MS&E 233 Lecture 8: Applications of PageRank to Recommendation Systems

Ashish Goel, scribed by Hadi Zarkoob

April 25

In the last class, we learnt about PageRank and Personalized PageRank algorithms. We saw that these algorithms can be used to rank nodes in a graph based on *network measures*. In this class we will see some applications of these algorithms.

- **Ranking tweets in Twitter**: To use PageRank for ranking tweets in Twitter we can construct a synthetic graph as follows. Represent each user and each tweet by a node. Draw a directed link from user A to B if A follows B. Also, draw a directed edge from a user A to a tweet t if A tweets or retweets t (see Figure 1). Now we can apply PageRank algorithm on this graph to obtain a ranking for tweets. If we ignore computational details, this algorithm provides a reasonable approach for ranking tweets in Twitter.

  The above method provides a global rank for each tweet. We can use a similar technique to obtain personalized rankings for a given user. To this end, we should use Personalized PageRank: We initiate a random walk from the user of interest and teleport back to her with some fixed probability at each step.

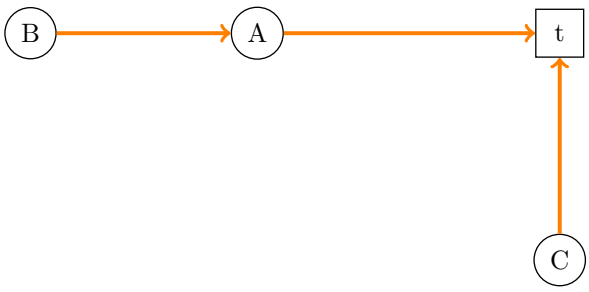

Figure 1: A synthetic graph for ranking tweets in Twitter. User B follows user A. User A tweets t. User C retweets t.

- **Recommendation systems**: Consider a system that consists of a set of users and a set of products. Each user determines a set of products that she has enjoyed by now. An example could be a movie rental website such as Netflix. Can we use PageRank to suggest new movies for each user based on the interests of other similar users?

  Suppose I want to find some new movies in Netflix that match my taste. Consider the following high-level approach to this problem:

  - A movie is *relevant* for me if other *similar* people liked it.
  - A person is *similar* to me if they like movies that are *relevant* to me.

  These relations are actually circular. But similar to what we saw in the PageRank, we can write interdependent equations based on them; and solve the resulting system of equations under appropriate conditions.

  A first try could be as follows:

$$\text{rel}(m) = \sum_{\text{individuals } i \text{ who liked } m} \text{sim}(i) \tag{1}$$

$$\text{sim}(i) = \sum_{\text{movies } m \text{ which } i \text{ liked}} \text{rel}(m) \tag{2}$$

  Although it seems to be a reasonable idea, there is a problem with this approach. Suppose there are two people who are equally similar to me and consider a product which both of them like. If one of these people likes every single product in the market, and the other one likes only a handful of products, we expect the set of products from the second person to be much more informative about my interests. We can include this effect in our model by normalizing similarity of individuals in (1) by the number of movies they liked. Following a similar line of reasoning, we should normalize relevance of each movie in Equation (2) by the number of people who liked that movie:

$$\text{rel}(m) = \sum_{\text{individuals } i \text{ who liked } m} \frac{\text{sim}(i)}{\text{number of movies } i \text{ liked}} \tag{3}$$

$$\text{sim}(i) = \sum_{\text{movies } m \text{ which } i \text{ liked}} \frac{\text{rel}(m)}{\text{number of people who liked } m} \tag{4}$$

  Similar to the PageRank algorithm, we can provide a random walk interpretation for the above set of equations. To this end, consider a bipartite graph with two groups of nodes. The first group represent users and the second group represent products. Whenever user u likes product m, we draw two edges, one from node u to m and the other one from node m to

u. (see Figure 2). It is not hard to show that equations (5) and (6) give steady-state probabilities of a random walk over this bipartite graph. We can sort steady-state probabilities of nodes in the right-hand side of the graph to obtain a ranking for movies.

To solve the system of equations in (5) and (6), we need to impose the following condition:

$$\sum_{\substack{\text{all products } m \\ \text{in the market}}} \text{rel}(m) = 1$$

This makes the solution well-defined. It should be compared to the one that we imposed when designing the PageRank algorithm in the last class.

One should note that the above algorithm provides a global rank for movies. In many cases, we want to have a ranking of movies specific to a user, lets say u. To achieve this, we can introduce a teleportation component to the random walk as follows. During the random walk, whenever we are in a *product* node we teleport back to node u with probability ϵ, and with probability $1-\epsilon$ we continue the normal random walk. This teleportation enforces the random walker to explore the area around the user of interest. With this change, steady-state probabilities of the random walk will be given by:

$$\text{rel}(m) = \sum_{\text{individuals } i \text{ who liked } m} \frac{\text{sim}(i)}{\text{number of movies } i \text{ liked}} \tag{5}$$

$$\text{sim}(i) = (1-\epsilon) \sum_{\text{movies } m \text{ which } i \text{ liked}} \frac{\text{rel}(m)}{\text{number of people who liked } m} + \begin{cases} \epsilon & \text{if } i = u \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

In the last class we saw a problem with the naive PageRank algorithm was that the random walker (the pageRank monkey) might get stuck in a subset of graph which has no or only a few outgoing edges to the outside world. As in the pageRank algorithm, the teleportation scheme introduced above helps to avoid this problem in our algorithm.

- **Suggesting friends over Twitter**: First, we note that users in Twitter can play a role as producer and a role as a consumer. In light of this property, we propose the following algorithm for providing friend suggestions over Twitter. Make two copies of each individual in the network: A producer copy and a consumer copy. Use these copies to construct a market similar to that introduced in recommendation systems (Figure 2). If user u follows v, u plays the role of consumer and v plays the role of producer. In this case, draw an edge from u to v and another edge from v to u in the bipartite graph. Performing personalized PageRank on the resulting graph provides new friend (producer) suggestions for each user.

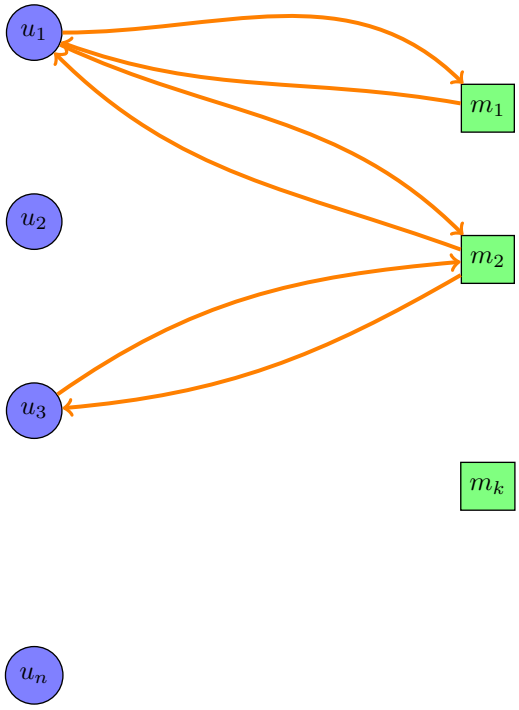

Figure 2: Application of Personalized PageRank for recommendation systems. Users are on the left-hand side and products are on the right-hand side.

- Empirical results [1] suggest that Personalized PageRank with normalized terms over-performs other methods while Personalized PageRank without normalizing terms performs rather poorly.
- **Conclusion**: PageRank is a powerful tool that ties search, advertising, recommendation and reputation systems. The merit of PageRank comes from its power in evaluating *network measures* in a connected system.

[1] See Appendix A of "Fast Incremental and Personalized PageRank", `http://arxiv.org/abs/1006.2880` . The PageRank-based algorithm described in class is called "SALSA" by the academic community.