INFORMATION RETRIEVAL AND WEB SEARCH

Project Report

Vinci Luca 868166

A.Y. 2022/2023

Summary

1. Introduction

1.1 PageRank

PageRank is a link analysis algorithm, named after Brin&Page **[1]**, and used by the Google Internet search engine, which assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative importance within the set.

PageRank relies on the **democratic** nature of the Web. Uses the vast link structure as an indicator of an individual page's value or quality. PageRank interprets a hyperlink from page x to page x as a **vote**, by page x, for page y.

However, PageRank looks at more than the sheer number of votes; it also analyzes the page that casts the vote. Votes casted by “important” pages weigh more heavily and help to make other pages more "**important**". This is the idea of **rank prestige** in social networks.

The Web is seen as a directed graph $G = (V, E)$. The PageRank score of page i, denoted by $P(i)$, is defined by:

$$P(i) = \sum_{(j,i)\in E} \frac{P(j)}{O_j}$$

O_j is the number of out-link of j. We can also see it as a matrix notation. Let $n = |V|$ be the total number of pages. We have a system of n linear equations with n unknowns. We can use a matrix to represent them. Let P be an **n-dimensional** column vector of PageRank values, i.e., $P = (P(1), P(2), ..., P(n))^T$.

Let A be the adjacency matrix of our graph with:

$$A_{ij} = \begin{cases} \frac{1}{O_i} & if\,(i,j) \in E \\ 0 & otherwise \end{cases}$$

We can write the n equations with:

$$P = A^T P$$

PageRank is computed iteratively; while at the very beginning (time t) all n pages have the same score $1/n$, as time passes (time $t + 1$) the $i - th$ page has instead the following PageRank value:

$$p_{i,t+1} = \frac{1-d}{n} + d \sum_{j=1}^{n} A_{ij} P_{j,t}$$

d represents a **damping factor** usually set to a value between 0 and 1 (the implementation uses $d = 0.85$).

In the previous formula the element A_{ij} represents the transition probability that the user in page i will navigate to page j, and is computed as:

- $1/O_i$ if i not a dangling node
- $1/n$ if i is a dangling node
- 0 if there is no edge between i and j

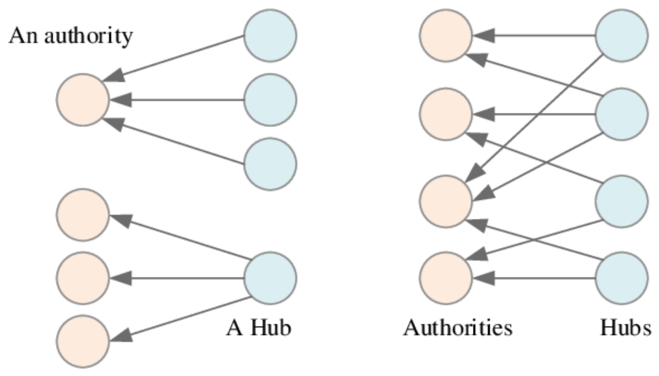
Dangling nodes are nodes with no outgoing links.

1.2 HITS

HITS, short for **Hypertext Induced Topic Search**, differs from PageRank as it operates as a dynamic ranking algorithm that depends on the user's search query. When a user initiates a search query, HITS begins by broadening the selection of pertinent pages provided by a search engine. Subsequently, it generates two distinct rankings for the extended collection of pages: an **authority ranking** and a **hub ranking**.

An **authority** is a page with many in-links. The idea is that the page may have authoritative content on some topic and thus many people trust it and thus link to it. A **hub** is a page with many out-links.
In the provided image, there is an illustration of both an authority page and a hub page.

The fundamental concept underlying HITS is that an effective hub page directs users to numerous high-quality authority pages, while a reputable authority page is linked to by numerous exceptional hub pages. This dynamic interaction results in a mutually reinforcing relationship between authorities and hubs.



[2]The basic concept of the HITS model

Given a broad search query, q , HITS collects a set of pages as follows:

- It sends query q to a search engine.
- It then collects the t ($t = 200$ is used in the HITS paper) highest ranked pages. This set is called the root set W.
- It then grows W by including any page pointed to by a page in W and any page that points to a page in W.
- This generates a larger set S, the base set.

HITS works on the pages in S, and assigns every page in S an **authority score** and a **hub score**.

Given S, $|S| = n$, we again use $G = (V, E)$ to denote the hyperlink graph of S. We use L to denote the adjacency matrix of the graph.

$$L_{ij} = \begin{cases} 1 & if\,(i, j) \in E \\ 0 & otherwise \end{cases}$$

Let the authority score of the page i be $a(i)$, and the hub score of page i be $h(i)$. The mutual reinforcing relationship of the two scores is represented as follows:

$$a(i) = \sum_{(j,i)\in E} h(j)$$

The authority score is given by the sum of the hub scores of the incoming links, and the hub score is given by the sum of the authority scores of the outgoing links.

$$h(i) = \sum_{(j,i)\in E} a(j)$$

Let a denote the column vector with all the authority scores: $a = (a(1), a(2), ..., a(n))^T$
Let h denote the column vector with all the authority scores: $h = (h(1), h(2), ..., h(n))^T$
Then:

$$a = L^T h$$
$$h = L\, a$$

The **computation** of authority scores and hub scores is the same as the computation of the PageRank scores, using **power iteration**. If we use a_k and h_k to denote authority and hub vectors at the k^{th} iteration, the iterations for generating the final solutions are:

$$a_k = L^T L\, a_{k-1}$$
$$h_k = L\, L^T\, h_{k-1}$$

starting with $a_0 = h_0 = (1, 1,... 1)$.
After each iteration, the values are also normalized (to keep them small). The iteration concludes when the **1-norms** of the residual vectors fall below predefined thresholds ε.

Pages exhibiting significant authority and hub scores are considered superior in terms of their authority and hub qualities. HITS identifies a limited number of top-ranked pages as authorities and hubs, which are then presented to the user.

While HITS consistently converges, it faces a challenge related to the uniqueness of the ultimate, converged authority and hub vectors.

1.3 InDegree

In-Degree is a metric employed to gauge the level of prominence a node holds within a graph. Similar to the preceding two metrics, it is attributed to each node and quantified by the count of inbound links directed toward that specific node. This can be expressed mathematically using the following formula:

$$P_d(i) = \frac{d_I(i)}{n-1}$$

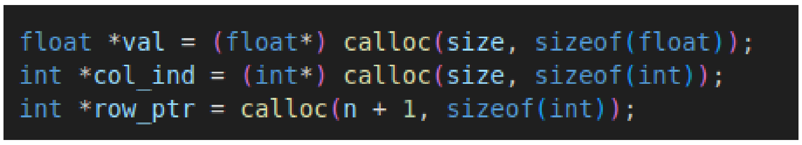
Where $d_I(i)$ represents the number of incoming links to node i and n represents the number of nodes. Notice that the denominator $n - 1$ is used to normalize the quantity $d_I(i)$, so that the final prestige of node i ranges from 0 to 1.

2. Implementation

2.1 PageRank Implementation

In the implementation the transposed adjacency matrix is not used, we are going to store the matrix in a sparse compressed form called compressed sparse row (CSR). This is because as the matrix grows, its sparseness also grows, and therefore the memory needed to store the matrix also increases.

We use, as per the course slides, the following vectors for the implementation of CSR:



```
float *val = (float*) calloc(size, sizeof(float));
int *col_ind = (int*) calloc(size, sizeof(int));
int *row_ptr = calloc(n + 1, sizeof(int));
```

Where:

- **`val`**: stores the values of the matrix
- **`col_ind`**: stores for each value which column it is in in the matrix
- **`row_ptr`**: respectively stores where a row of the matrix begins

Let's scroll through the dataset file line by line in order to build the CSR. To obtain a matrix that is stochastic, irreducible and aperiodic at the moment of calculation we consider the entry of dangling nodes with $1/n$ and add a link from each page to every page, and give each link a small transition probability controlled by a parameter d, $0 < d < 1$.

The algorithm we use to implement PageRank is the **Power Iteration Method**. The termination condition is defined by the case in which two consecutive iterations of the algorithm produce two almost identical p-vectors. At the end of the algorithm we obtain the final PageRank values.

2.2 HITS Implementation

The initial step in implementing the HITS algorithm involves computing the **authority** and **hub** scores using the adjacency matrix and its transpose. Therefore, the computation of these two matrices marks the primary operation to be carried out.

The same structures are used as for implementing PageRank, please refer to the source code.

The process of calculating the authority and hub scores begins with the initialization of scores at time $k + 1$, and it proceeds by iteratively computing $h_{k+1} = L\, a_k$ and $a_{k+1} = L^T h_k$ as it traverses through the edges. This iterative process concludes when we achieve convergence for both scores based on the specified criteria.

2.3 InDegree Implementation

For the implementation of InDegree we simply take the transposed matrix in order to recover the number of incoming edges for each node.

3. Results

The tests are carried out on a machine with the following characteristics:

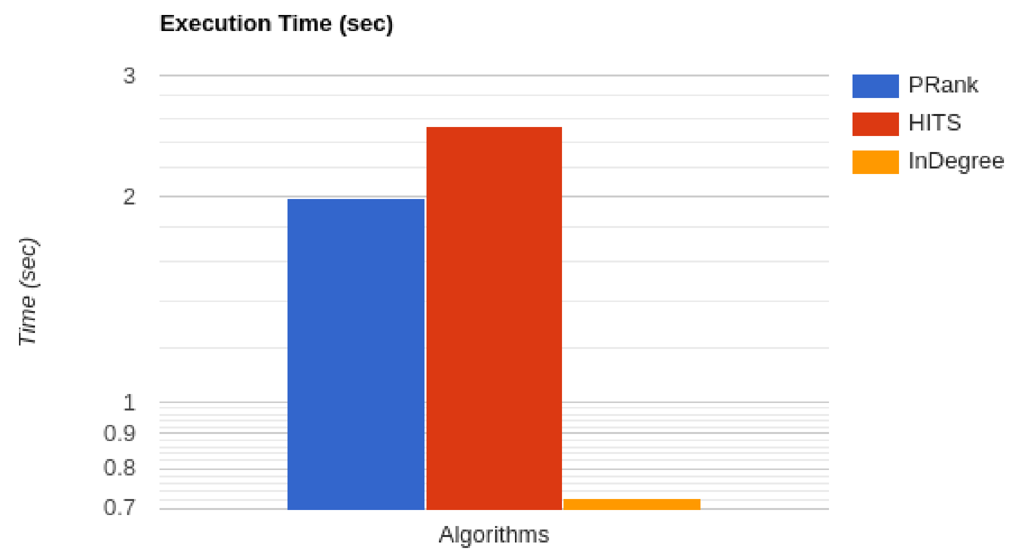
- **CPU**: 11th Gen Intel® Core™ i5-1135G7 @ 2.40GHz × 8
- **OS**: Ubuntu 22.04.3 LTS

The tests are also carried out on the following data sets:

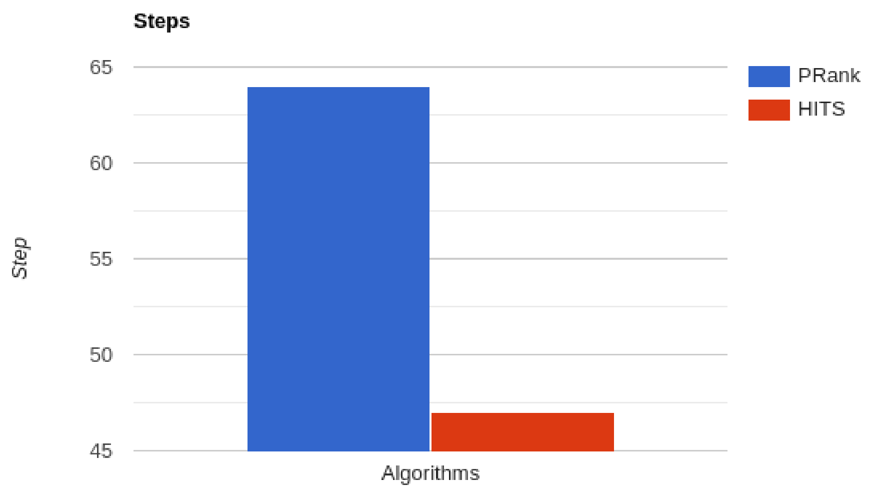
- **web-BerkStan** with **685’230 nodes** and **7’600’595 edges**
- **web-NotreDame** with **325’729 nodes** and **1’497’134 edges**
- **web-Stanford** with **281’903 nodes** and **2’312’497 edges**

3.1 BerkStan Dataset

Let's see how the three algorithms perform compared on the **web-BerkStan** dataset, let's initially compare the execution time and then the number of steps performed for PageRank and HITS.

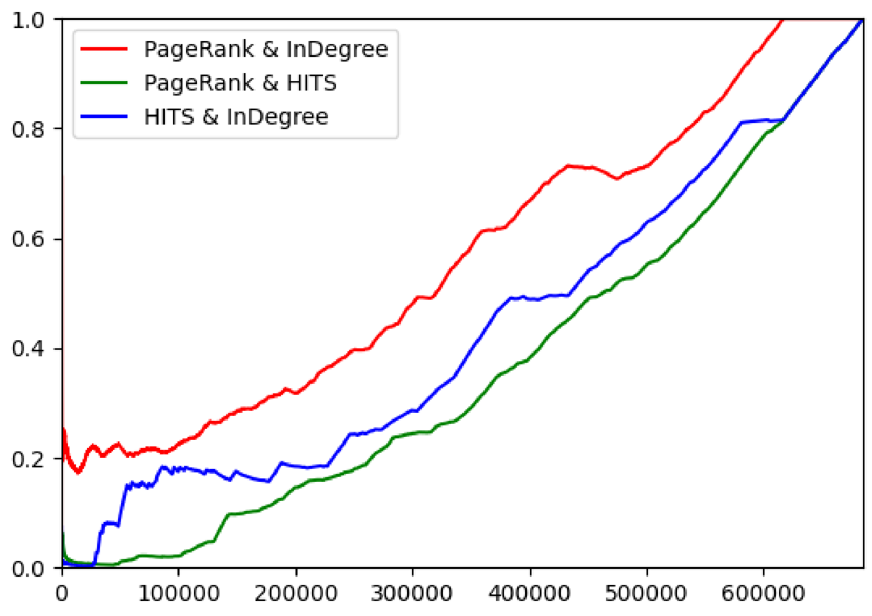


As evident from these results, PageRank outperforms HITS in terms of elapsed time in this particular case. In Degree proves to be the fastest algorithm, and this outcome is attributed to the simplicity of In Degree computation, which involves only a single analysis of the sequence of edges.



On the contrary, however, we can see that PageRank takes more steps to reach convergence despite taking less time than HITS, this fundamentally depends on the optimizations adopted for PageRank which make the algorithm more efficient in terms of time and memory.

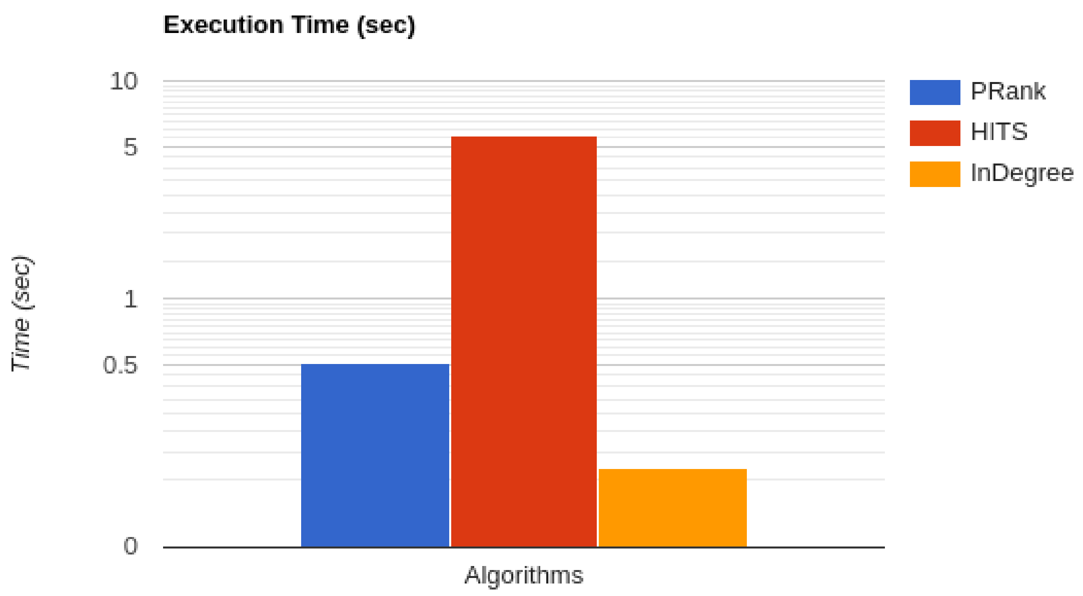
Below we can see the graph that compares each pair of ranking algorithms on the results obtained on the dataset. For each pair, the Jaccard coefficient was calculated for the top-k results, in order to analyze their similarity.



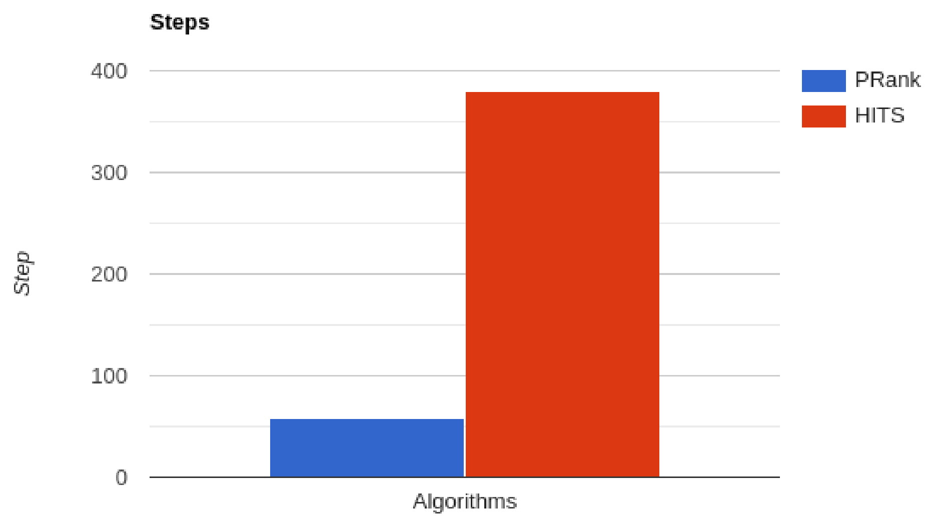
For each pair we can see an approximately linear growth as k increases, we can see how the similarity between PageRank and InDegree for this dataset is higher than PageRank and HITS.

3.2 Notre-Dame Dataset

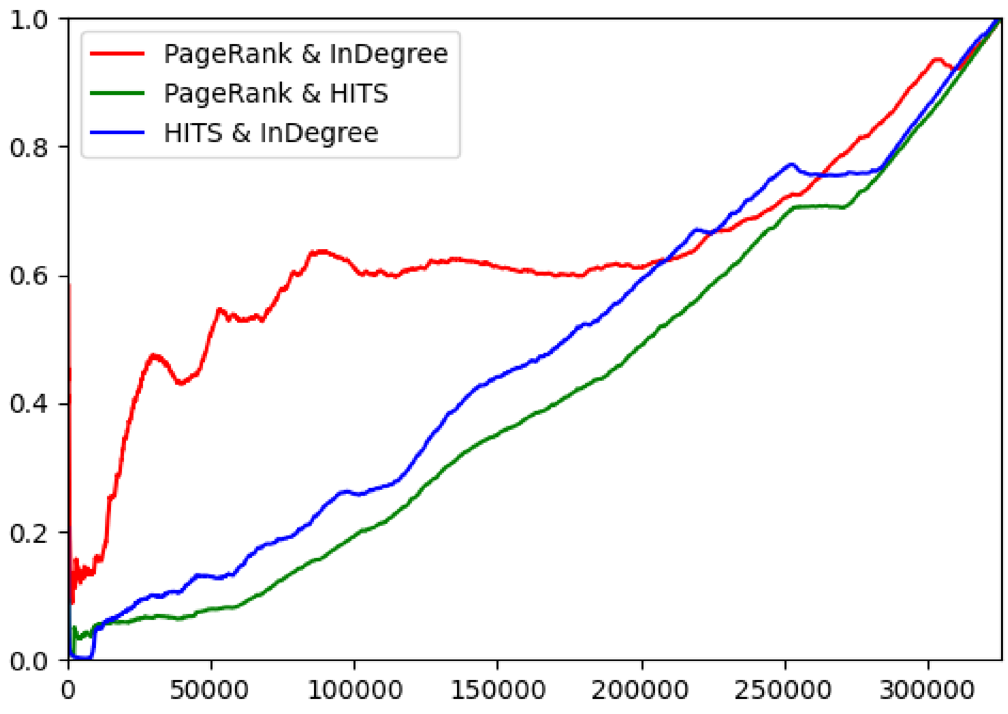
Let's see how the three algorithms perform compared on the **web-Notre-Dame** dataset, let's initially compare the execution time and then the number of steps performed for PageRank and HITS.



Also for this dataset we can see that PageRank performs better than HITS, and yet InDegree is the fastest as it only operates once on the set of arcs.



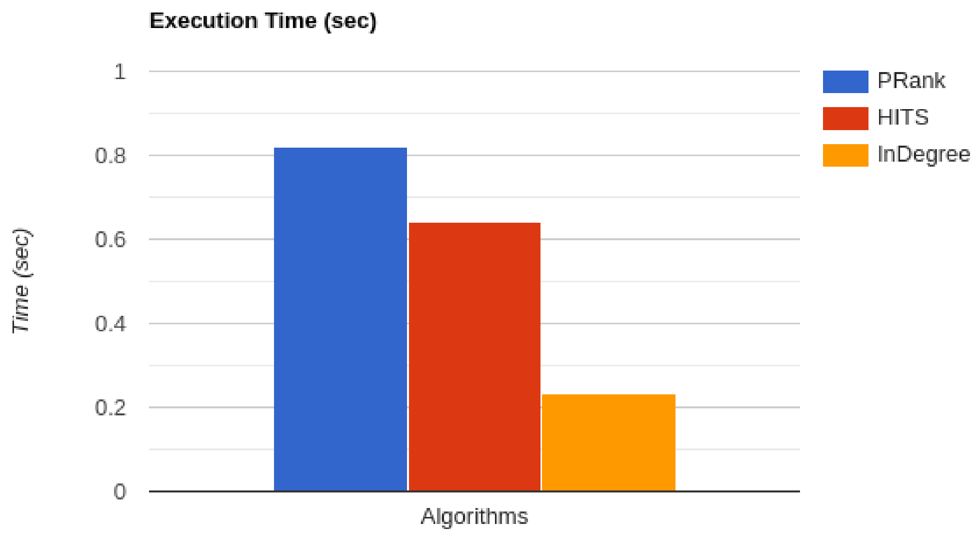
PageRank performed better than HIST both in terms of time spent and number of steps to converge.



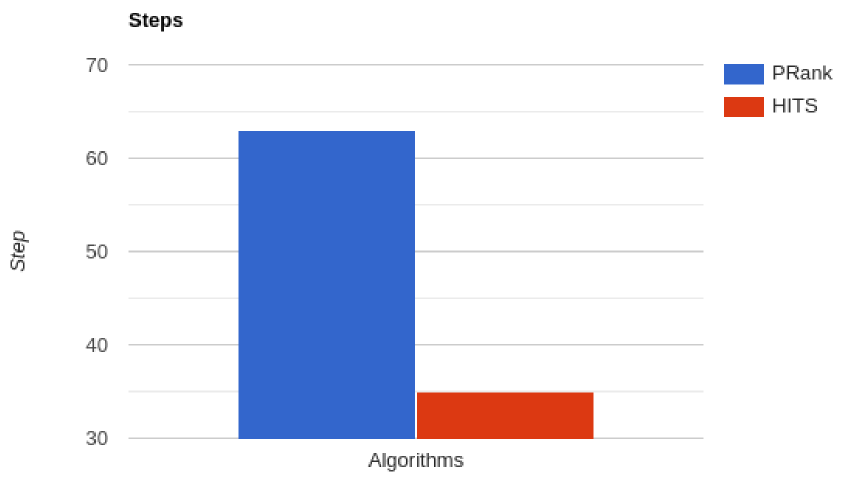
We can see how PageRank and InDegree reach a high similarity indicator around **top-100000**.
We see approximately linear growth for PageRank and HITS and for HITS and InDegree. We can still see how PageRank and HITS have less similarity than PageRank and InDegree.

3.3 Stanford Dataset

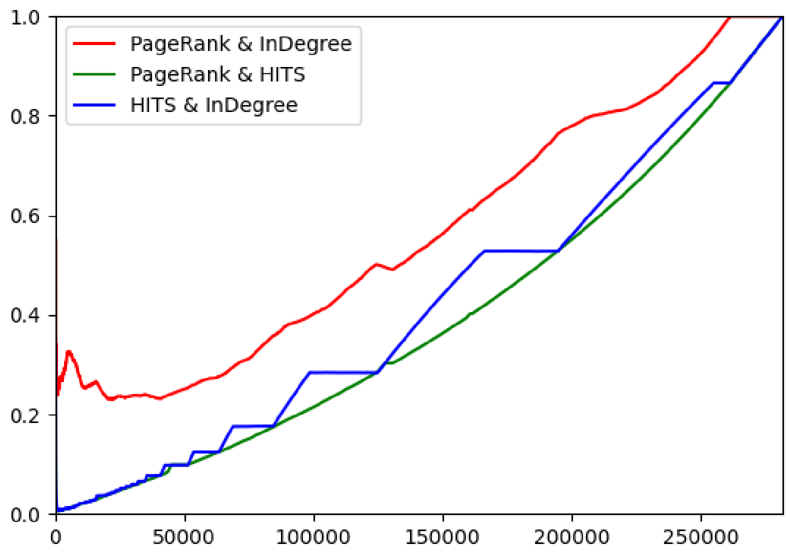
Let's see how the three algorithms perform compared on the **web-Stanford dataset**, let's initially compare the execution time and then the number of steps performed for PageRank and HITS.



As we can see in this case PageRank performs slightly worse than HITS in terms of execution time.



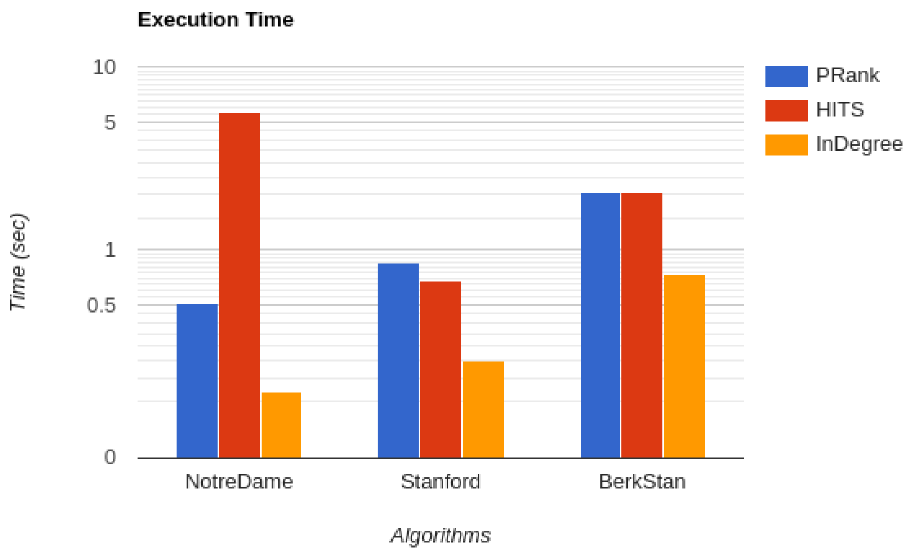
As we can see in this case PageRank converges after a greater number of steps than HITS.



Even with this dataset we see that PageRank and InDegree have a higher similarity than the other pairs, reaching a similarity of approximately 1 at top-250000.

3.4 Execution Time

Let's compare all the execution times of the various algorithms on the various datasets.



It can be seen how the execution time of PageRank increases in very direct proportion to the increase of the edges in the graph as well as for InDegree. While HITS we see that it performs worse on the NotreDame dataset, which is the dataset with fewer edges and nodes.

References

1. *Brin, S. and Page, L. (1998)* ***The Anatomy of a Large - Scale Hypertextual Web Search Engine****. In: Seventh International World-Wide Web Conference (WWW 1998), April 14-18, 1998, Brisbane, Australia.*
2. *https://www.researchgate.net/figure/The-basic-concept-of-the-HITS-model-used-in-our-method_fig1_334751984*