differences in the recovery rates of vertex 1 on *preferential attachment* model on tree between the two methods when returning only $1, 2, 3, 4$ or 5 vertices.

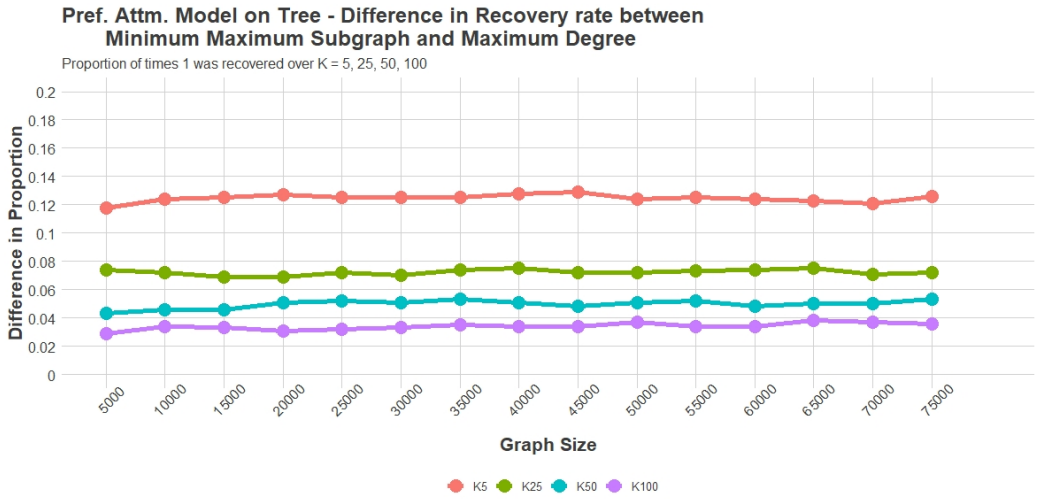

Figure 10: Differences in the recovery rates of vertex 1 on *preferential attachment* model on tree between the two methods when returning only $5, 25, 50$ or 100 vertices.

3.4 Proposed Improvements

Inspired by the analysis, we have two observations regarding the two methods of

$$\underbrace{\text{minimum maximum connected component}}_{\text{method } (A)} \quad vs. \quad \underbrace{\text{maximum degree}}_{\text{method } (B)}$$

We also note that when we say "computing φ values", we are referring to method (A):

1. Their accuracy rates closely resemble each other.
2. Method (A) can get very expensive compared to method (B), indeed while computing vertex degree can be done in real time, computing φ value even for only $5,000$ vertices per simulation can take more than 24 hours when the sizes get above $70,000$ for each batch of $1,000$ simulations for instance.

As such, we desire to keep the accuracy while bringing down the computation cost. We propose a combination of both methods.

We consider a 2-step procedure, in which we first look at M vertices with the maximum degree, to narrow down the set of vertices on which we now compute the φ value. We can describe the algorithm as:

Algorithm 2 Recovering vertex 1: Variant 1

1: Input: an unlabeled tree graph $T^0(n)$
2: Step 1: Return M vertices with the maximum degree
3: Step 2: Compute φ values among those M vertices
4: Output: K vertices with the smallest φ values

We now run the proposed algorithm on the same set of data. Figures 11 and 12 show the result. We have some observations:

1. There appeared no significant (or even observable) changes in the accuracy when we computed the φ values on only a subset of vertices compared to that on all the vertices.
2. This gives even more support on how robust method (A) is: whereas the theoretical result can be not quite as concise as we would wish, in practice, the algorithm can get both precise and concise very quickly.

The simulation gives us supporting evidence for our proposed algorithm, that to save computation time, we can incorporate the naive method of looking at the vertex degree to narrow down the set of suspected vertices.

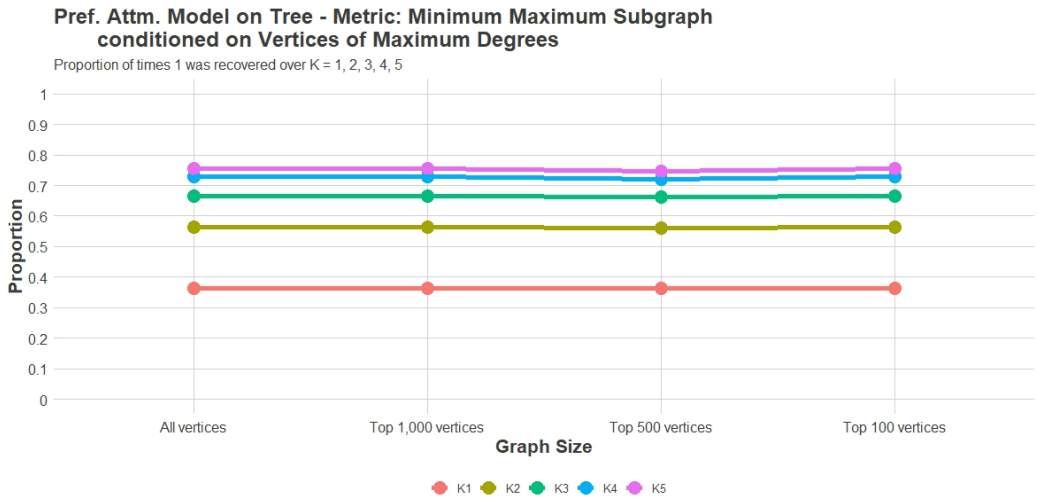

Figure 11: Recovery rates of vertex 1 on *preferential attachment* model on tree under the proposed 2-step algorithm to narrow down the set of vertices we want to compute φ, and returning only $1, 2, 3, 4$, or 5 vertices.

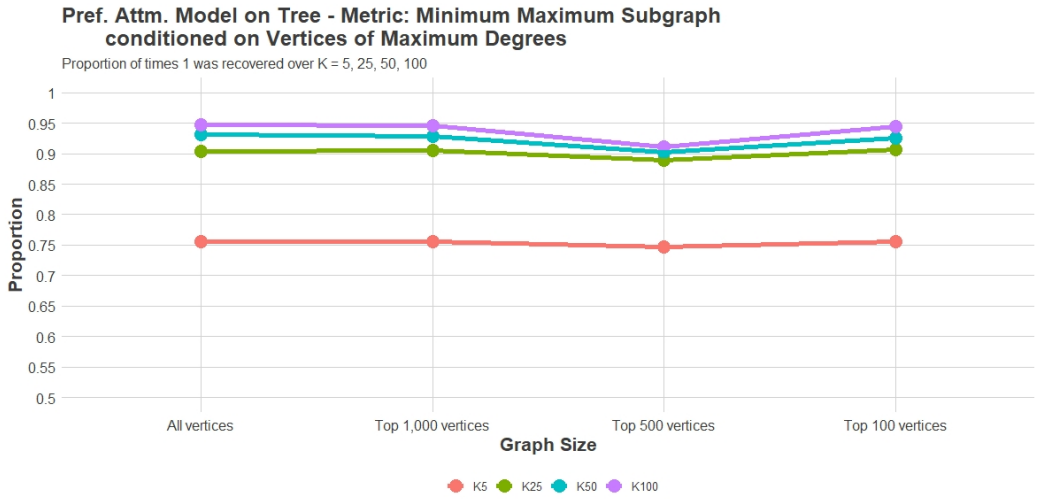

Figure 12: Recovery rates of vertex 1 on *preferential attachment* model on tree under the proposed 2-step algorithm to narrow down the set of vertices we want to compute φ, and returning $5, 25, 50$ or 100 vertices.

4 Recovering First L Vertices

4.1 Set-up and Objective

4.1.1 Set-up

We follow the same set-up introduced in section 3.1.1. In particular, we focus on the *preferential attachment* model on tree.

4.1.2 Objectives

Generalizing the idea of recovering the seed graph G^0, we would like to now recover the first L vertices. Ideally, we look for an algorithm that could be as accurate and precise as possible, similar to the performances we see in section 3.3.2.

4.2 Algorithm

A natural first guess is to simply apply algorithm 3.2 and possibly return more vertices. At the same time, a different and common approach is to do a local search conditioned on what we know at the current stage. The idea is popular, such as by Frieze and Pegden in [6], where the authors propose using local search starting at an arbitrary vertex u to recover the vertex 1. We will apply the idea but to recover L vertices.

We propose a *sequential local search* algorithm in stages, where at each stage we consider only the set of neighboring vertices of the current known vertices and then update those known vertices.

The algorithm can be described iteratively as:

Algorithm 3 Recovering first L vertices via local search

Input: an unlabeled tree graph $T^0(n)$
Initialize a list $\mathcal{M} = \emptyset$ by default
for each $l \in 1 : L$ **do**
    if $\mathcal{M} = \emptyset$ **then**
        Return K vertices with the smallest φ values
    else
        Create $\mathcal{X}$ containing neighbors of all vertices in $\mathcal{M}$
        Choose from $\mathcal{X}$ K vertices with the smallest φ values
    end if
    Update $\mathcal{M}$ with those K vertices returned

In other words, we start by recovering vertex 1. Having returned a set of vertices which we believe contain 1, denoted by K_1, we look at the vertices' neighbors, and choose from them another set of vertices with the smallest φ values, denoted by K_2. We note that $K_1 \cap K_2 = \emptyset$, and we believe in the likelihood that $\{1, 2\} \subset K_1 \cup K_2$. To recover subsequent vertices, we repeat the same procedure.

We conjecture two observations:

Conjecture 4.1

Let $G(n)$ be a graph grown under a preferential attachment model on tree with an increasing label starting from 1. Let $u > 1$ be a vertex not 1. Let $\mathcal{X} = N(t)$ be the set of vertices neighboring t. Then

$$\forall v \in \mathcal{X} : \quad \lim_{n\to\infty} \mathbb{E}\Big[\big|(T,u)_{\{1\}}\big|\Big] \;\geq\; \mathbb{E}\Big[\big|(T,u)_{\{v\}}\big|\Big]$$

where $(T,u)_{\{v\}}$ is the subtree rooted at v not containing u, and $(T,u)_{\{1\}}$ is the subtree rooted at a child of u containing 1 and not containing u.

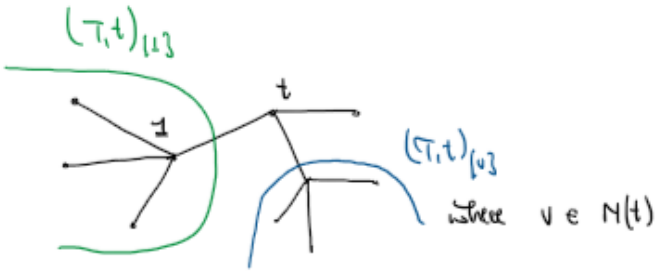

Figure 13: Examples of how conjecture 1 applies.

Conjecture 4.1 asserts that as $n \to \infty$, for any vertex $u > 1$, we can expect that computing $\varphi(u)$ reduces to computing the size of the subtree containing vertex 1.

Conjecture 4.2

Let $G(n)$ be a graph grown under a preferential attachment model on tree with an increasing label starting from 1. Let $L = [l]$ for $l \geq 1$ be fixed. Denote $u = l + 1$, the first vertex after the first l vertices. Let $\mathcal{X} = N(L)$ be the set of vertices neighboring L. Then

$$\forall v \in \mathcal{X} : \quad \lim_{n\to\infty} \mathbb{E}\Big[\big|(T,u)_{\setminus L}\big|\Big] \;\geq\; \mathbb{E}\Big[\big|(T,v)_{\setminus L}\big|\Big]$$

where $(T,u)_{\setminus L}$ is the subtree rooted at u not containing L.

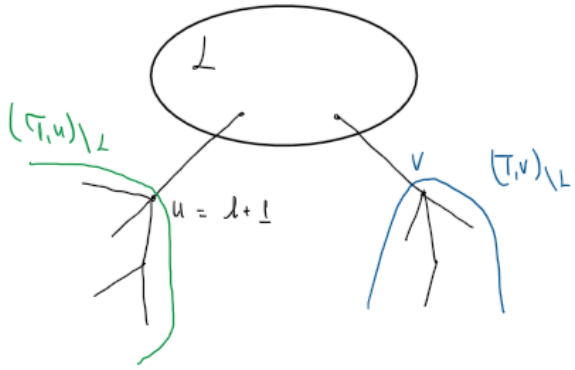

Figure 14: Examples of how conjecture 2 applies.

Conjecture 4.2 asserts that for any vertices u and v neighboring the first L vertices, as $n \to \infty$, if $u < v$ (u appears before v), then we can expect that the size of the subtree rooted at u not containing L is at least as big as that rooted at v.

The proofs are still a work in progress at this point.

Assuming the two conjectures, we combine them to give support to the proposed algorithm: suppose we are at step l and now have a set of vertices from previous steps, denoted by $\mathcal{X}$ (which corresponds to L in figure 14:

1. For any vertex $u \in N(\mathcal{X})$, it is reasonable to expect that calculating $\varphi(u)$ reduces to looking at the size of the subtree rooted at a child of u which contains $\mathcal{X}$.

2. For any u and $v \in N(\mathcal{X}$, comparing $\varphi(u)$ and $\varphi(v)$ reduces to comparing the sizes of $(T, u)_{\setminus L}$ and $(T, v)_{\setminus L}$, as seen in figure 14, since the portions containing L and what grow out of L are the same.

3. In comparing $(T, u)_{\setminus L}$ and $(T, v)_{\setminus L}$, if $u < v$ (u appeared before v), then it is reasonable to expect that $|(T, u)_{\setminus L}| \geq |(T, u)_{\setminus L}|$, which implies that $\varphi(u) \leq \varphi(v)$.

4. Hence, in doing pairwise comparison of the φ values, we could expect that vertex $L + 1$ will have the minimum φ value.

4.3 Simulations

4.3.1 Simulation Data

We would like to study the two algorithms proposed. We will test the algorithms on the same data set used in section 4.3.1..

4.3.2 Simulation Results

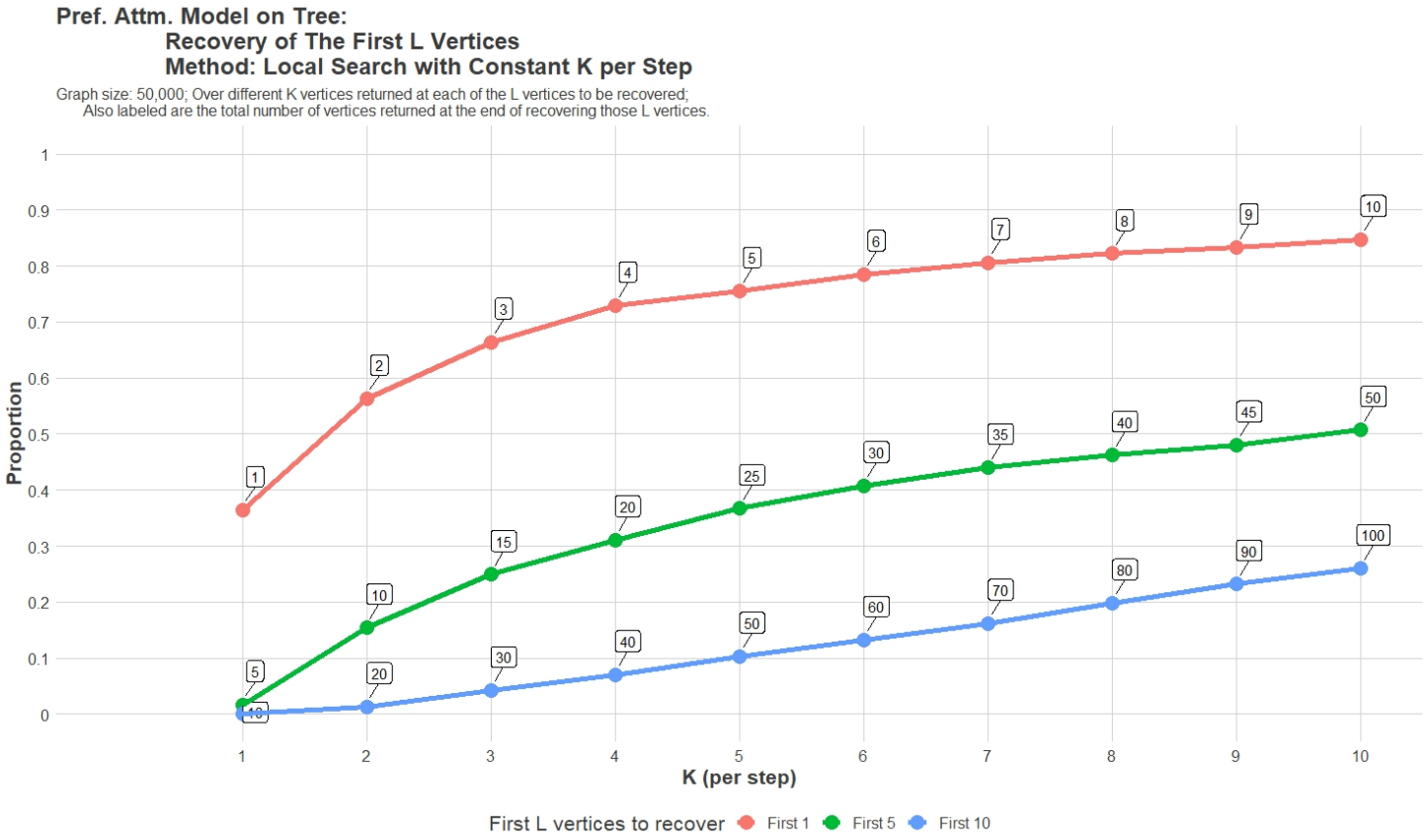

Figure 15: Recovery rates of the first L vertices on *preferential attachment* model on tree via the *sequential local search* across different K from 1 to 10 at each step on graphs of size $50,000$. Also labeled is the total number of vertices returned at the end of the search.

Figure 15 shows the recovery rate of the first $1, 5$ and 10 vertices under the *local search* algorithm on graphs of size $50,000$. At each step, we returned K from 1 to 10 vertices. We

have some observations:

1. The recovery rates decreased significantly when we wanted to recover 5 and 10 vertices, compared to simply recovering vertex 1.

2. Although the recovery rates appear low, they appear to be increasing at least linearly.

Figure 16 shows the recovery rate of the first 5 and 10 vertices under the *local search* algorithm across different graph sizes. At each step, we returned K being $1, 5$ and 10 vertices. We have some observations:

1. Fixing the number of vertices returned per step (K), the recovery rates remained essentially unchanged whether we were recovering the first $1, 5$ or 10 vertices.

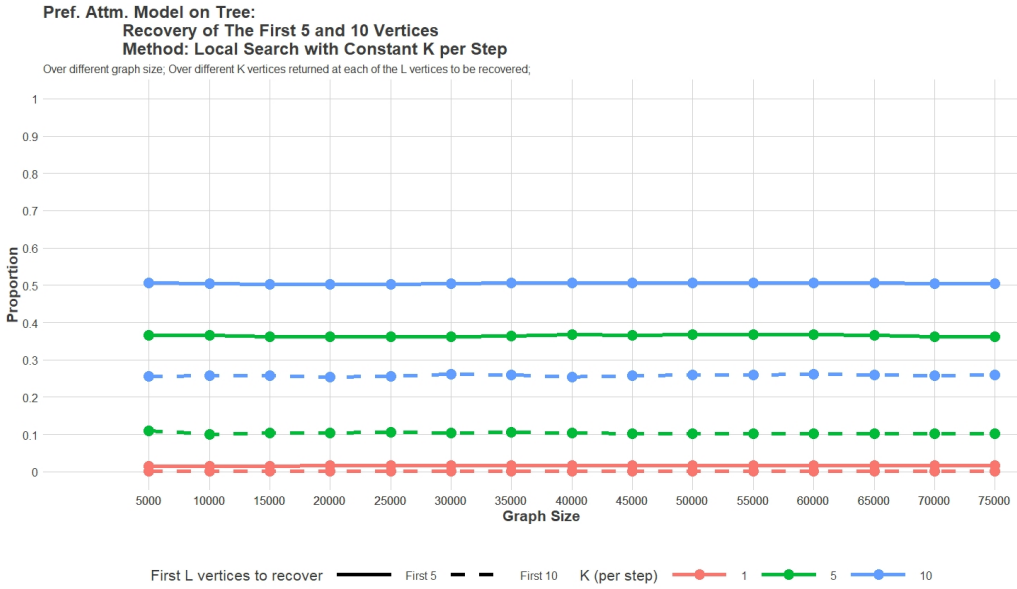

Figure 16: Recovery rates of the first 5 and 10 vertices on *preferential attachment* model on tree via the *sequential local search* with $K \in \{1, 5, 10\}$ across different graph sizes.

We note the significant observation of the recovery rates being stable across a wide range of graph sizes. This gives an impression that we would only need to tune K, the number of vertices returned at each step, to increase the recovery rate. This behavior is further confirmed by figures 17 and 18.

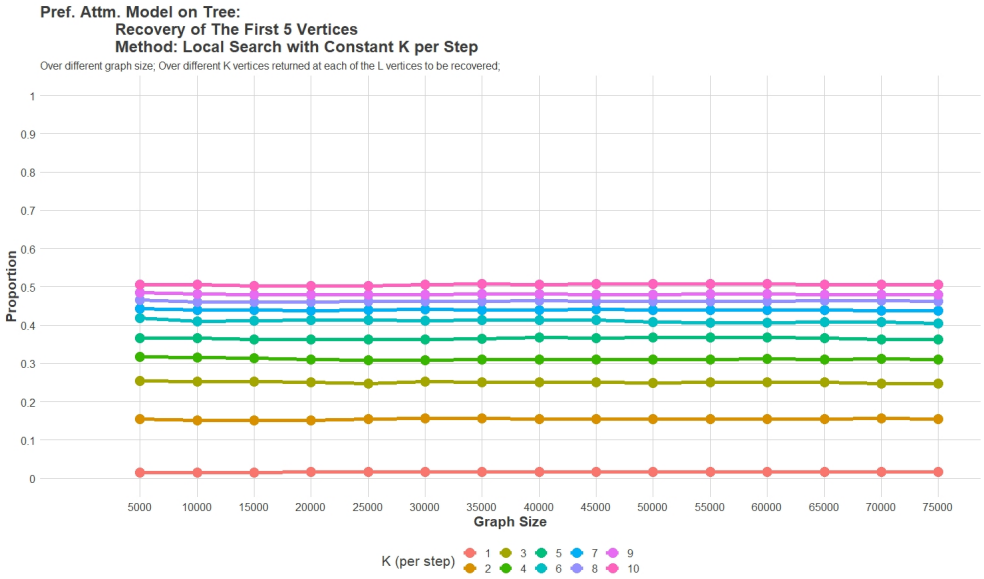

Figure 17: Recovery rates of the first 5 vertices on *preferential attachment* model on tree via the *sequential local search* across different K.

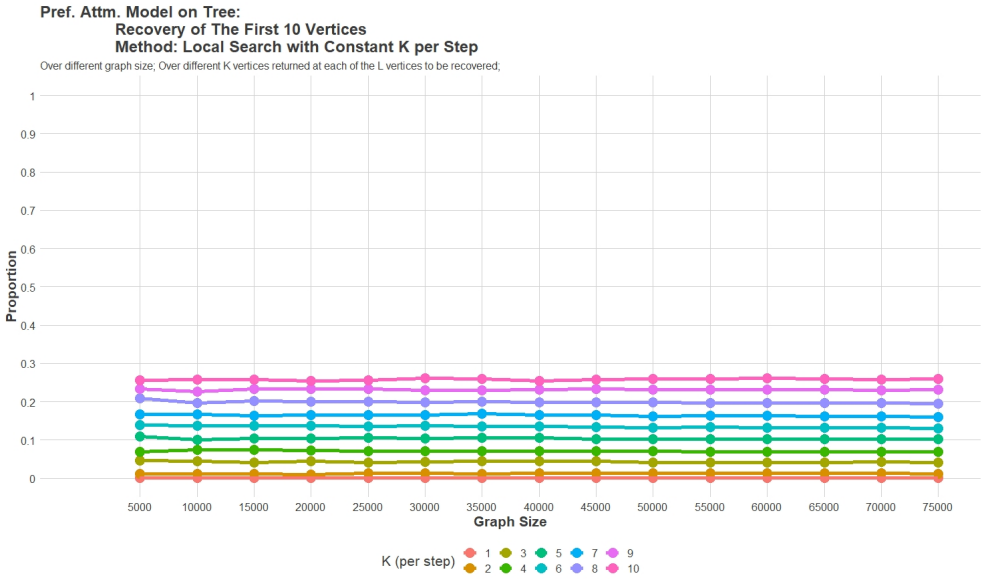

Figure 18: Recovery rates of the first 10 vertices on *preferential attachment* model on tree via the *sequential local search* across different K.

Given those observations, we can now compare the *local search* algorithm with the simple search of simply returning K vertices with the smallest φ values.

Figure 19 shows the recovery rates between the two algorithms. We have some observations:

1. There appear no significant differences between the two algorithms: for the same number of vertices returned, the accuracy of a *simple search* closely resembles that of the more complicated *simple search*.

2. However, when returning more vertices, the gap in the accuracy appear to be widening, as illustrated by the gaps on the far right of each set of curves.

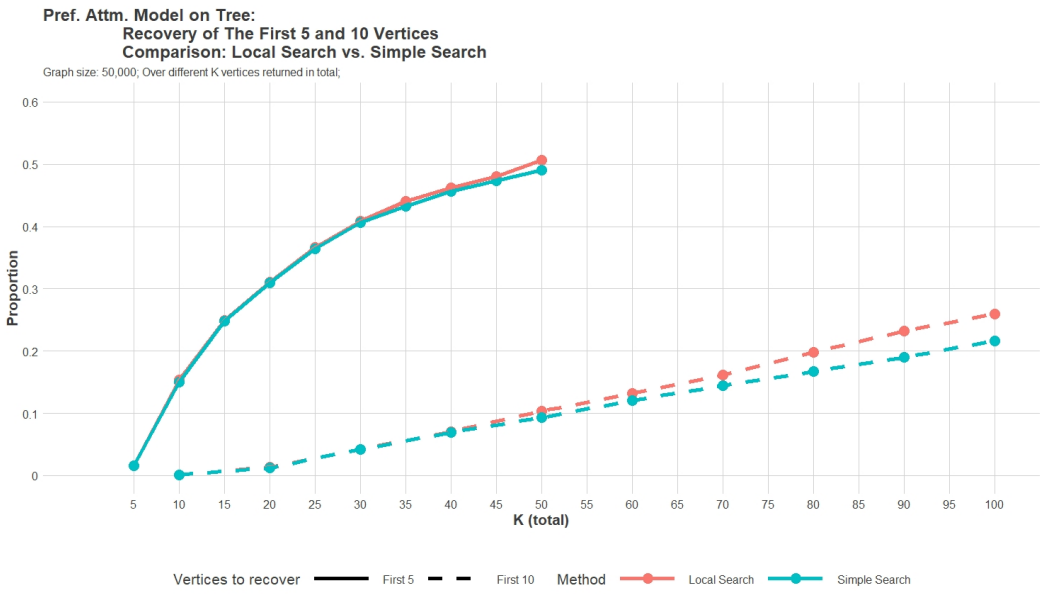

Figure 19: Comparison of the recovery rates of the first 5 and 10 vertices via the *local search* and the *simple search* algorithms.

Those observations give us an impression that while there is little difference between the two algorithms initially, the gap would start to grow as we return more vertices. We plan to further study the trade-off between accuracy and precision of returning the first L vertices.

4.4 A Second Look at the Problem

We note the significant drop in the accuracy rates when we started recovering more than just vertex 1. We would like to look further into this behavior.

We will use a variant of the *local search* algorithm as described in 3. In particular, we will look to recover vertex L assuming that we know exactly all the $\mathcal{X} = \{1, 2, \ldots, L-1\}$ vertices and will search among the neighbors of $\mathcal{X}$. We will refer to this as *local search with perfect knowledge*. The objective is to study how hard the problem becomes once we start looking for vertices away from 1.

Figures 20, 21 and 22 show the recovery rates of a local search with perfect knowledge, using the methods of both the φ values and degree. We have some observations:

1. Comparing between recovering vertex 2 and vertex 10, the drops in accuracy can be significant.
2. We could expect the accuracy of using the maximum degree to closely match that of using the φ values for the earliest vertices. However, as we move further away from vertex 1, the difference in accuracy can get more significant, as illustrated in figure 22.

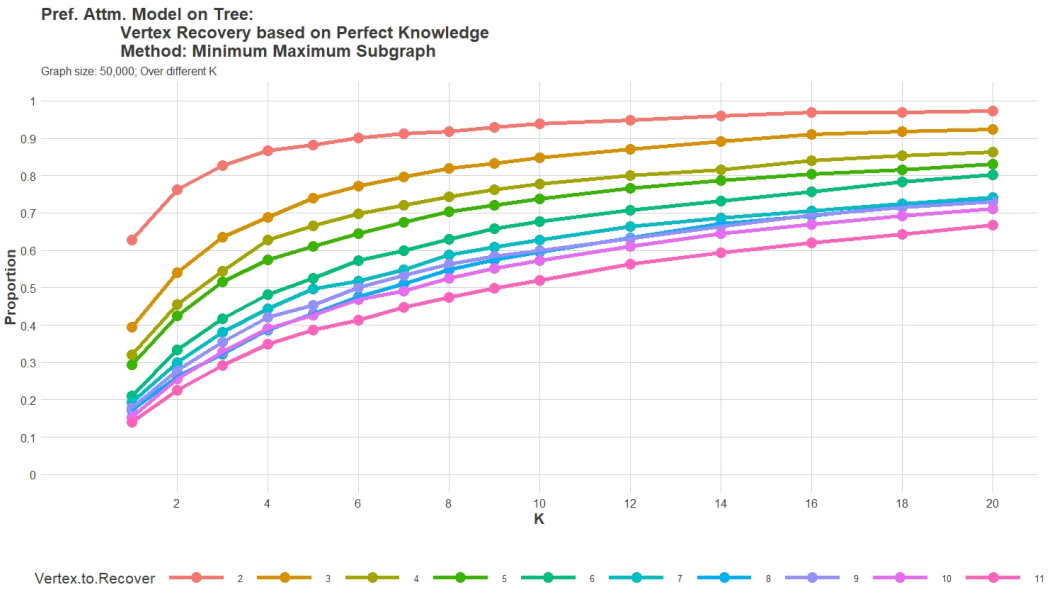

Figure 20: Recovery rates of the first 11 vertices via a local search with perfect knowledge using φ values.

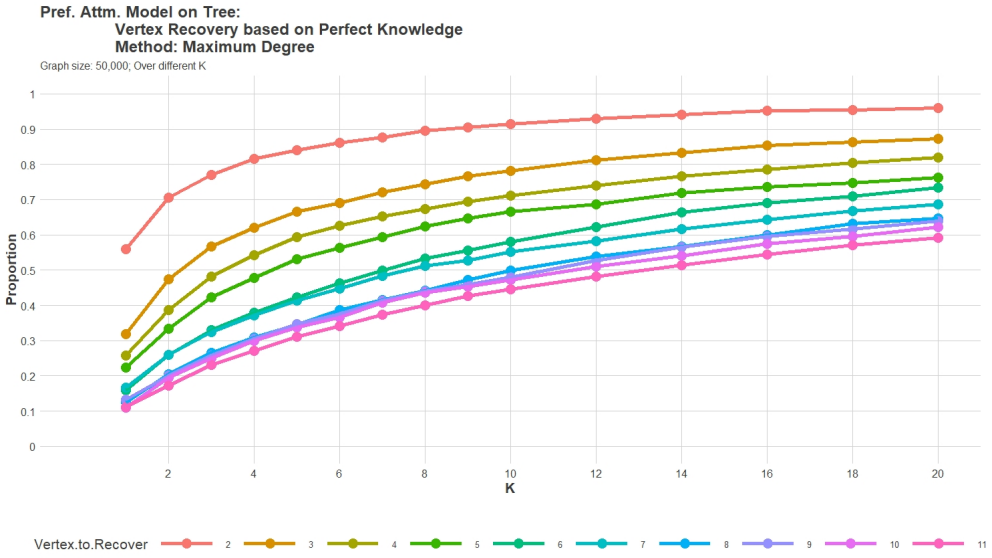

Figure 21: Recovery rates of the first 11 vertices via a local search with perfect knowledge using vertex degree.

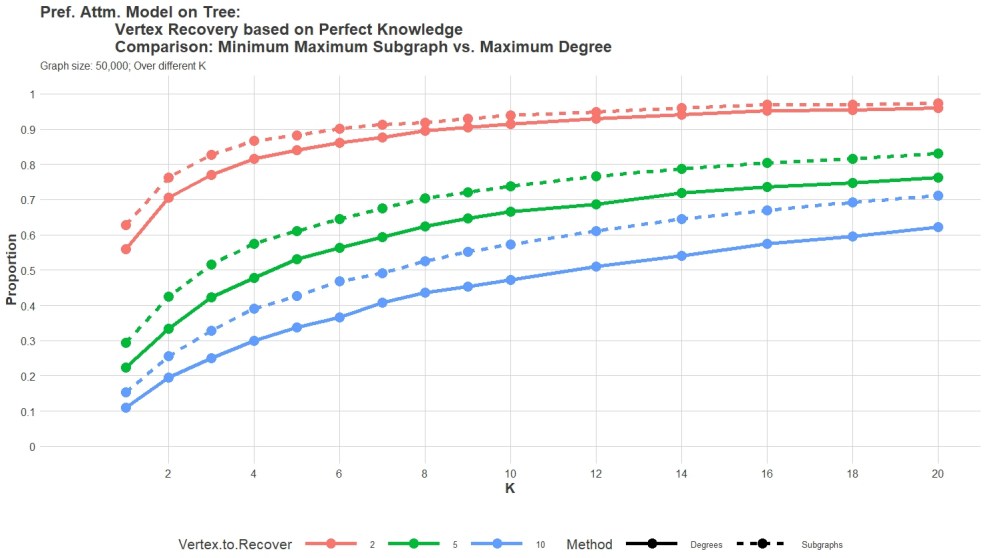

Figure 22: Comparison of the recovery rates via a local search with perfect knowledge.

We observe that the problem of recovering vertices after 1 can get hard quickly. Although the experiment appears to show that it would still be possible to achieve certain high level of accuracy, we will have to trade a lot in terms of being concise. In particular, we will have to return a much larger range of vertices to get to certain accuracy.

This observation is indeed noted by Lugosi and Pereira in [8]. The paper studies the same problem of recovering the seed graph. Although the paper focuses only *uniform attachment* model on tree, we suspect that the result will carry to a large extent over to that of *preferential attachment* model on tree. The paper shows that even when we assume additional structures on the seed graph, as soon as the seed graph exceed certain size, any algorithm must miss at least half the number of vertices with probability at least some ε. We cite the results here.

Theorem 4.1: Uniform attachment model on tree, seed graph is a path, [8]

Let $\varepsilon \in (0, e^{-e^2})$. Suppose that T_n is a uniform attachment tree with seed $S_l = P_l$ (being a path) for

$$l \leq \frac{\log(1/\varepsilon)}{\log\log(1/\varepsilon)}$$

then for any $n \geq 2l$ (n is the tree size), any seed-finding algorithm that outputs a vertex set H_n of size l has

$$\mathbb{P}\Big(\big|H_n \cap P_l\big| \leq \frac{l}{2}\Big) \geq \varepsilon$$

Theorem 4.2: Uniform attachment model on tree, seed graph is a star, [8]

Let $\varepsilon \in (0, e^{-e^2})$. Suppose that T_n is a uniform attachment tree with seed $S_l = E_l$ (being a star) for

$$l \leq \frac{\log(1/\varepsilon)}{\log\log(1/\varepsilon)}$$

then for any $n \geq 2l$ (n is the tree size), any seed-finding algorithm that outputs a vertex set H_n of size l has

$$\mathbb{P}\Big(\big|H_n \cap P_l\big| \leq \frac{l}{2}\Big) \geq \varepsilon$$

5 Conclusion

In this paper, we studied the problem of recovering the first vertices in a model assumed to have grown under the *preferential attachment* rules. We focused on a special case where the graphs are trees. We revisited the approach of looking at the maximum subgraph created when removing each vertex. We provided a more detailed proof and studied the efficiency with simulations. We summarize the key findings below:

1. In recovering vertex 1, we can get accurate in terms of finding a set of vertices in which 1 belong and that the set is small very quickly. In particular, we can achieve an accuracy of about 90% by returning only 20 vertices regardless of the graph size, as seen in figure 4.

2. Once we want to recover more than just vertex 1, the problem becomes much harder quickly. Not only is the drop in accuracy significant, but we would now need to return a lot more vertices to be of certain confidence that the first L vertices have been fully recovered.

3. This phenomenon brings to our attention the trade-off between being

   $$\underbrace{\textit{accurate}}_{\substack{\text{that } L \text{ vertices} \\ \text{have been recovered}}} \quad \textit{vs.} \quad \underbrace{\textit{precise}}_{\substack{\text{that only small number} \\ \text{of vertices have been returned}}}$$

   The findings from section 4.3.2 suggest that if we are willing to forgo being precise, we could improve the accuracy, although the trade-off appear significant.

4. Once we have fixed the objective (number of vertices to recover), the method (either using vertex degree or the φ values), and the approach (local search of simple search), the accuracy remains constant regardless of graph sizes, ranging from size of $5,000$ to even 15 times larger of size $75,000$. This suggests that the difficulty of the problem lies in the structure of such a model and not so much on the size.

5. Although always trailing behind in terms of accuracy, simply looking at the vertex degree can get us quite far, compared to using the more complicated algorithm of looking at the φ values. However, computing vertex degree is straightforward and is much cheaper. As such, when the computation cost can get expensive, we propose a two-step approach where (1) we look at the vertex degree to narrow down the set of suspected vertices, and then (2) compute the φ values with little compromise on the accuracy, as seen in section 3.4.

6 Work in Progress

Here are some works currently in progress:

1. Conclusively proving the two conjectures proposed in section 4.2.
2. Further studying the trade-off between accuracy and precision in recovering the first L vertices.
3. In recovering the first L vertices via the *sequential local search*, K (the number of vertices returned at every step) can be treated as a parameter. The paper studied a special case where K was kept constant.
4. A variant of the *preferential attachment* model on tree: we are considering a *preferential attachment* model on triangle where at each new iteration, instead of choosing a random vertex, we choose a random edge and connect the new vertex to the two end vertices, forming a new triangle.
5. The computation cost of computing the φ values, and compare that with computing vetex degree.

7 References

[1] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[2] B´ela Bollobás, Oliver Riordan, Joel Spencer, and Gábor Tusnády. The degree sequence of a scale-free random graph process. *Random Structures & Algorithms*, 18(3):279–290, 2001.

[3] Sébastien Bubeck, Luc Devroye, and Gábor Lugosi. Finding adam in random growing trees. *Random Structures Algorithms*, 50(2):158–172, Nov 2016.

[4] F. Chung, F.R.K. Chung, F.C. Graham, L. Lu, Conference Board of the Mathematical Sciences, National Science Foundation (U.S.), P.R.M.K.F. Chung, American Mathematical Society, and K.F. Chung. *Complex Graphs and Networks*. Conference Board of the mathematical science. American Mathematical Society, 2006.

[5] S. N. Dorogovtsev, J. F. F. Mendes, and A. N. Samukhin. Structure of growing networks with preferential linking. *Physical Review Letters*, 85(21):4633–4636, Nov 2000.

[6] Alan Frieze and Wesley Pegden. Looking for vertex number one, 2014.

[7] Remco Hofstad. *Random Graphs and Complex Networks*. 02 2008.

[8] Gabor Lugosi and Alan S. Pereira. Finding the seed of uniform attachment trees, 2018.

[9] Derek De Solla Price. A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5):292–306.