Notes on Social and Behavioral Networks

Cristian Di Pietrantonio

October 8, 2017

Contents

Chapter 1

Meme flow

On the web, information is published by one or more *sources* and often spreads quickly to other parties. The most popular kind of information that spreads quickly is a meme. From Wikipedia:

> A meme is an idea, behavior, or style that spreads from person to person within a culture - often with the aim of conveying a particular phenomenon, theme or meaning represented by the meme.

In this chapter we are going to define a mathematical model that describes the spread of information. We call *node* an entity on the web that holds the information of interest; it can be a blog, a website or a social network profile. Nodes can receive or take information from other nodes, enabling the spread of the meme. Nodes can also publish the same meme independently (without "copying" each other). Whenever a node publishes information, a timestamp of the event is also available. At the end, considering a some particular meme, all we can see is a bunch of nodes having published with an associated timestamp. What we would like to know is the *social graph* that links these nodes, i.e., who retrieves information from who.

1.1 The Trace Spreading Model

Authors in [**?**] introduce the Trace Spreading Model, where a *trace* is a sequence of nodes ordered by their timestamp.

Define an undirected graph $G = (V, E)$, where:

- V is the set of nodes that holds the information of interest;
- There is only a source of information, chosen uniformly at random;
- $e = \{v, v'\} \in E$ iif information propagated from v to v', $v, v' \in V$ (the actual direction is not important, as you will notice later).

- Each edge $e = \{v, v'\}$ has a *weight* $w(e)$ sampled i.i.d. in $Exp(\lambda)$, which represent the time needed for the information to propagate from v to v'. In fact, for our purpose, any distribution will do (i.i.d. is the important part).

In this model, the trace follows the shortest path tree from the source.

So, having this model and a set of traces (and every trace contains all the nodes), is it possible to reconstruct the unknown social graph G?

1.2 The First-Edge Algorithm

It turns out that having $O(n\Delta \log n)$ traces is sufficient for (perfect) reconstruction with high probability. In some graphs, $\Theta\left(\frac{n\Delta}{\log^2 n}\right)$ traces are necessary. In the above statement, n is the number of nodes in the graph and Δ is its maximum degree.

Input: T, the set of traces
$E \leftarrow \{\}$ //the set of edges in the graph ;
for $t \in T$ **do**
  $u \leftarrow t[0]$ //the first node in t;
  $v \leftarrow t[1]$ //the second node in t;
  $E \leftarrow E \cup \{(u, v)\}$;
end

Algorithm 1: The first-edge algorithm.

Consider Algorithm 1. What it does is simply adding an edge between the first and second node in a trace, for every trace. That is because we can only be sure about the existence of that edge. The first node in a trace is the source (for that trace), since it has the lowest timestamp; the second node, which has the second lowest timestamp, could only get the information from the first node.

Theorem 1. *If there are $|T| \geq cn\Delta \log n$ traces, the first-edge algorithm can reconstruct the unknown graph $G = (V, E)$ with probability $p \geq 1 - \frac{1}{n^{2(c-1)}}$, where $n = |V|$, $\Delta = \max_v \deg(\mathrm{v})$ and c is a constant.*

Proof. An edge $\{u, v\}$ will be in the graph if and only if u and v appears at the beginning of at least one trace. We will prove that it happens with high probability.

Consider the following event.

$$\xi_1 = \text{“A given edge } \{u, v\} \in E \text{ is inferred using the trace } t.\text{”}$$

The probability of ξ_1 is given by

$$Pr\{\xi_1\} = Pr\{t[0] = x\} Pr\{t[1] = y, y \in \{u, v\} \setminus \{x\} \mid t[0] = x\},\ x \in \{u, v\} \tag{1.1}$$

$$= \frac{2}{n} \frac{1}{deg(y)}, y \in \{u, v\} \tag{1.2}$$

$$\geq \frac{2}{n\Delta}. \tag{1.3}$$

In other words it is the probability of one of the two nodes u, v to appear as first node in the trace multiplied by the probability that the other appears as second node in the trace. Since the source is selected uniformly at random, every node has the same probability $\frac{1}{n}$ to be picked; for us, either u or v will do. Next, assume without loss of generality (wlog) that u is picked as first node. The second node will now be drawn from the neighbors of u (convince yourself that there can't be a node y after the first node in the trace without it being one of its neighbors). The probability that the neighbor picked is u is $\frac{1}{deg(u)}$ that can be overestimated using Δ.

Now, the complementary event of ξ_1 is "the edge $\{u, v\}$ is not inferred using trace t" and its probability is $(1 - \frac{2}{n\Delta})$. For the edge not to be inferred at all, there should be no trace from which that edge can be extracted.

$$Pr\{\textit{An edge in the graph is not inferred}\} = \left(1 - \frac{2}{n\Delta}\right)^{|T|} \leq \left(1 - \frac{2}{n\Delta}\right)^{cn\Delta \log n} \tag{1.4}$$

We will show that this probability is tiny. To do so, we must use the following lemma.

Lemma 1.

$$1 - x \leq e^{-x} \tag{1.5}$$

Proof. Studying the function we see that

$$e^{-x} = (-e^{-x})' = (e^{-x})'' > 0$$

so its tangent in every point is below the function. Taking the tangent in $x_0 = 0$ we have

$$\begin{aligned} e^{-x} &\geq (e^{-x_0})'x + e^{-x_0} \\ &= -e^{-x_0} + 1 \\ &= -x + 1 \\ &= 1 - x. \end{aligned}$$

□

Going back to our proof we have

$$\begin{aligned} Pr\{\textit{Edge } \{u,\ v\} \textit{ in the graph is not inferred}\} &= \left(1 - \frac{2}{n\Delta}\right)^{|T|} && (1.6) \\ &\leq \left(1 - \frac{2}{n\Delta}\right)^{cn\Delta \log n} && (1.7) \\ &\leq \left(e^{-\frac{2}{n\Delta}}\right)^{cn\Delta \log n} && (1.8) \\ &= e^{\log n^{-2c}} && (1.9) \\ &= \frac{1}{n^{2c}}. && (1.10) \end{aligned}$$

Finally, we want to compute the probability that the algorithm fails. Let

$$\xi_{a,b} = \text{“}\textit{Edge } \{a,\ b\} \textit{ is not inferred by the algorithm”}.$$

Then

$$\begin{aligned}
Pr\{\textit{The algorithm fails}\} &= Pr\left\{ \bigcup_{\{a,b\}\in E} \xi_{a,b} \right\} && (1.11)\\
&\leq \sum_{\{a,b\}\in E} Pr\{\xi_{a,b}\} && \text{(by union bound)}\\
&= |E| Pr\{\xi_{a,b}\} && (1.12)\\
&\leq n^2 \frac{1}{n^{2c}} && (1.13)\\
&= \frac{1}{n^{2(c-1)}}. && (1.14)
\end{aligned}$$

Notice that we can make this probability tiny as we want by increasing c, i.e. by increasing the number of traces. □