FPPR: Fast Pessimistic PageRank for Dynamic Directed Graphs

Rohith Parjanya, Suman Kundu

Cognitive and Social Analytics Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology Jodhpur, India
{parjanya.1, suman}@iitj.ac.in

Abstract. The paper presents a new algorithm for updating PageRank on dynamic directed graphs after the addition of a node. The algorithm uses the expected value of the random surfer to calculate the score of the newly added node and nodes of the existing chain where the new node is added. The complexity of the algorithm for k updates is $\mathcal{O}(k \times d_{avg})$. Extensive experiments have been performed on different synthetic and real-world networks. The experimental result shows that the rank generated by the proposed method is highly correlated with that of the benchmark algorithm of Power Iteration.

Keywords: Dynamic Network, Randomized Algorithm, Link Sensitivity Index, Dynamic PageRank

1 Introduction

Calculating PageRank [13] in a dynamic network is an important and challenging research problem. A network is dynamic if its topology changes with time, either by adding nodes and edges or by deleting nodes and edges. Dynamic networks are ubiquitous in the age of information technology, e.g., Twitter mentions network changes with each Tweet post or product purchasing network changes for every product purchased, or the friendship of Facebook (or professional connection in LinkedIn) network changes over time with the addition of new friends (or contacts). In such cases, fast recalculation of the PageRank plays a vital role in providing real-time services to its users. A node, if added to a network, brings several edges along with it. This disrupts the existing PageRank scores. Furthermore, the score of the newly added node needs to be determined. The trivial solution is to recalculate the PageRank for the whole network. However, it is time-consuming and not sustainable when nodes are added to the network rapidly. In this connection, the research problem is to recalculate the PageRanks for the affected nodes only instead of the whole network. Can we recalculate fast without affecting the actual rank of the nodes?

PageRank, first proposed in [13], is conventionally used for ranking web pages. It is based on the idea of Eigenvector centrality. Since its inception in 1999 [13], many different variants of PageRank algorithms [9, 11, 4, 10, 14, 19, 16] are proposed. Many of these algorithms work only on static networks. While the deterministic algorithms are slow for large networks, many randomized Monte Carlo based approaches [4, 16, 14, 19, 2] are developed to provide fast solutions. Monte Carlo based algorithms provide a good estimate of the PageRanks. However, the modern network changes over time at a rapid speed. Many dynamic PageRank [7, 5, 10, 18, 8, 12] approaches are proposed to deal with the dynamic network. However, they cannot cope with the pace of the streaming graph, especially for Big Data Social Networks. A simple and efficient algorithm to find the PageRanks for topological changes in dynamic streaming graphs is the need of the hour.

In this paper, we propose a simple algorithm to update the PageRanks in a directed network after adding a new node in the network. The proposed algorithm, namely, Fast Pessimistic PageRank (FPPR), uses the expected value of a random surfer to calculate the score of the new node by adding the scores contributed by inlinks and estimating to what extent the chain may contribute to the score of the node. The same method is used to update existing nodes through the outlinks of the newly added node. The proposed algorithm executes in the $\mathcal{O}(k \times d_{avg})$ time complexity for k newly added nodes. Here d_{avg} is the average degree of the k nodes added to the graph. Further, FPPR takes only $\mathcal{O}(|V|)$ additional space, specifically, $4 \times |V|$ space in worst case. The experiment performed over several synthetic and real-world networks shows that the ranking of the proposed method is highly correlated with the benchmark Power Iteration based recalculation and better than Fast Incremental PageRank(FIPR) [18].

The paper is organized as: Section 2 provides a brief literature review, the proposed FPPR method and its rationale are presented in Section 3, experiments performed and corresponding results are reported in Section 4 and finally, Section 5 describes the conclusions of the research work.

2 Related Work

PageRank [13] defines the importance of a node based on the quality of the neighboring nodes and the degree of the node. It is traditionally used for the hyperlink network of WWW. In simple terms, it uses a random surfer model where the random surfer clicks out-links with probability $1 - \rho$ and terminates its walk to start a new walk at a random page with probability ρ.

There are many Monte Carlo based approaches [4, 16, 14, 19, 2] for approximating the PageRank. Monte Carlo methods majorly follow four different strategies. These are (i) Monte Carlo end-point with a random start, (ii) Monte Carlo end-point with cyclic start, (iii) Monte Carlo complete path, and (iv) Monte Carlo complete path stopping at the dangling nodes. Out of these methods, the last one shows better performance in terms of execution time compared to the former three approaches [16]. In this method, R number of random walks are simulated starting from each node, and the random walk terminates at dangling nodes. The final rank vector is defined as $\pi = \frac{\text{[visits of each node]}}{\text{total visits}}$.

Many algorithms for the dynamic network were proposed in the literature. These are broadly classified into two categories viz, (i) aggregation algorithms [7, 5, 10] and (ii) Monte Carlo based algorithms [18, 12]. Although aggregate methods do not recompute the PageRanks of the whole graph, it aggregates the vicinity of the disturbed area imposed by the new changes in the graph. Thus, these methods recompute the PageRanks locally. The disadvantage of these methods is the aggregation overhead. In addition, the accuracy highly depends on the size of the subset selected for the aggregation or the area of the vicinity [3].

On the other hand, Monte Carlo based approaches use the theory of Markov chains [8]. In order to recalculate PageRanks on network changes, all the random walks which pass through the updated edges (or nodes) are required to be adjusted [18, 12]. Hence, it requires keeping track of all the random walks it performed earlier, and the same reason adds the overhead of additional space. For example, in FIPR [18], at each node, the unique segment IDs pass through the node, and the total number of visits to that node is stored. In addition, a hashtable is kept in each node, with unique segment IDs and the nodes that each segment covers whenever the graph is updated, visited, and the segments are modified in particular to that updated node or edge. Hence, these methods are not suitable for large-scale streaming graphs due to the significant memory overhead.

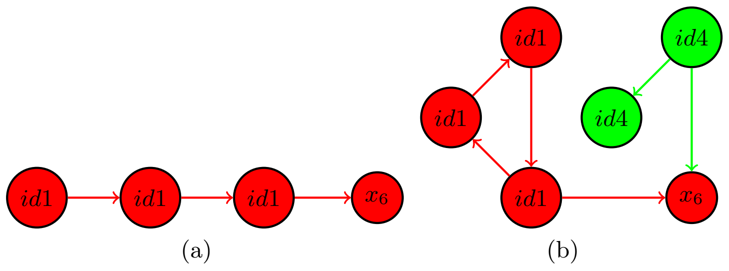


Fig. 1. Examples

3 Proposed Fast Pessimistic PageRank (FPPR) Algorithm for Dynamic Directed Graphs

The real-time use cases like finding the top-k popular products in an online shopping cart or finding top-k spreaders of news considering reshare network etc. will need high-speed computation of PageRank. In this section, we present the FPPR algorithm, which is capable of recalculating the PageRanks of a directed network upon the addition of new nodes. In order to achieve this, FPPR calculates the expected score of random surfers considering the static graph, i.e., if the Monte Carlo method is executed on the graph after the addition of the node, what would be the expected score generated by the random walks. For example, in

Fig. 1(a), let 'x6' be a new node. Intuitively, the expected value of 80% (considering $\rho = 0.2$) of random walks through outlink reaching 'x6' must have visited the new node 'x6' considering a static Monte Carlo simulation. This is what we would like to calculate in the first phase (Approximate Visits, Definition 2). We assume the incoming link is linear in the second phase (Link Sensitivity Index, Definition 1), as shown in Fig. 1(a). For example, R random walks starting from the node of 'id1' must have reached 'x6' in 3 hops, i.e., the expected number of visits is $R * (0.8)^3$. Similarly, the next node must have gone 'x6' in 2 hops (the expected visit is $R * (0.8)^2$), and so on. All these contribute to the score of the new node. Note that the same formulation may not be valid for Fig. 1(b) 'id4'. In such a case, the algorithm is prone to error. However, the algorithm thinks pessimistic that "the 'id4' has a loop". Then all random walks made on those nodes must have reached the node 'x6', and our linear assumption is valid. Accordingly, we are calculating the PageRank scores. A similar kind of process is adopted in addressing the outlink from the new node as well.

Definition 1 (Link Sensitivity Index (LSI)). *LSI defines the extent one node affects the whole link chain it belongs. We assume the link chain is linear and the LSI is mathematically defined by:*

$$c + c(c)^1 + c(c)^2 + c(c)^3 + ... \rightarrow \frac{c(1-(c)^n)}{(1-c)} \tag{1}$$

Here, $c = 1 - \rho$ is the probability that a random surfer moves forward and n is the length of the chain. In our experiment we took $c = 0.8$.

Definition 2 (Approximate Visits (AV)). *AV estimates the contribution of the incoming links to the newly added node that a random walk might have used if the random walk is executed from the scratch. If nodes have bidirectional edges, a max of approximate visits is considered for both nodes. It is calculated as:*

$$AV(v) \leftarrow \sum_{u \in \Gamma_{in}(v)} \frac{1}{d_{out}(u)} * (1-\rho) * (AV(u)) \tag{2}$$

Here, $\Gamma_{in}(.)$ and $d_{out}(.)$ return the set of incoming neighbors and out degree respectively.

FPPR does not keep track of all the random walk segments or aggregations. FPPR takes only $4 \times |V|$ space. That is, space complexity is $\mathcal{O}(|V|)$. For a network with billions of nodes, this provides a significant improvement.

The Algorithm 1 shows the steps of FPPR algorithm. The *linkID* of a node is the ID given to a node to identify the chain to which it belongs. In other words, *linkID* keeps track of all links or distinct chains in the graph. One may note that a lower *linkID* is assigned to a node if multiple links exist through that node. This convention will help resolving the conflicts. As we are keeping track of every link(chain) in the graph, we store the length of each *linkID*.

Algorithm 1 Calculate the approximate PageRank of the newly created node

```
Input: new node $u$, $R = 1000$
for all $v \in \Gamma_{in}(u)$ do
    Assign linkID to the node as described in text
    $linkID[v].length \leftarrow linkID[v].length + 1$
    $temporary\_Var \leftarrow R \times LSI(n = linkID[v].length)$ {LSI calculated by Eqn. 1}
    $linkSensitivityIndex \leftarrow \max(linkSensitivityIndex, temporary\_Var)$
    $approxVisits \leftarrow AV(v)$ {Using Eqn. 2}
end for
$approxVisits \leftarrow approxVisits + R + linkSensitivityIndex$
$finalResult[u] \leftarrow approxVisits$, $totalVisits \leftarrow totalVisits + approxVisits$
$approxVisits, linkSensitivityIndex \leftarrow 0$
for all $v \in \Gamma_{out}(u)$ do
    if $u$ has bidirectional edge with $v$ then
        replace both nodes with maximum visits
    else
        Assign link ID to the node as described in text
        $linkID[u].length \leftarrow linkID[u].length + 1$
        $temporary\_Var \leftarrow R \times LSI(n = linkID[u].length)$ {LSI calculated by Eqn. 1}
        $approxVisits \leftarrow AV(u)$ {Using Eqn. 2}
        $finalResult[v] \leftarrow approxVisits + linkSensitivityIndex$
        $totalVisits \leftarrow totalViists + approxVisits + linkSensitivityIndex$
    end if
end for
return $finalResult \div totalVisits$
```

4 Experiments and Results

Experiments have been conducted on synthetically generated networks as well as real-world networks. Both the Random Network and Barabasi-Albert [1] networks are used. The real world networks are Wiebo reshare network of [6]. The salient features of these graphs are presented in Table 1. Barabasi-Albert networks support growth but as the Erdos-Renyi random graph do not support growth, we use Algorithm 2 to generate the network. In order to get the dynamic character, we started with one node and added each node according to their creation for Barabasi-Albert and Random networks, on the other hand for real-world network, the timestamp is used.

Algorithm 2 Random graph generator

```
Input: number of nodes $N$ {single node '0' exist in the graph}
for all $x \in \Gamma_{in}(1, N)$ do
    add $x$ to Graph $G$
    $indegree \leftarrow random[0, 1]$
    if $indegree$ then
        $G.addEdge(random[0, x - 1], x)$
    end if
    $outdegree \leftarrow random[0, x - 1]$
    $shuffle.list[0, ..x - 1]$
    while $outdegree$ do
        $G.addEdge(x, list[outdegree])$
        $outdegree--$
    end while
end for
```

Table 1. Dataset Description

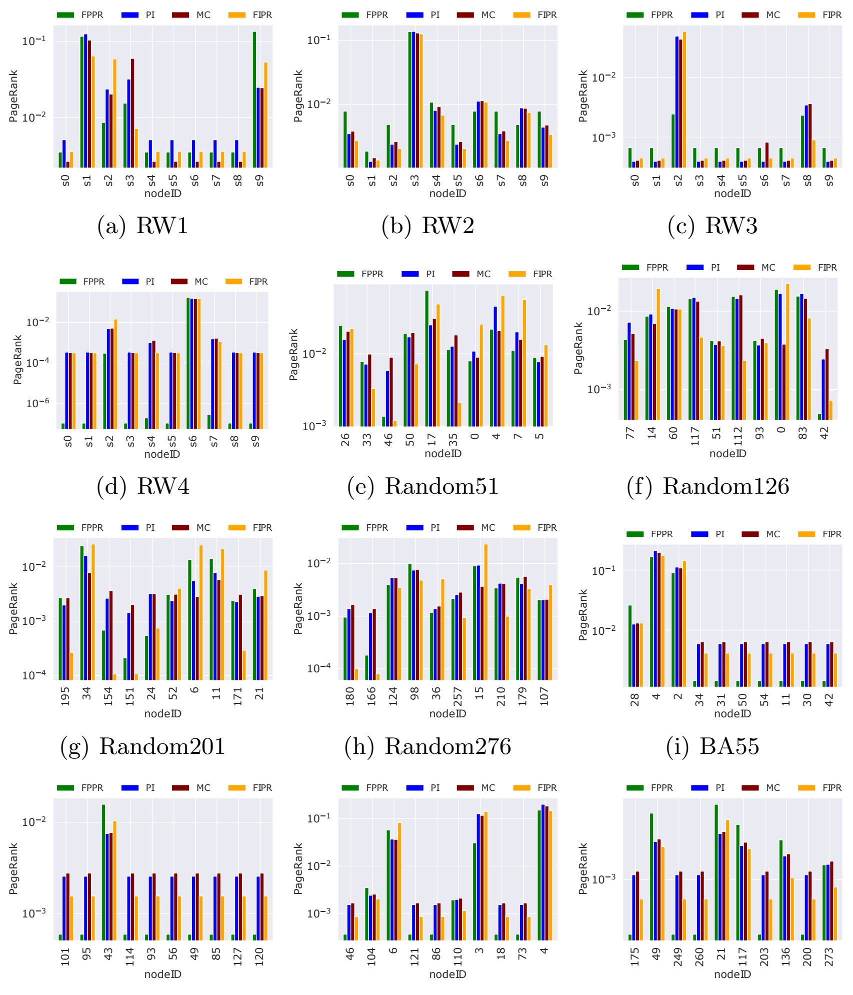
Name	Nodes	Edges	Min In-Deg	Max In-Deg	Min Out-Deg	Max Out-Deg
RW1 [6]	40	66	0	12	0	6
RW2 [6]	206	206	0	45	0	2
RW3 [6]	759	780	0	201	0	3
RW4 [6]	817	901	0	28	0	297
Random51	51	93	0	5	0	7
Random126	126	233	0	5	0	8
Random201	201	386	0	7	0	9
Random276	276	543	0	10	0	7
BA1 [1]	55	154	0	35	0	3
BA2 [1]	130	376	0	61	0	3
BA3 [1]	205	604	0	101	0	3
BA4 [1]	280	829	0	135	0	3

Comparing Methods The proposed methods have been compared with the following methods.

- **PowerIteration (PI)** [13]: We consider this method as a benchmark in our experiments. In the experiment, we restarted PI algorithm and recalculated PageRank for the whole graph on each node addition.
- **Static Monte Carlo (MC)** [16]: Static Monte-Carlo method is the method of approximate PageRanks of the network using Monte Carlo method. We implemented the version of the complete path with dangling nodes. Similar to PI, we recalculate the PageRanks every time a node is added to the network. The number of random walk is considered in the experiment is 1000.
- **Fast Incremental PageRank on Dynamic Networks (FIPR)** [18]: The method is designed for dynamic networks and proposed in 2019. As our algorithm is designed for dynamic networks, we included this method as a related research. Parameter R is set to 16 in the experiments.

4.1 Results

Accuracy All comparing algorithms were executed on different graphs. As expected, the absolute values of PageRanks by different algorithms of a node are different. The results for 10 random nodes for all the datasets are shown in Fig. 2 for reference. Hence, comparing different algorithms in terms of absolute values of the PageRank is not fare. Therefore the Spearman's rank correlation coefficient [15] is used to compare the ranking of the proposed FPPR with that of the comparing methods. The Spearman correlation between two vectors will be high when observations have a similar rank between the two variables and it will be lower otherwise. A value of it in between 0.8 to 1 is considered to be strongly correlated [17]. The results of Spearman's ranking coefficient for all nodes in the network are shown in the Fig. 4. It is evident from the result that the proposed FPPR performed equal or better than FIPR for all the network except RW2 while comparing with the PI and MC as benchmarks. Note that MC and PI are highly correlated with each other as both of these are recalculated over the full graph once a new node is added to the graph.



(a) RW1 (b) RW2 (c) RW3

(d) RW4 (e) Random51 (f) Random126

(g) Random201 (h) Random276 (i) BA55

(j) BA130 (k) BA205 (l) BA280

Fig. 2. PageRank for Randomly Sampled 10 nodes of different algorithms.

Spearman's rank correlation with changes in network: As part of experiment, we would like to see how the Spearman's correlation coefficient changes with addition of nodes. We recalculate Spearman's correlation coefficient of the proposed algorithm against the benchmark Power Iteration for the addition of each 10 nodes. The result is plotted in Fig. 3 for 6 data sets. The value dipped around 26% for both random networks, while the Barabasi-Albert network shows consistent improvement in the value of Spearman's correlation coefficient. For both real-world networks, the values are constant because, in general, real-world retweet networks have few influential nodes (hubs) responsible for the tweet's diffusion resulting in high page rank scores throughout the evolution of the graph.

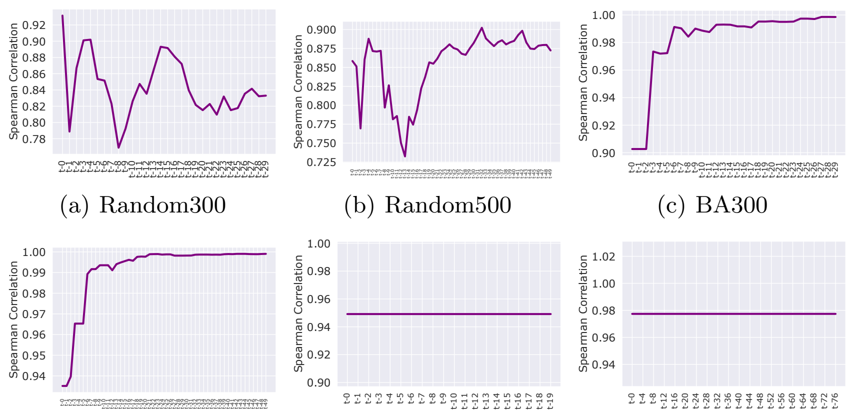


Fig. 3. Spearman Correlation of FPPR VS PI over time. Each time tick denotes addition of 10 nodes in the network.

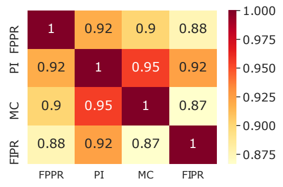
Execution Time: We checked the overall execution time of different comparing algorithms and found that the proposed algorithm is magnitude faster than all the methods we have experimented with for all data sets. The comparison is presented in the Fig. 5. As expected, FIPR takes second lowest time as it is designed for dynamic network and only updates the PageRank locally.

Table 2. Computation complexities of different PageRank algorithms, k is number of updates, n is number of nodes, E is edgeset, R is the no of random walk simulations.

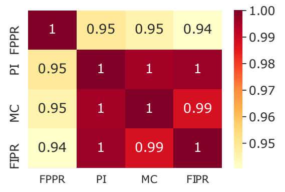
Algorithm	Time complexity
Power Iteration	$\Omega\frac{(kn^2)}{1(1-\rho)}$
Monte Carlo method	$\Omega\frac{(knR)}{(\rho)}$
FIPR	$O\frac{(kn)}{(\lvert E\rvert)}$
Our Method	$\mathcal{O}(k \times d_{avg})$

4.2 Computation Complexity

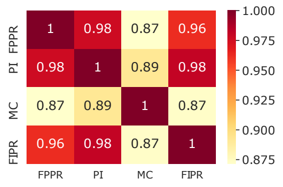
Computation complexities of different algorithms are shown in Table 2. Thus, the worth case complexity of the proposed algorithm is $\mathcal{O}(k \times d_{avg})$, where k is the number of node added to the network.



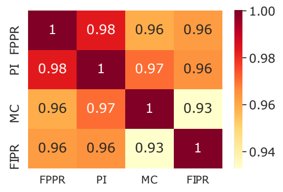
(a) RW1



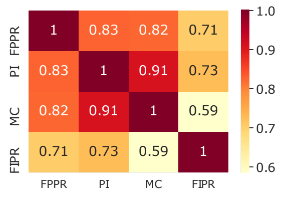
(b) RW2



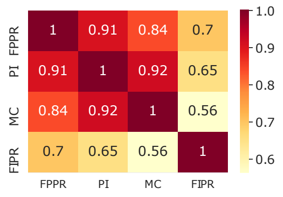
(c) RW3



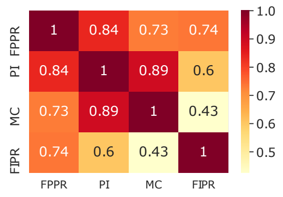
(d) RW4



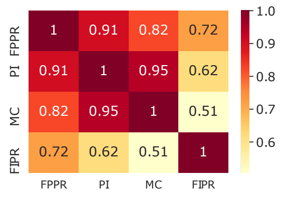
(e) Random51



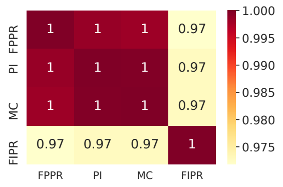
(f) Random126



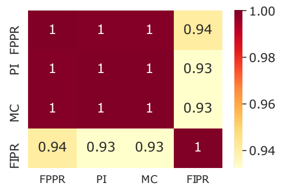
(g) Random201



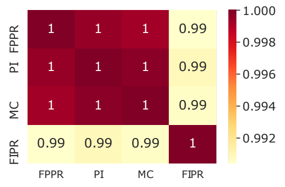
(h) Random276



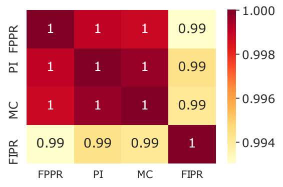
(i) BA55



(j) BA130



(k) BA205



(l) BA280

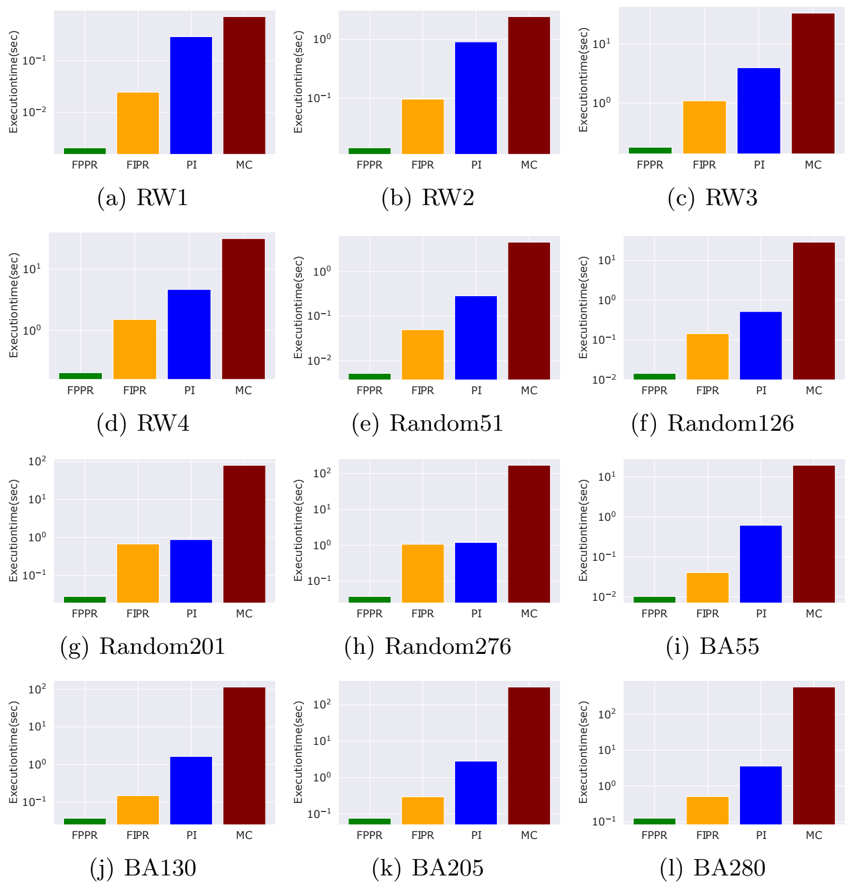
Fig. 4. Spearman Correlation between all four approaches over the different data sets

5 Conclusion

In the present paper, we proposed a new algorithm for calculating PageRank for dynamic directed graph. We consider the node addition for the same. We showed thorough experimental results that the proposed algorithm perform better in terms of ranking from the existing FIPR algorithm. The execution time is much faster while providing better results. Power Iteration algorithm is considered as benchmark.

Experiments were performed on wide range of networks including random networks, Barabasi-Albert network, and real world network. It is also shown that the PageRank value differs between algorithms while the ranking shows high correlation in terms of Spearman ranking correlation index.

Although the algorithm is executed for node addition, we believe this algorithm can be modified and applied to the link addition as well. Similarly, node and edge deletions may also be employed. These are kept as the future works in the current research.



(a) RW1 (b) RW2 (c) RW3 (d) RW4 (e) Random51 (f) Random126 (g) Random201 (h) Random276 (i) BA55 (j) BA130 (k) BA205 (l) BA280

Fig. 5. Comparing plots of execution time of different algorithms

References

1. Réka Albert and Albert-László Barabási. Statistical Mechanics of Complex Networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.
2. K Avrachenkov, N Litvak, D Nemirovsky, and N Osipova. Monte Carlo methods in pagerank computation: When one iteration is sufficient. *SIAM Journal on Numerical Analysis*, 45(2):890–904, 2007.
3. Bahman Bahmani, Abdur Chowdhury, and Ashish Goel. Fast incremental and personalized PageRank. *Proceedings of the VLDB Endowment*, 4(3):173–184, 2010.
4. La Breyer. Markovian page ranking distributions: some theory and simulations. Technical report, 2002.
5. Steve Chien, Cynthia Dwork, Ravi Kumar, Daniel R Simon, and D Sivakumar. Link Evolution: Analysis and Algorithms. *Internet Mathematics*, 1:277–304, 2004.
6. Yuwei Chuai and Jichang Zhao. Anger makes fake news viral online, 2020.
7. Prasanna Desikan, Nishith Pathak, Jaideep Srivastava, and Vipin Kumar. Incremental page rank computation on evolving graphs. In *14th International World Wide Web Conference, WWW2005*, pages 1094–1095, 2005.
8. Valerie Isham and E. Seneta. Non-Negative Matrices and Markov Chains. *Journal of the Royal Statistical Society. Series A (General)*, 146(2):202, 1983.
9. Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
10. Amy N Langville and Carl D Meyer. Updating PageRank with iterative aggregation. In *Proc. of Thirteenth International World Wide Web Conference*, pages 1124–1125, New York, 2004.
11. R Lempel and S Moran. Stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Computer Networks*, 33(1):387–401, 2000.
12. Qun Liao, ShuangShuang Jiang, Min Yu, Yulu Yang, and Tao Li. Monte carlo based incremental pagerank on evolving graphs. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon, editors, *Advances in Knowledge Discovery and Data Mining*, pages 356–367, Cham, 2017. Springer International Publishing.
13. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. *World Wide Web Internet And Web Information Systems*, 54(1999-66):1–17, 1998.
14. Ozlem Salehi. *PageRank algorithm and Monte Carlo methods in PageRank Computation.* PhD thesis, Bogazici University, 2007.
15. C Spearman. The Proof and Measurement of Association between Two Things. *The American Journal of Psychology*, 15(1):72, 1904.
16. Brian Vargas. *Exploring PageRank Algorithms : Power Iteration & Monte Carlo Methods.* PhD thesis, California State University, San Marcos, 2020.
17. Jerrold H Zar. Spearman Rank Correlation. In *Encyclopedia of Biostatistics.* John Wiley & Sons, Ltd, 2005.
18. Zexing Zhan, Ruimin Hu, Xiyue Gao, and Nian Huai. Fast incremental pagerank on dynamic networks. In Maxim Bakaev, , Flavius Frasincar, , and In-Young Ko, editors, *Proc. of International Conference on Web Engineering*, volume 11496 LNCS, pages 154–168, Cham, 2019. Springer International Publishing.
19. Zhi Zhou. *Evaluation of Monte Carlo Method in PageRank.* PhD thesis, University of Missouri-Columbia, 2015.