REVIEW

Edge-Weighted Personalized PageRank: Breaking A Decade-Old Performance Barrier

Wenlei Xie, David Bindel, Alan Demers, Johannes Gehrke (KDD '15)

1. *Summary of the paper*

The goal of this paper is to increase propose the edge-weighted personalized Pagerank and to increase the calculation speed by using model reduction approach.

2. *Key contributions of the paper*

Up until now, there have been many methods to speed up the personalized, or 'edge-weighted' Pagerank methods. However, it has always been open problem to actually solve this fully. The paper proposes the model reduction to solve this issue, which outperforms previous methods. Speeding up the calculation process also helps the Pagerank system to work in online field (real-time calculation latency).

3. *Three strong points*

It's a stronger version of original Pagerank. Pagerank has been used widely, in variety of applications like object databases, social networks, and recommendation systems. It is a graph walker that randomly moves through the nodes in a graph, by transitioning to other nodes through edge and teleporting to random places. These walks will show vector x at time t, and this can be represented by the distribution of walker locations at time t, by the following equation on the left. Then, Pagerank vector x is the stationary vector, shown on right..

$$\boxed{x^{(t+1)} = \alpha P x^{(t)} + (1-\alpha)v}$$

$$\boxed{Mx = b, \text{ where } M = (I - \alpha P), \quad b = (1-\alpha)v.}$$

Since the original Pagerank is just a graph topology, we can add 'weight', or 'bias' to transition or teleport, shown below, left and right respectively.

$$\boxed{Mx(w) = (1-\alpha)v(w), \quad w \in \mathbb{R}^d}$$

$$\boxed{M(w)x(w) = (1-\alpha)v, \quad w \in \mathbb{R}^d.}$$

This makes the Pagerank more **personalized**. Then, if we make the newly proposed method very fast at calculation, this can be used with user feedback online. This can be done by **model reduction**, which reduces order models, which was often unfeasible to perform numerical simulations using the complete full order model. This is just SVD. But to speed up, it uses Discrete Empirical Interpolation Method (**DEIM**) method. Interpolation means to enforce the equations as a small number of nodes inside of the network so rather than thinking to enforce the Pagerank everywhere but to pick the few.

4. *Three weak points*

Although the difference is not that much thanks to highly researched methods, this must be mentioned. SVD makes the dimension smaller for faster calculation, but this makes it **inevitable for the calculation to not be as accurate as the original dimension calculation**. Also, although this part has been mentioned in future work section, **all these calculations must be done in the host-side**. If the Pagerank calculations can be done from each node, in other words, even if each node has computing ability (which usually does in online field), the central host must be doing the calculations. Lastly, even though the paper has proposed a better method of Pagerank, it still has its base on Pagerank. The fundamental problem of Pagerank, which is that the **old nodes that have been included before get weighted too much compared to the nodes that have been added later on.**

5. *Other general comments*

Pagerank is a great algorithm for GNN as well, because many Graph Neural Network models are based on Random Walk, and Pagerank, roughly speaking, is a way to organize random walk of various lengths. The difference is that Pagerank uses a positive real value α, where $\alpha \in [0, 1)$, to control the diffusion of a combination of random walks. If the computational cost of Pagerank can be reduced to the extent this paper has mentioned, implementing the method to GNN will be able to guide us to the new state-of-the-art performance in GNN.

Reference: https://www.math.ucsd.edu/~fan/wp/lov.pdf