Kleinberg's HITS (Hubs and Authorities) Algorithm

Teodora Dobos
Technical University of Munich
Department of Informatics
Garching, Germany
teodora.dobos@tum.de

I. Introduction

One of the key functions of a web search engine is ranking pages according to their relevance to a given query. To realize this, there exist multiple approaches, such as ranking based on the number of matched terms with the query string or, conversely, ignoring the page content in the ranking process. In this context, we might ask ourselves what characterizes a relevant page and how it can be identified in the entire web graph: must it contain the whole query string or is term matching not an essential criterion for determining the page's importance? Is there any structural relationship between the pages relevant to a query? Is popularity of a page a factor to consider when calculating its relevance?

In this paper, we analyze Kleinberg's HITS (Hubs and Authorities) Algorithm and try to answer the questions listed above. HITS stands for Hyperlink-Induced Topic Search and denotes a link analysis algorithm invented by Jon Kleinberg in the context of the IBM CLEVER project [12] in 1998.

The general idea of the algorithm is to analyze the network structure of a hyperlinked environment and thus to retrieve the most authoritative (i.e. relevant) pages for a query. In this sense, a link gives a concrete indication of the following type of judgement: if the creator of page p included a link to page q inside p, then he considers that q is relevant to the topic covered in p. Hence, we could say that the page p has conferred authority on q. Extending this idea, Kleiberg defines two types of pages: *hubs*, which are pages that point to other pages that contain relevant information to a specific query (in the previous example, p is a hub), and *authorities*, which, for the given query, are the most relevant pages. Thus, two scores are associated to a page: a hub score, denoting the quality as an "expert" of the page, and an authority score, reflecting the quality of the page with respect to its content. The HITS algorithm works following the principle of repeated improvement, which means that the same steps for updating the hub and authority scores are performed in multiple iterations, until satisfactory results are achieved.

Kleinberg's algorithm aims to identify the most important authorities at a global level. This means that the algorithm can deal with *broad-topic queries* (e.g. "Find information about European countries"), which are underspecified and hence affected by the so-called *Abundance Problem* ("The number of pages that could reasonably be returned as relevant is far too large for a human user to digest" [5]). HITS works on a focused subgraph of the WWW graph, proposing a new technique for representing and filtering large volumes of data (i.e. web pages).

The Hubs and Authorities algorithm can be seen as an application of spectral graph theory, given that the structure on which the whole algorithm concentrates is the web graph. Although it might not seem evident, there is a strong connection between calculating the hub and authority scores corresponding to the pages being considered in the query processing phase and the computation of the eigenvectors of a specific matrix associated to the web graph. In this sense, linear algebra is a useful tool for analyzing the HITS algorithm, proving its convergence and efficiently calculating its results.

The remainder of this paper is structured as follows: section Algorithm concentrates on the implementation details, addresses the issue of convergence and examines the strengths and weaknesses of the algorithm, section Improvements describes three approaches to enhance the performance of HITS and section Conclusion summarizes the most important points of this paper.

II. Algorithm

A. Subgraph Construction

The algorithm operates on an induced subgraph of the whole network which is created at query-time. In this section, we describe the steps required for computing the subgraph that will later serve as the basis for calculating the hub and authority scores.

We recall some fundamental notions of graph theory and apply them to the linked structure of the web. Let $G = (V, E)$ be a directed graph, where V represents the set of all web pages and a tuple $(p, q) \in E$ if page p has a link to page q. Given a subset of pages $W \subseteq V$, we say that $G[W]$ is an induced subgraph if its vertex set is W and its edge set consists of all the edges in E that have both endpoints in W. Further, we denote with $\Gamma_{out}(p)$ the collection of pages to which p points and with $\Gamma_{in}(p)$ the set of pages that contain references to p.

Let Q_σ be the set of all pages containing a given query string σ. The algorithm could perform its steps on this collection, but there are two major disadvantages of this possible approach. The first one refers to the computational effort induced by applying a non-trivial algorithm on a set that might have millions of elements. In this case, the algorithm may need hours until it calculates appropriate search results or it might not even terminate. The second drawback relates to the fact that usually most of the best authorities do not contain the query string and are not elements of Q_σ. Consequently, applying the next steps of the HITS algorithm on Q_σ would lead to a poor quality of the search results.

A more appealing idea is to perform the algorithm on a subset S_σ (from now on, we call it *base set*) that has three main characteristics [5]:

(a) S_σ is small.
(b) S_σ has numerous relevant pages.
(c) S_σ includes most (or many) of the strongest authorities.

In the following, we describe the required steps for computing the S_σ subset.

Step 1: top $t \approx 200$ highest-ranked pages for σ from a text-based search engine (e.g. AltaVista) are selected. We call the set containing these pages the *root set* and denote it with R_σ. The elements of the root set are selected based on relevance ranking for the given query (e.g. using TF*IDF). Because $R_\sigma \subset Q_\sigma$, there are often extremely few links between the pages in R_σ, thus the subgraph induced on this set is sparse. However, a strong authority $q \notin R_\sigma$ is likely referenced by a page $p \in R_\sigma$ (i.e. $q \in \Gamma_{out}(p)$). Nevertheless, we notice that properties (a) and (b) hold for R_σ, while the third one may not always be true. Therefore, further actions are needed to transform the subset R_σ, so that property (c) holds.

Step 2: R_σ is expanded by applying the algorithm BuildSubgraph which takes the root set and a natural number d as arguments. The result is the subgraph G_σ [3].

```
Algorithm BuildSubgraph
Data: $R_\sigma, d$
Result: $G_\sigma$
Set $S_\sigma = R_\sigma$
forall $p \in R_\sigma$ do
    Add all pages in $\Gamma_{out}(p)$ to $S_\sigma$
    if $|\Gamma_{in}(p)| \leq d$ then
        Add all pages in $\Gamma_{in}(p)$ to $S_\sigma$
    else
        Choose $\Gamma_d(p) \subseteq \Gamma_{in}(p)$ such that $|\Gamma_d(p)| = d$
        Add $\Gamma_d(p)$ to $S_\sigma$
```

We observe that all pages referenced by a page p that is in the root set R_σ are added to the base set S_σ, while this is not the case for the pages contained in $\Gamma_{in}(p)$. The reason for allowing at most d pages that point to a specific $p \in R_\sigma$ relates to property (a) listed before: each page from the *WWW* graph could be pointed to by several hundreds of pages, so adding all of them to the base set would increase its size considerably.

After performing steps 1 and 2 we obtain a subgraph (we call it G_σ) on which the algorithm will further operate. Figure 1 illustrates the root and the base sets.

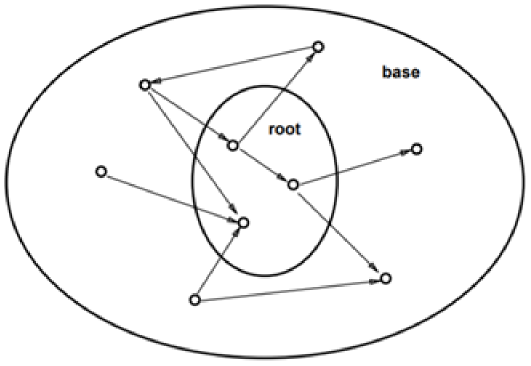

Figure 1. Root and base sets (Source: [5]).

B. Hubs and Authorities Computation

The next step of the algorithm focuses on calculating the hub and authority scores. We recall the subgraph G_σ (determined in the previous section), whose link structure will be now examined. The hub and authority scores are computed using an iterative procedure which will be described in the following. The number of steps required for obtaining reasonable search results is not fixed and the problem regarding the convergence of the algorithm will be discussed in a subsequent section.

A page is relevant for a given query if many other pages point to it. Therefore, an appealing idea would be to sort pages by their in-degree, so that at the end of the algorithm, the pages having the biggest such value will be ranked first. Although this approach might produce high-quality results, it brings one major drawback. For instance, Kleinberg states that, on the query "java", the pages having the highest in-degree are **"www.gamelan.com"** and **"java.sun.com"** (both related to Java programming language). Other pages that are ranked in the top results are websites advertising **Caribbean vacations** and the homepage of **Amazon books.** We notice the lack of thematic unity between these pages and the fact that there is no clear differentiation between the pages that are relevant to the query topic (i.e. the strong authorities) and the ones that are universally popular (i.e. with large in-degree, regardless of the underlying topic). We definitely need a different approach that solves this issue.

The following assumption motivates the solution proposed by the HITS Algorithm: it does not suffice for authoritative pages which are relevant to the query to have large in-degrees, there should also exist an overlap in the collection of pages that point to them [5]. In this context, it is reasonable to consider *hub pages* (and hub scores) that are referencing pages relevant to the searched topic.

Authority and hub values are defined in a *mutually reinforcing relationship* [1]. A good authority is pointed to by numerous hubs, while a good hub points to many good authorities. We now present a method used for the computation of hubs and authorities.

Two non-negative weights are assigned to each page included in the base set: the authority weight and the hub weight. For a given page $p \in S_\sigma$, let $a^{\langle p \rangle}$ be the authority weight and $h^{\langle p \rangle}$ be the hub weight. We set the invariant that for each type of weight, all square values corresponding to the pages in the set S_σ are normalized such that their squares sum to 1. Expressed mathematically: $\sum_{p \in S_\sigma} (a^{\langle p \rangle})^2 = 1$ and $\sum_{p \in S_\sigma} (h^{\langle p \rangle})^2 = 1$.

We define two operations on weights, namely $\mathscr{I}$ and $\mathscr{O}$ [5]. For given weights $\{a^{\langle p \rangle}\}$ and $\{h^{\langle p \rangle}\}$, the $\mathscr{I}$ operation updates the authority scores as follows:

$$a^{\langle p \rangle} = \sum\nolimits_{q:(q,p) \in E} h^{\langle q \rangle} \tag{1}$$

Similarly, the $\mathscr{O}$ operation updates the hub scores as follows:

$$h^{\langle p \rangle} = \sum\nolimits_{q:(p,q) \in E} a^{\langle q \rangle} \tag{2}$$

Note that E is the set of all edges (links) between the pages included in the focused subgraph G_σ on which our algorithm operates.

Equation (1) suggests that if p is referenced by pages with large hub values, then it receives a high authority score, while the equation (2) says that if p points to pages with high authority values, then it gets a large hub score. The mutual recursion between hubs and authorities is preserved accordingly. Figure 2 illustrates the computation of authority and hub scores based on two small graphs.

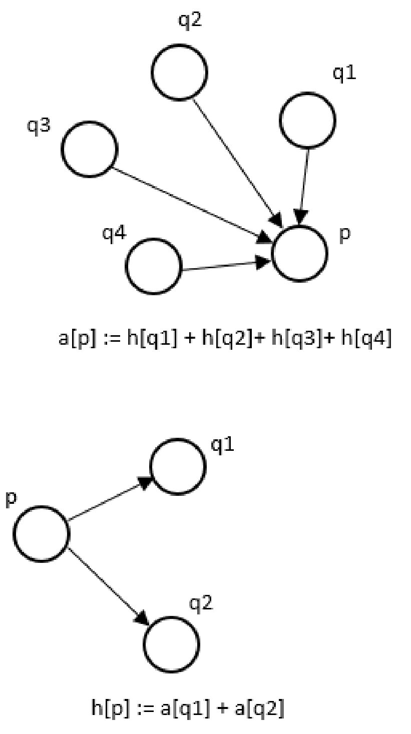

Figure 2. The basic operations for updating authority and hub scores.

We are now able to present a pseudocode corresponding to the HITS algorithm. But first, we describe some procedures that are used in the pseudocode although not being explicitly defined in this paper (their name is self-descriptive). The procedure Normalize(v) updates the array passed as parameter by dividing each value of it by the Euclidean Norm (square root of sum of the squares of all entries in v). Further, the function getMaxAndRemoveIt(v) returns the biggest value from v and removes it from the array (the array v is accordingly modified).

Let k be the number of iterations the algorithm performs and z the vector $(1,1,...,1)^T \in \mathbb{R}^n$. The algorithm HITS is defined as follows [5]:

```
Algorithm HITS
Data: $G_\sigma, k, c$
Result: $a_k, h_k$, best_authorities, best_hubs
Set $a_0 = z$
Set $h_0 = z$
for $i \leftarrow 1$ to $k$ do
    Apply $\mathscr{I}$ to $(a_{i-1}, h_{i-1})$ and obtain new $a$-weights $a'_i$
    Apply $\mathscr{O}$ to $(a'_i, h_{i-1})$ and obtain new $h$-weights $h'_i$
    Set $a_i$ = Normalize $(a'_i)$
    Set $h_i$ = Normalize $(h'_i)$
for $j \leftarrow 1$ to $c$ do
    best_authorities[j] = getMaxAndRemoveIt($a_k$)
    best_hubs[j] = getMaxAndRemoveIt($h_k$)
```

The vectors a_k and h_k store the authority and hub scores calculated in each iteration. However, these arrays are not sorted, so the last step that our algorithm needs to perform is filtering the top c authorities and the top c hubs (usually $c \approx 5-10$). This is done in the second for loop.

The space complexity of the algorithm is linear in the size of the set of edges of the subgraph, since G_σ (and implicitly its edge set) is computed at the beginning of the algorithm and does not need to be recalculated (and stored) in each iteration. The whole WWW network must be considered when the root set is determined and, afterwards, for each page p in the root set both $\Gamma_{in}(p)$ and $\Gamma_{out}(p)$ must be analyzed. Hence, the calculation of G_σ requires high computational costs. In a subsequent part of this paper we will present an idea to reduce the time complexity of the HITS algorithm.

C. Convergence

In the previous sections, we described the HITS algorithm but we did not make any assumption regarding its convergence. In his original paper [5], Kleinberg did not specify any formal convergence criteria for the algorithm. He just mentioned that an upper bound for the number of the required iterations is $k_{max} = 20$. In this section, we prove that the HITS algorithm converges to fixed points that can be easily computed for any adjacency matrix.

We recall the procedure Normalize(v) mentioned in the section *Hubs and Authorities Computation* and discuss its purpose. After each step of updating the authority and hub scores, the algorithm calls this function and, consequently, avoids the undesired situation in which each iteration would produce diverging scores.

It is intuitively clear that for arbitrarily large values of k, the algorithm converges to fixed values a^* and h^* (the bigger the number of the iterations is, the "better" the results of the algorithm are and given that S_σ is finite, there must exist a "best" result). We prove now that the algorithm always converges to the arrays a^* and h^* and try to find these.

The calculation of hub and authority scores can also be expressed using matrix operations. Suppose that the base set S_σ has n elements. Let $A \in \mathbf{N}^{n \times n}$ be the adjacency matrix of the subgraph G_σ on which the algorithm operates and let a and h be the vectors containing authority and hub weights respectively. The operation $\mathscr{I}$ for updating the authority scores of all pages in the subgraph can be written like this:

$$\mathscr{I} : a \leftarrow A^T h \tag{3}$$

Following the same idea, the operation $\mathscr{O}$ for updating the hub scores of all pages in the subgraph can be expressed in this way:

$$\mathscr{O} : h \leftarrow Aa \tag{4}$$

We set the initial scores $a_0 = \mathbf{1}$ and $h_0 = \mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^n$ denotes the column vector whose entries are all 1. After applying rules (3) and (4) k times and keeping in mind that authority scores a_k are computed using h_{k-1} and hub scores h_k are calculated using a_k we get:

$$a_k = A^T AA^T AA^T A ... A^T AA^T \mathbf{1} = (A^T A)^{k-1} A^T \mathbf{1}$$

and

$$h_k = AA^T AA^T AA^T ... AA^T \mathbf{1} = (AA^T)^k \mathbf{1}.$$

We recall now some notions from linear algebra that are important in our context. Let $M \in \mathbb{R}^{n \times n}$ be a symmetric matrix and λ a number such that $M\omega = \lambda\omega$ holds. Thus, λ is an eigenvalue of M and ω is the eigenvector associated to it (the set of all such ω is a subspace of $\mathbb{R}^n$ and forms the eigenspace corresponding to the eigenvalue λ; its dimension represents the multiplicity of the given eigenvalue). For such (symmetric) matrix, there exist at most n distinct real eigenvalues and the sum of their multiplicities is n. Let $\lambda_1(M)$, $\lambda_2(M)$,..., $\lambda_n(M)$ be these eigenvalues indexed in order of decreasing absolute value (each eigenvalue appears a number of times equal to its multiplicity). We recall some statements of Perron-Frobenius Theorem [14] [9]:

(i) The largest eigenvalue λ of M is positive and has multiplicity 1.
(ii) Each other eigenvalue of M is in modulus strictly less than λ.
(iii) The largest eigenvalue λ has a corresponding eigenvector with all entries positive.

Hence, we have:

$$|\lambda_1(M)| > |\lambda_2(M)| \geq \ ... \ \geq |\lambda_n(M)|. \tag{5}$$

For each distinct eigenvalue, we choose an orthonormal basis of its eigenspace. Let $\omega_1(M)$, $\omega_2(M)$, ..., $\omega_n(M)$ be the eigenvectors from these bases, where $\omega_i(M)$ belongs to the eigenspace of $\lambda_i(M)$. We call $\omega_1(M)$ the (unique) *principal eigenvector.* Let us show that the sequences a_1, a_2, ... , a_k and h_1, h_2, ... , h_k converge (to limits a^* and h^* respectively) [5].

The matrices AA^T and $A^T A$ are symmetric and thus have real eigenvalues. We note that a_k is the unit vector in the direction $(A^T A)^{k-1} A^T z$ and h_k is the unit vector in the direction $(AA^T)^k z$, where $z = \mathbf{1}$. From linear algebra we know that "for a symmetric $n \times n$ matrix M and a vector v not orthogonal to the principal eigenvector $\omega_1(M)$, the unit vector in the direction of $M^k v$ converges to $\omega_1(M)$ as k increases without bound" [5].

We set $M = AA^T$, $v = z$ and conclude that the sequence h_1, h_2, ... , h_k converges to $\omega_1(AA^T)$.

Following the same idea, one can show that $\lambda_1(A^T A) \neq 0$ (results from statement (i) above) and thus $A^T z$ is not orthogonal to $\omega_1(A^T A)$. Consequently, the sequence a_1, a_2, ... , a_k converges to $\omega_1(A^T A)$. Nevertheless, since both matrices $A^T A$ and AA^T have exclusively positive real entries, their principal eigenvectors have only positive entries (statement (iii)). Thus, the limits to which the authority and hub scores converge are positive.

We notice that the computation of the final authority and hub scores is reduced to the calculation of the principal eigenvectors of the matrices $A^T A$ (authority matrix) and AA^T (hub matrix) respectively. This can be realized by applying the Power Iteration algorithm [15], which has $O(n^2)$ complexity, with n being the size of the matrix. Both matrices are diagonalizable (can be easily proven), so the algorithm is suitable for our problem.

Another possibility is to use Singular Value Decomposition on the adjacency matrix $A = USV^T$. Thus, we can write:

$$A^T A = VS^T U^T USV^T = V(S^T S)V^T = V\Sigma V^T$$

and

$$AA^T = USV^T VS^T U^T = U(SS^T)U^T = U\Sigma U^T,$$

where Σ is a diagonal matrix containing the eigenvalues of A and S is a diagonal matrix containing the singular values (if σ_i is a singular value, then $\sigma_i = \sqrt{\lambda_i}$) of A. The first column vectors of V and U are the principal eigenvectors of $A^T A$ and AA^T respectively. Although this decomposition method is very costly (complexity $O(n^3)$) in comparison with the Power Iteration algorithm and requires a lot of memory (we need to store three matrices), it computes the principal eigenvectors corresponding to AA^T and $A^T A$ "simultaneously" (if we use Power Iteration, we need to run the algorithm twice in order to get both the hub and the authority vectors). Furthermore, we mention a drawback of the HITS algorithm: if the query is ambiguous (e.g. "java"), the root set may contain multiple subtopics or *communities* and the algorithm might favor only the dominant subtopic. In this context, the Singular Value Decomposition is a general tool to detect communities in the graph represented by the adjacency matrix A: if we consider the k eigenvectors of V (or $A^T A$) associated with the k largest eigenvalues, the largest authority score of each of these k vectors indicate a densely connected community [2].

D. Evaluation

In this section, we present some sample results which were obtained after applying the HITS algorithm on the web graph. The experimental data first appeared in Kleinberg's original paper describing the algorithm [5]. The query that has been used is "java". In the following, we list the top authorities for the given query. The corresponding authority score is indicated next to each link.

(java) Authorities
.328 http://www.gamelan.com/
(Gamelan)
.251 http://java.sun.com/
(JavaSoft Home Page)
.190 http://www.digitalfocus.com/digitalfocus/faq/howdoi.html
(The Java Developer: How Do I...)
.190 http://lightyear.ncsa.uiuc.edu/srp/java/javabooks.html
(The Java Book Pages)
.183 http://sunsite.unc.edu/javafaq/javafaq.html
(comp.lang.java FAQ)

According to Kleinberg, none of the authorities listed above appeared in the root sets. This fact is not surprising, since the majority of the retrieved pages do not contain any occurrences of the initial query string, despite being relevant to the searched topic. Furthermore, the author (indirectly) divides each experiment into two stages: a first one, which includes a "black-box" call to a text-based search engine (AltaVista) and produces the root set R_σ, and a (more extensive) second one, which ignores the textual content of the pages and focuses exclusively on the link structure of the web. Thus, we can conclude that a substantial amount of the HITS algorithm can be achieved through a "pure" analysis of the link structure.

E. Strengths and Weaknesses

So far, we have described the HITS algorithm without questioning its performance. In this section, we highlight the most important advantages and disadvantages of the algorithm.

Although synthesized here into just two paragraphs, the strengths of HITS recommend it to be one of the most used ranking algorithms nowadays. In the following, we present the main advantages of HITS.

A1 In terms of resource demand, HITS is very efficient (has linear space complexity in the size of the set of edges in G_σ), given that the dense subgraph G_σ on which the algorithm operates is relatively small compared to the whole web graph. Thus, the size of the adjacency matrix A is moderate, this fact constituting a considerable advantage of our algorithm.

A2 HITS is sensitive to user query (as opposed to PageRank) and it ranks pages according to the query string by computing relevant authority and hub pages. Experiments [5] have shown that HITS calculates authority and hub scores correctly, so the algorithm can be applied with the guarantee that the results maximize the satisfaction of the user with respect to the ranked pages.

HITS algorithm was originally proposed two decades ago, when the web network was considerably smaller than it is today. Some of the weaknesses that even Kleinberg anticipated at that time have become more evident with the expansion of the web graph (best example is D1), while other drawbacks have appeared as a result of the vast (growing) topic diversity of the web (see D2). In the following, we briefly describe the most important disadvantages of the HITS algorithm.

D1 Since HITS is a query dependent algorithm, one of its drawbacks is the substantial amount of expense at query-time [4]. The computation of the neighborhood graph G_σ having the vertex set S_σ is the most expensive step of the algorithm. The subsequent calculation of scores for the vertices in G_σ is cheaper, although Power Iteration (time complexity $O(n^2)$ for each iteration) is usually used to compute the fixed-points of the hub and authority vectors. The considerable computational cost at query-time is one disadvantage of HITS over query-independent algorithms such as PageRank.

D2 HITS performance may be affected by the topic drift problem [8], which is caused by the tightly-knit community effect (TKC). A tightly-knit community is a small but highly interconnected set of pages [6]. The TKC effect happens when such a community gets high scores in link-analysis algorithms (HITS!), although the pages contained in the TKC are not authoritative on the topic. Thus, the topic drift problem appears at a greater extent in the base set, which may not contain the most relevant pages for a given query and, consequently, the algorithm might compute unsatisfactory results.

D3 The algorithm is suitable only for search engines that operate on small sets of documents and for a limited number of users. The cause of these limitations is the fact that the operations $\mathscr{I}$ and $\mathscr{O}$ must be executed on the fly at query time [1], given that the adjacency matrix of the focused subgraph is defined only after the user types his query. Hence, HITS is not feasible for today's search engines, which need to handle tens of millions of queries per day.

D4 HITS has small robustness to spam. Spamming is the act of fraudulently increasing the relevance of specific pages (by creating subgraphs that e.g. contain numerous hyperlinks from artificially created sites to a page whose relevance is meant to be increased), which results in these appearing among the first results for a user query [1]. Our algorithm has no method to prove the "notoriety" of a page that is used for calculating the hub and authority scores and, therefore, has small resistance to spam.

D5 The algorithm cannot identify irrelevant authorities. HITS naively assumes that every page q referenced by a page p which is in the root or base sets is relevant for the topic induced by the query. Hence, the authority score of an irrelevant page q might be "illegally" boosted in case the referencing page p is not (closely) connected to the query. Although p might be in R_σ and thus contain the query string, it is not guaranteed that p is relevant to the searched topic. This is a weakness of the general full-text-search technique which is applied in the step *Subgraph Computation* of our algorithm.

D6 The algorithm cannot identify irrelevant hubs. An undesired situation may occur when a page p that contains links to a large number of separate topics may receive a high hub score although p is not relevant to the given query. Together with the disadvantage described above (D5), this drawback is strongly related to the topic drift and spamming problems which were previously mentioned.

F. PageRank and HITS

Currently used in the Google search engine, PageRank is also a link analysis algorithm, but it is based on a general idea which is completely different from the HITS approach. A detailed explanation of PageRank can be found here [13]. In the following, we present the most important differences (from a mathematical point of view) between the HITS and PageRank algorithms.

In PageRank, the importance of a page can be judged by looking only at the pages that link to it, while in HITS both in- and out-links are considered. Also, a link from a page i to j indicates that j is relevant to the topic covered in i and the greater the number of pages that point to j is, the more important j is considered. However, the PageRank approach is not only about counting the in-links. A web page which has only one page p pointing to it may be more important than another page that has 15 in-links if p is an authoritative site, such as *www.google.com* or *www.cnn.com* [10].

Another difference between these algorithms refers to the type of matrix which is used for calculating the rank of each web page. HITS basically works on the adjaceny matrix of an induced subgraph of the webgraph, while PageRank builds a stochastic matrix by analyzing the out-links of each page in the *WWW* network. Thus, in the original PageRank algorithm, the importance vector (i.e. vector which assigns a rank to each page) in the k-th iteration is $v_k = A^k v$, where v is the initial vector. Moreover, the sequence of vectors $v, Av, A^2v, ...A^kv$ tends to an equilibrium value v^*, which is usually chosen to be the unique eigenvector to the eigenvalue 1, having the sum of all entries 1 (probabilistic vector) [10]. This equilibrium value is also the stationary distribution of a Markov chain that describes a random walk on the web. Hence, PageRank uses both probabilistic and link analysis ranking models, while HITS is based only on a link analysis model.

While HITS has no method to deal with tightly-knit communities or simply with a graph that consists of disconnected components (honestly speaking, the induced subgraph which HITS computes is rarely such a graph due to the way in which the base set is obtained from the root set), PageRank was adapted by Page and Brin such that the random surfer on the web can never stop its "walk" once it reaches a page with no outgoing links. Thus, instead of using a matrix A in which each element from the row i of the column j is $1/n$ (with n being the number of pages to which page j points to) if there is a link from j to i and 0 otherwise, a new matrix M is computed:

$$M = (1-p)\cdot A + p\cdot B,\ B = \frac{1}{n}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

where the damping factor p is the probability that the surfer quits the current page and "teleports" to a new one (which is equally chosen from all the pages in the web graph) [10]. The matrix M is column stochastic and the (final) PageRank vector is the probabilistic eigenvector of M corresponding to the eigenvalue 1.

Being link analysis algorithms, PageRank and HITS are applications of spectral graph theory that aim to rank the webpages given a specific query. Both algorithms can be reduced to eigenvector problems and are nowadays applied in different information retrieval systems.

III. Improvements

Ever since it was invented, the HITS algorithm has been continuously analyzed by numerous researchers who attempted to find new techniques that can enhance its performance. In this section, we present three improvements of HITS that address different drawbacks.

A. Bloom Filters-Based Approach to Speed Up Subgraph Computation

We recall a considerable weakness of the HITS algorithm: the substantial amount of expense at query-time dedicated to the computation of the focused subgraph G_σ which structures the results to a given query. The technique that we present in the following was introduced by Sreenivas Gollapudi, Marc Najork, and Rina Panigrahy [4] and moves the most expensive part of the computation offline. It builds a neighborhood graph (we call it N) that will be used to calculate the HITS scores. The idea is to create, at index-construction time, a database that maps web page URLs to summaries of their neighborhoods. Later, at query-time, the results satisfying a query are ranked by performing these steps: first, each page in the root set is looked up in the summary database, then, the neighborhood graph N is approximated based on the neighborhood summaries of the pages in the root set and, finally, the operations for calculating the hub and authority scores are performed on the graph N.

We first need to define what the summary of the neighborhood graph of a web page u is. We call a page v an *ancestor* of u if v contains a hyperlink to u. The page v is a *descendant* of u if the latter page points to v. The summary of the neighborhood graph of a web page u consists of a summary of the ancestors and a summary of the descendants of u, each being represented by a Bloom filter. Hence, a Bloom filter contains a subset of limited size of ancestors or descendants plus a subset of web page identifiers [4].

A Bloom filter [11] is a space-efficient probabilistic data structure used to test the membership of an element in a given collection. It represents a set using an array F of m bits and it utilizes k hash functions $h_1.h_2,...,h_k$ to manipulate the array. Each hash function h_i maps an element of the set to a value in $[1,m]$. False positive matchings (i.e. test says that an element belongs to the set, although this is not true) are possible, but false negatives (i.e. test says that the element is not in the set, although this assumption is false) are not.

An element e is added to the set by setting $F[h_i(e)]$ to 1 for each $i \in [1,k]$. To test whether e is in the set, one has to check if $F[h_i(e)]$ is 1 for all $i \in [1,k]$. Removing an element is impossible in a Bloom filter, since false negatives are not allowed (an element maps to k bits, and even though setting one of those bits to 0 suffices to remove the element, it may also cause the removal of other elements that "share" the modified bit). For a Bloom filter of size m and a set having n elements, the optimal number of hash functions k (so that the false-positives are minimized) is $\frac{m}{n}\ln 2$ [11]. The probability of false positives is, in this case, $(\frac{1}{2})^k$ [4]. Figure 3 illustrates a Bloom filter representing the set $\{x,y,z\}$. There are 3 used hash functions (this can be deduced from the differently colored arrows that map each element to 3 bits being set to 1). The membership test of the element w is illustrated in the lower part of the figure: because the application of one hash function to w results in one bit being 0, the element w is surely not in the Bloom filter.

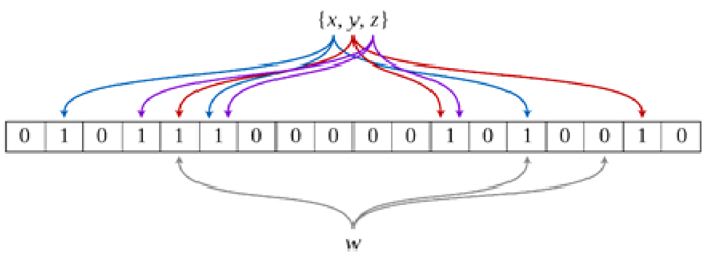

Figure 3. Bloom filter representing the set {x, y, z} (Source: [11]).

The proposed technique is based on consistent sampling. We use the notation $BF[X]$ to denote a Bloom filter representing the set X. Let $C_n[X]$ be a consistent unbiased sample of elements from the set X and $C_n[X] = X$ if $|X| < n$. Let further $I_n(u) = C_n[v \in V : (v,u) \in E]$ be a consistent sample of (at most) n of the ancestors of a node u and $O_n(u) = C_n[v \in V : (u,v) \in E]$ be a consistent sample of n descendants of u. We compute a summary $(BF[I_a(u)], I_b(u), BF[O_c(u)], O_d(u))$ for each page u in the web graph, where $a,b,c,d \in \mathbb{N}$ depend on the size of the neighborhood graph we want to have. This operation is done at index-construction time.

At query time, we construct a *cover set* C by looking up the summaries for all elements in the root set:

$$C = R \cup \bigcup_{u \in R} I_b(u) \cup \bigcup_{u \in R} O_d(u).$$

Next, we compute the neighborhood graph $N = (C,E)$ whose vertices are the pages contained in the cover set and the edges are constructed as follows: given two vertices $u \in R$ and $v \in C$, we first check if the Bloom filter $BF[I_a(u)]$ contains v. If the membership test returns true, we add an edge (v,u) to the graph. Similarly, we verify whether $BF[O_c(u)]$ contains v and, in case the answer is positive, we add an edge (u,v) to N. Hence, the graph N is our approximation of the neighborhood graph of R. In the succeeding steps of our algorithm, this graph is used to calculate the HITS scores by applying the procedure described in the section *Hubs and Authorities Computation.*

The approximate neighborhood graph N is different from the graph G_σ that we described in a preceding section. We constructed N by sampling the pages directly reachable from the root set (we computed $O_d(u)$ for every $u \in C$). However, the exact graph G_σ contains all vertices pointed to by at least one of the elements of the root set. Consequently, the size of N is smaller (and thus the computational effort required to build the graph is reduced), but experiments [4] have proven that this fact does not affect the quality of the results computed by the HITS algorithm.

Another difference between N and G_σ is that the approximate graph contains edges pointing from $C \cap I_c(u)$ to $u \in R$ and from $u \in R$ to $C \cap O_c(u)$. This means that the edges between vertices in C that are not elements of the root set are excluded.

The last considerable issue that stops the graphs N and G_σ from being identical is the way in which we represented $I_a(u)$ and $O_c(u)$: we did not use exact set representations, but instead we approximated these collections by using Bloom filters. Thus, additional edges have been introduced and their number depends on the false positive probability of the Bloom filters (the higher the probability, the larger the number of spurious edges is).

Although it might not seem evident, the proposed approximation of the neighborhood graph preserves the characteristics relevant to a ranking algorithm. The sampling process and thus the exclusion of specific edges, plus the inclusion of "phantom" edges as a result of getting false positives in the Bloom filter are factors against the implementation of this approach. On the other hand, consistent sampling preserves co-citation relationships among pages in the root set and the experimental validation conducted by the authors of this technique [4] shows that the described algorithm maintains the properties relevant to HITS.

B. I-HITS

Being a purely link-based algorithm, HITS does not consider the content of a page when it analyzes its importance. As suggested multiple times in this paper, the number of links that point to a page is the only measure for determining its degree of relevance. These links are not treated differently (i.e. they are all "equally" important) and this approach often leads to a poor quality of the results that the algorithm computes, meaning that, although some pages are not relevant to the search query, they receive high authority scores and are thus "recommended" to the user. The (indirectly) described problem is called topic drift and its cause is the inclusion of authoritative, yet irrelevant pages in the base set.

In the following, we present the I-HITS algorithm (Improved HITS) introduced by Xinyue Liu, Hongfei Lin and Cong Zhang [7] that aims to solve the topic drift problem by computing the similarity and popularity for each page $p \in S_\sigma$. These new benchmarks reflect the relevance of a specific page to a specified query.

Given a query σ and a page p, let S_p be the similarity between p and σ. The *similarity* measures how relevant the page is with respect to the the query topic. It can be computed using the cosine similarity, if we consider the query and the page as two vectors with real values and if we are interested in the number of terms that appear both in the query and in the page. Let i be a source page and j be a target page so that i contains a hyperlink to j. The anchor text describes the target page j, summarizing its topic with a high degree of accuracy [7]. Hence, a rational idea is to compute the similarity between the anchor text and the query σ instead of the similarity between the target page j and σ. This way, the computational complexity is significantly reduced. The elements of adjacency matrix A (which is actually not an adjacency matrix - in the common sense - anymore) are now expressed as follows:

$$A_{i,j} = \begin{cases} (1+S_i)\cdot(1+S_j) & \text{if there is a link from } i \text{ to } j \\ 0 & \text{otherwise.} \end{cases}$$

Thus, this matrix does not only indicate whether a page i contains a link to a page j, but it also reflects the similarity of the two considered pages with respect to the query.

We focus now on the notion of *popularity* of a page p. Intuitively, a page is considered popular if there exist other pages which point to p. HITS measures popularity only quantitatively, assuming that the larger the number of pages that reference p is, the more popular p becomes. On the other side, the I-HITS algorithm appreciates popularity both qualitatively and quantitatively and introduces two types of popularity that are important in the calculation of the hub and authority scores.

The idea is that each page p that points to k pages $q_1, q_2, ..., q_k$ assigns popularity-scores $W_{(p,q_i)}$ to each $q_i, i \in [k]$ such that $\sum_{i\in[k]} W_{(p,q_i)} = 1$. The popularity scores are calculated based on the "hubness" (W^{out}) or on the "authoritiness" (W^{in}) of the target page [7].

Let $I_i = deg_{in}(i)$ and $O_i = deg_{out}(i)$ denote the number of pages that point to i and the number of pages that i points to respectively. Also, $R(j)$ represents the set of all pages to which j contains references.

We define:

$$W^{out}_{(j,i)} = \frac{O_i}{\sum_{p\in R(j)} O_p}$$

to be "the popularity from the number of outlinks" [7] or the popularity-score conferred to i with respect to the link from j to i. The formula above takes into consideration not only the out-degree of i, but also the out-degree (and thus the hub values) of other pages that are referenced by j.

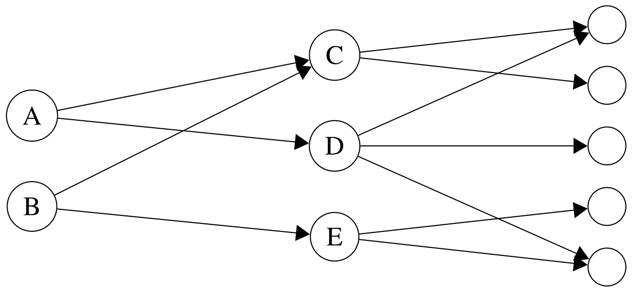

Figure 4. Example of a linked structure of the web (Source: [7]).

Similarly, we define:

$$W^{in}_{(j,i)} = \frac{I_i}{\sum_{p\in R(j)} I_p}$$

to be "the popularity from the number of inlinks" [7] which will be used in computing the authority score of i.

Based on Figure 4, we now want to decide whether page C is more popular than page D. We compute $W^{out}_{(A,C)} = O_C/(O_C + O_D) = 2/(2+3) = 2/5$ and $W^{out}_{(A,D)} = O_D/(O_D+O_C) = 3/(3+2) = 3/5$. It follows that page C is a more popular hub than page D with respect to the source page A. Further, we analyze which page is a more popular authority. We compute $W^{in}_{(A,C)} = I_C/(I_C+I_D) = 2/(2+1) = 2/3$ and $W^{in}_{(A,D)} = I_D/(I_D+I_C) = 1/(1+2) = 1/3$ and conclude that C is more authoritative than D.

We define now two operations on weights, namely $\mathscr{I}'$ and $\mathscr{O}'$, that represent the core of the I-HITS algorithm. We denote with $B(j)$ the set of all pages that contain links to j and with $F(j)$ the collection of all pages to which j points. Given the weights $\{a^{\langle p\rangle}\}$ and $\{h^{\langle p\rangle}\}$ for the pages in the subgraph G_σ, the $\mathscr{I}'$ operation updates the authority scores as follows:

$$\mathscr{I}' : a^{\langle i\rangle} = \sum_{j\in B(i)} h_j \cdot (1+s_i)\cdot(1+s_{ji}) \cdot \frac{I(i)}{\sum_{p\in F(j)} I(p)}.$$

The $\mathscr{O}'$ operation updates the hub scores following a similar principle:

$$\mathscr{O}' : h^{\langle i\rangle} = \sum_{j\in F(i)} a_j \cdot (1+s_i)\cdot(1+s_{ij}) \cdot \frac{O(i)}{\sum_{p\in B(j)} O(p)}.$$

We note that the two operations are significantly different compared to the operations $\mathscr{I}$ and $\mathscr{O}$ that the HITS algorithm performs. While HITS computes the authority score of a page p only by summing up the hub values of the pages that point to p, I-HITS also considers the similarity of the page to the given query and the page's popularity among other authorities in the network.

```
Algorithm I-HITS
Data: $G_\sigma$
Result: $a_i, h_i$
for $p \in S_\sigma$ do
    Set $a_0^{\langle p \rangle} = 1$
    Set $h_0^{\langle p \rangle} = 1$
i = 0
while $a_i$ and $h_i$ do not converge do
    Set i = i + 1
    forall $p \in S_\sigma$ do
        Apply $\mathscr{I}'$ and obtain new authority value $a_i^{\langle p \rangle}$
        Apply $\mathscr{O}'$ and obtain new hub value $h_i^{\langle p \rangle}$
```

In the following, we present an experiment conducted by the researchers who proposed the I-HITS algorithm [7] in 2012. They distinguished three types of web pages with respect to their relevance to a specific query. Thus, a page can be:

1) Highly relevant (HR): if it has a high authority score and contains very important information.
2) Relevant (R): if it has relevant, but not important information.
3) Not-relevant (NR): if it does not include neither the keywords of the given query, nor relevant information about it.

The term "important" may be ambiguous in our context, so we first need to clarify how the appropriate category is chosen for a specific page. An objective classification of the results computed by I-HITS (or HITS) is achieved based on the responses of several impartial people (given a query σ and a page p, they are asked how relevant p with respect to σ is) and other criteria (TREC's P@10 evaluation criteria [7]).

The experiment compared the results computed by HITS and I-HITS for the same queries. These are "alcohol" and "搜索引擎" ("search engine"). Table 1 describes the base sets S_1 and S_2 on which the algorithms operated (Nodes := $|S_i|$, $i \in \{1,2\}$, Links := total number of hyperlinks in the graph that has S_i as vertix set).

Table I
EXPERIMENTAL DATA

Query	Nodes	Hubs	Authorities	Links
alcohol	1964	1441	1213	11083
search engine	2884	2142	1744	37941

We present now the results to the query "alcohol" calculated by HITS and I-HITS. Some of the websites that were ranked by HITS and I-HITS in this experiment do not exist anymore (we marked them accordingly). The highly relevant (HR) and relevant (R) pages are written in bold and italics respectively. The results are listed in decreasing order of relevance.

Results computed by HITS:
(1) **https://www.niaaa.nih.gov/**
(2) **http://www.health.org** (this website no longer exists)
(3) **http://faculty.washington.edu/chudler/alco.html**
(4) *http://www.alcoholfreechildren.org/*
(5) **https://nasadad.org/**
(6) *https://www.nofas.org/*
(7) **http://ncadi.samhsa.gov/govpubs/ph323** (this website no longer exists)
(8) **https://alcoholchange.org.uk/**
(9) *http://www.atf.treas.gov/* (this website no longer exists)
(10) **http://www.hsph.harvard.edu/nutritionso** (this website no longer exists)

Results computed by I-HITS:
(1) *http://www.ndp.govt.nz/publications/review-* (this website no longer exists)
(2) **https://www.wrap.org/**
(3) **http://www.alcoholmedicalscholars.org/**
(4) **https://www.niaaa.nih.gov/**
(5) **http://www.health.org** (this website no longer exists)
(6) **https://alcoholchange.org.uk/**
(7) **http://ncadi.samhsa.gov/govpubs/ph323** (this website no longer exists)
(8) **http://faculty.washington.edu/chudler/alco.html**
(9) **https://www.cdc.gov/alcohol/**
(10) *http://www.atf.treas.gov/* (this website no longer exists)

Analyzing both tops, we notice that I-HITS computes slightly more highly relevant pages in comparison to HITS (8 vs. 7). However, the first ranked page by I-HITS is only "relevant", while the first website suggested by HITS is highly relevant to the given query. Hence, this example query does not convince us that I-HITS is more powerful than our initial algorithm.

Let us now analyze the results to the query "search engine". It is important to mention that the initial query was written in Chinese and, therefore, the results are predominantly Chinese sites.

Results computed by HITS:
(1) **https://www.google.com/?hl=zh-CN**
(2) *http://www.gseeker.com/*
(3) http://www.wangtam.com/50226711/c_wav
(4) http://www.yuleguan.com/
(5) https://www.chinaventurenews.com/
(6) http://www.tjacobi.com/
(7) http://www.money-courier.com/ (this website no longer exists)
(8) http://www.geekervision.com/
(9) http://www.in-women.com/
(10) https://www.tracingadget.com/ (this website no longer exists)

Results computed by I-HITS:
(1) *http://www.xpue.net* (this website no longer exists)
(2) *http://www.toooold.com/* (this website no longer exists)
(3) *http://www.bbssearch.cn/* (this website no longer exists)
(4) **https://www.google.com/?hl=zh-CN**

(5) http://www.1hd.cn/ (this website no longer exists)

(6) **http://www.baidu.com/**

(7) *http://bizsite.sina.com.cn/ (*this website no longer exists)

(8) http://it.sohu.com/7/0903/35/column213613 (this website no longer exists)

(9) **http://www.youdao.com/?keyfrom=so163redir**

(10) https://www.qq.com/?froma

We observe that 2 out of the top 10 results returned by HITS are "relevant" and 1 of them is "highly relevant" to the search topic. On the other side, 7 of the top 10 pages ranked by I-HITS are relevant and 3 of them are highly relevant. Thus, I-HITS improves the proportion of highly relevant and relevant pages by 20% and 50% respectively.

Moreover, this example illustrates one of the biggest drawbacks of HITS, namely the topic drift problem. If we access the links from the top computed by HITS, we notice that these are far from being relevant to the topic query. For example, the fifth ranked page advertises a drug for giving up smoking and the ninth one is a website that sells clothes for women. However, the top computed by I-HITS is not "affected" by this problem, since 7 out of 10 pages are relevant to the search topic. Consequently, this example query illustrates the advantage of I-HITS over HITS with respect to the topic drift problem.

We can conclude that there are situations in which I-HITS can compute better results than the Hubs and Authorities algorithm. However, applying I-HITS to rank a set of pages does not guarantee the absence of the topic drift problem.

C. A Stochastic Approach for Link-Structure Analysis (SALSA)

Probably the most popular improvement of HITS is the SALSA algorithm, which was originally proposed by R. Lempel and S. Moran in 2000. SALSA is based on the stochastic properties of random walks, and assumes that authorities should be "visible" from many pages in the subgraph $G_\sigma = (S_\sigma, E)$ induced by the base set. Thus, a random walk on this subgraph will visit authorities with high probability [17]. The algorithm defines and analyzes two Markov chains - a chain of hubs and a chain of authorities -, whose state spaces contain all pages from the base set. The transition matrices of these two Markov chains are the new authority and hub matrices and are used to compute both scores associated to a page.

The subgraph computation step in SALSA is similar to the one performed by HITS. Each non-isolated page s from the subgraph is represented by two new nodes s_a and s_h. Further, SALSA builds a *bipartite undirected graph* $\hat{G} = (V_h, V_a, E)$ with the vertex sets:

$$V_h = \{s_h \mid s_h \in S \text{ and out-degree}(s_h) > 0\},$$

$$V_a = \{s_a \mid s_a \in S \text{ and in-degree}(s_a) > 0\}$$

and edge set:

$$E = \{\{s_h, r_a\} \mid s_h \in V_h,\ r_a \in V_a \text{ and } s_h \to r_a \text{ in } S\}.$$

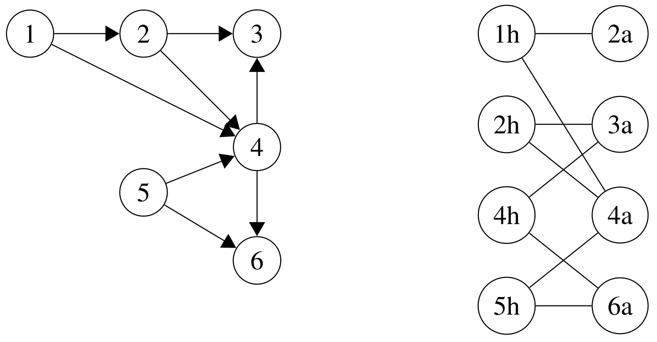

Figure 5. Transforming the graph on the left side into a bipartite graph [17].

Hence, a link from a source page s to a target page r is represented by an undirected edge between s_h and r_s. Figure 5 illustrates the transformation of a non-bipartite graph into a bipartite one.

The algorithm defines two Markov chains: $(X_n)_{n=0}^{\infty}$ with state space V_a and $(Y_n)_{n=0}^{\infty}$ with state space V_h. An important fact is that these Markov chains are random walks, but not in the "normal" sense: state transitions are generated by traversing two links in a row, one link forward and one link backwards (or the other way round) [17]. For example, in Figure 5 a step in a random walk consists of going from state 2h, then to 3a and then to 4h.

Naturally, these Markov chains start off from different sides of $\hat{G}$ (i.e. if μ_0^a and μ_0^h are the initial distributions of the chain of authorities and the chain of hubs respectively, then $\mu_0^a(i) > 0$ if $i \in V_a$ and $\mu_0^a(j) = 0$ for all $j \in V_h$ and $\mu_0^h(j) > 0$ if $j \in V_h$ and $\mu_0^h(i) = 0$ for all $i \in V_a$). Also, each random walk visits nodes only from one side of $\hat{G}$, given that it traverses paths consisting of two edges in each step [17].

It is intuitively clear that the best authorities for a given query σ should be visible from many pages in $\hat{G}_\sigma$ (i.e. reachable from many other pages by a direct link or by short paths) and hence a random walk on V_a will visit them with high probability [17]. Following the same reasoning, the best hubs will be among the vertices most often visited by a random walk on V_h.

Let us now analyze the transition matrices of these two Markov chains. We define the stochastic matrix (i.e. matrix having the sum of all entries in a row equal to 1) $\hat{H}$ corresponding to the chain of hubs as follows:

$$\hat{h}_{i,j} = \sum_{\{k \mid (i_h,k_a),(j_h,k_a) \in \hat{G}\}} \frac{1}{deg(i_h)} \cdot \frac{1}{deg(k_a)}.$$

The (i,j)th entry of the stochastic matrix associated to the chain of authorities $\hat{A}$ is calculated in the following way:

$$\hat{a}_{i,j} = \sum_{\{k \mid (k_h,i_a),(k_h,j_a) \in \hat{G}\}} \frac{1}{deg(i_a)} \cdot \frac{1}{deg(k_h)}.$$

Note that $deg(p_i)$ is the in-degree of page p_i if $i = a$ (i.e. the page is on the authority side of the graph) and out-degree of p_i otherwise (if $i = h$ and thus p_i is on the hub side).

A positive transition probability $\hat{a}_{i,j}$ means that there exists at least one node k which points to both nodes i and j and thus

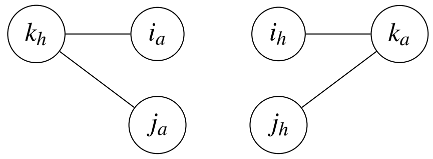

Figure 6. Visual representation of positive probabilities $\hat{a}_{i,j}$ (left) and $\hat{h}_{i,j}$ (right).

page j is reachable from i in two moves (which represent one step in the random walk starting on the authority side - see Figure 6.): we start from i_a, follow the edge $\{i_a, h_k\}$ and then the edge $\{h_k, j_a\}$. A positive probability $\hat{h}_{i,j}$ can be explained in a similar manner.

Let us now focus on the main idea of the HITS algorithm and its improvement SALSA, namely the computation of the hub and authority scores. The stochastic matrices $\hat{H}$ and $\hat{A}$, which we previously described, serve as the basis for the calculation of these values. Thus, the principal eigenvector of $\hat{H}$ contains the scores corresponding to the best hubs in our subgraph. Similarly, the principal eigenvector of $\hat{A}$ stores the authority values of the best authorities. The fact that the principal eigenvector of the authority (hub) matrix contains the best authorities (hubs) can be explained using notations of linear algebra (as we did in section *Convergence*) or analyzing the Markov chain of authorities (hubs).

We assumed that the graph $\hat{G}$ is connected and, therefore, both Markov chains are irreducible (because for every two states (pages) i and j we can reach j from i). Also, both chains are aperiodic (i.e. have period 1), because when visiting any authority (hub), there is a positive probability to revisit it on the next entry to the authority (hub) side of the bipartite graph [6] (given that all the nodes are non-isolated). Hence, both Markov chains are ergodic. Further, the Ergodic Theorem tells us that the principal eigenvector of an ergodic Markov chain is actually its stationary distribution. Thus, if π is the stationary distribution of the chain of authorities and a_n is the distribution of the Markov chain for n-th step of the random walk, we have:

$$\lim_{n \to \infty} a_n = \pi.$$

Therefore, the entries with the highest value of π correspond to the sites most frequently visited by the (infinite) random walk on the authority side [6]. Thus, the SALSA algorithm converges to the stationary distribution of the Markov chain of authorities (because we are searching for the best authorities).

Another way to calculate the authority and hub scores is by analyzing two matrices obtained by performing some operations on A. We recall that A is the adjacency matrix associated to the subgraph on which our algorithm operates. Let A_r be the matrix which results by dividing each nonzero entry by the sum of the entries in its row and, similarly, let A_c be the matrix obtained by dividing each nonzero entry by the sum of the entries in its column. With this in mind, the matrix $\hat{H}$ consists of the nonzero rows and columns of $A_r A_c^T$ and $\hat{A}$ is composed of the nonzero rows and columns of $A_c^T A_r$ [6]. Thus, the SALSA algorithm uses both row (A_r) and column (A_c) weighting when computing the hub and authority scores (we remember that HITS does not use any type of weighting, it just considers the adjacency matrix A when computing both page scores).

Experiments comparing the results computed by HITS and SALSA show that the main advantage of SALSA over HITS is that it is less vulnerable to the TKC effect. The reason for this is the improvement of the subgraph quality [17]. In the following, we present an experiment conducted by the inventors of the SALSA algorithm [6].

The single-topic query that has been used is "movies". At the expansion of the root set (i.e. when constructing the base set), the links were filtered: intra-doimain links (these are usually navigational aids inside an intranet and hence do not confer "appropriate" authority), *CGI scripts* (*CGI* stands for Common Gateway Interface and defines a standard way in which information may be passed to and from the browser and server) and ad-links were ignored when constructing the focused subgraph. The results computed by the HITS algorithm are listed in the following (the value of the link in the pricipal eigenvector of the authority matrix is shown next to each link):

(movies) Authorities - HITS
.1673 http://go.msn.com/npl/msnt.asp
(MSN.COM)
.1672 http://go.msn.com/bql/whitepages/asp
(White Pages - msn.com)
.1672 http://go.msn.com.nsl/webevents.asp
(Web Events)
.1672 http://go.msn.com/bql/scoreboards.asp
(MSN Sports scores)

According to R. Lempel and S. Moran, the top 30 authorities computed by Kleinberg's algorithm were all *go.msn.com* sites and were not at all relevant to the given query. Furthermore, except for the first one, all the results received the same authority score (0.1672). Another interesting fact is that, given that same-domain links were not allowed, none of the top authorities was pointed by a *go.msn.com* site. Although HITS is, to some degree, prone to give the best authority scores to rather irrelevant pages, it still does not (normally) compute such horrible results... To understand why did these four pages score so well, let us now analyze the principal community of hubs (i.e. the best hubs) for our query "movies":

(movies) Hubs - HITS
.1692 http://denver.sidewalk.com/movies
(movies: denver.sidealk)
.1619 http://boston.sidewalk.com/movies
(movies: boston.sidewalk)
.1688 http://twincities.sidewalk.com/movies

(movies: twincities.sidewalk)
.1686 http://newyork.sidewalk.com/movies
(movies: newyork.sidewalk)

All these hubs are part of the *Microsoft Network (msn)* [6] and point to whole the set of authorities computed by HITS (the fact that almost all authorities received the same score is now explained). Also, these hubs are elements of the root set and contain the query string "movie". Thus, a tightly-knit community consisting of the elements that are the best authorities (according to HITS, at least) has been created as a result of the fact that many of the pages in the root set point to them. Let us now consider the results computed by SALSA for the same query "movies":

(movies) Authorities - SALSA
.2533 http://us.imdb.com/
(The Internet Movie Database)
.2233 http://www.mrshowbiz.com/
(Mr Showbiz)
.2200 http://www.disney.com/
(Disney.com-The Web Site for Families)
.2134 http://www.hollywood.com/
(Hollywood Online:...all about movies)
.2000 http://www.imdb.com/
(The Internet Movie Database)
.1967 http://www.paramount.com/
(Welcome to Paramount Pictures)
.1800 http://www.mca.com/
(Universal Studios)
.1550 http://www.discovery.com/
(Discovery Online)
.1533 http://www.film.com/
(Welcome to Film.com)
.1300 http://www.mgmua.com/
(mgm online)

We notice that the results listed above are all relevant to the given query. According to the authors of the experiment [6], similar results (i.e. the same community of authorities) could be computed by HITS if the rows and columns corresponding to the top 30 authorities from the matrix A^TA are deleted. This means that the *msn.com* community is "removed" from our network.

The previous example illustrates the TKC effect and the extent to which HITS and SALSA are vulnerable to it.

To sum up, SALSA computes a weighted in/out-degree ranking, as we could see when we described the structure of the transition matrices $\hat{H}$ and $\hat{A}$. Therefore, this algorithm is computationally lighter than HITS, this fact representing a fundamental advantage, given that both SALSA and HITS are computed at query time. SALSA can also deal better with the TKC effect, as we could see from the example which we presented. An interesting fact is that Twitter uses a SALSA style algorithm to recommend users what accounts to follow [16].

IV. Conclusion

HITS is a link analysis algorithm that rates web pages by assigning them a hub and an authority score. Instead of performing its steps on the whole web network, the algorithm constructs a focused subgraph by retrieving the pages that contain a given query string and including other pages which point to or are pointed by the former ones.

The HITS algorithm introduces two weights for each page, an authority and a hub weight. The former indicates the degree to which the page contains relevant information to the given query and the latter reflects the value of the links included in the page. Moreover, HITS updates the hub and authority scores in an iterative manner and it always converges to fixed points which can be computed using the adjacency matrix of the subgraph. In fact, the computation of the final hub and authority values is reduced to finding the principal eigenvectors of the matrices AA^T and A^TA respectively. To realize this, the Power Iteration method is generally used, although other approaches, such as Singular Value Decomposition are also possible and have specific benefits.

Because it is query dependent, the algorithm has a significant drawback related to the substantial amount of time required for computing the subgraph. As a solution to this issue, an idea based on Bloom filters has been described. Another weakness of the HITS algorithm is the impossibility of avoiding the TKC effect and the topic drift problem, given that the links in the network are all treated equally. In this sense, two improvements have been presented in this paper. The former is the I-HITS algorithm, which calculates the popularity and the similarity of each page and afterwards computes the corresponding hub and authority scores. The latter is the SALSA algorithm, which takes into consideration the stochastic properties of two random walks performed on the focused subgraph induced by a given query.

In my opinion, the power of HITS lies in the idea of considering (almost exclusively) the link structure of the webgraph when determining the most relevant pages for a specific query. Being an application of spectral graph theory, the HITS algorithm represents an innovative approach for the ranking process performed by every modern search engine.

References

[1] Giorgio Ausiello and Rossella Petreschi Eds. *The Power of Algorithms - Inspiration and Examples in Everyday Life*. Springer-Verlag, 2013.

[2] Dr. Klaus Berberich and Dr. Pauli Miettinen. *HITS*. URL: http://resources.mpi-inf.mpg.de/departments/d5/teaching/ws13_14/irdm/slides/irdm-4-3.pdf. (date accessed: 26.11.2019).

[3] Ulrik Brandes and Thomas Erlebach (Eds.). *Network Analysis - Methodological Foundations*. Springer-Verlag, 2005.

[4] Sreenivas Gollapudi, Marc Najork, and Rina Panigrahy. *Using Bloom Filters to Speed Up HITS-like Ranking Algorithms*. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/waw2007.pdf. (date accessed: 30.10.2019).

[5] Jon M. Kleinberg. *Authoritative Sources in a Hyperlinked Environment*. URL: http://www.cs.cornell.edu/home/kleinber/auth.pdf. (date accessed: 23.10.2019).

[6] R. Lempel and S. Moran. *The Stochastic Approach for Link-Structure Analysis (SALSA) and the TKC Effect*. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.38.5859&rep=rep1&type=pdf. (date accessed: 28.11.2019).

[7] Xinyue Liu, Hongfei Lin, and Cong Zhang. *An Improved HITS Algorithm Based on Pagequery Similarity and Page Popularity*. URL: https://pdfs.semanticscholar.org/e7e1/82659614da4f92daca8d8455fd11350f198a.pdf. (date accessed: 31.10.2019).

[8] Punit Patel and Kanu Patel. *A Review of PageRank and HITS Algorithms*. URL: http://ijarest.com/papers/finished_papers/A%20Review%20of%20PageRank%20and%20HITS%20Algorith.pdf. (date accessed: 24.10.2019).

[9] Raluca Remus. *Lecture #4: HITS Algorithm - Hubs and Authorities on the Internet*. URL: http://pi.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture4/lecture4.html. (date accessed: 24.10.2019).

[10] Raluca Remus. *Lecture 3: PageRank Algorithm - The Mathematics of Google Search*. URL: http://pi.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture3/lecture3.html. (date accessed: 20.12.2019).

[11] Wikipedia. *Bloom filter*. URL: https://en.wikipedia.org/wiki/Bloom_filter. (date accessed: 30.10.2019).

[12] Wikipedia. *CLEVER project*. URL: https://en.wikipedia.org/wiki/CLEVER_project. (date accessed: 26.11.2019).

[13] Wikipedia. *PageRank*. URL: https://en.wikipedia.org/wiki/PageRank. (date accessed: 21.12.2019).

[14] Wikipedia. *Perron–Frobenius theorem*. URL: https://en.wikipedia.org/wiki/Perron-Frobenius_theorem. (date accessed: 24.10.2019).

[15] Wikipedia. *Power Iteration*. URL: https://en.wikipedia.org/wiki/Power_iteration. (date accessed: 30.10.2019).

[16] Wikipedia. *SALSA algorithm*. URL: https://en.wikipedia.org/wiki/SALSA_algorithm. (date accessed: 28.11.2019).

[17] Yanchun Zhang, Jeffrey Xu Yu, and Jingyu Hou. *Web Communities: Analysis and Construction*. Springer-Verlag, 2006.