IEICE TRANS. INF. & SYST., VOL.E91–D, NO.2 FEBRUARY 2008

PAPER Special Section on Foundations of Computer Science

Improvements of HITS Algorithms for Spam Links

Yasuhito ASANO[†a)], ***Member***, **Yu TEZUKA**[††], ***Nonmember***, ***and*** **Takao NISHIZEKI**[†††], ***Fellow***

SUMMARY The HITS algorithm proposed by Kleinberg is one of the representative methods of scoring Web pages by using hyperlinks. In the days when the algorithm was proposed, most of the pages given high score by the algorithm were really related to a given topic, and hence the algorithm could be used to find related pages. However, the algorithm and the variants including Bharat's improved HITS, abbreviated to BHITS, proposed by Bharat and Henzinger cannot be used to find related pages any more on today's Web, due to an increase of spam links. In this paper, we first propose three methods to find "linkfarms," that is, sets of spam links forming a densely connected subgraph of a Web graph. We then present an algorithm, called a trust-score algorithm, to give high scores to pages which are not spam pages with a high probability. Combining the three methods and the trust-score algorithm with BHITS, we obtain several variants of the HITS algorithm. We ascertain by experiments that one of them, named TaN+BHITS using the trust-score algorithm and the method of finding linkfarms by employing name servers, is most suitable for finding related pages on today's Web. Our algorithms take time and memory no more than those required by the original HITS algorithm, and can be executed on a PC with a small amount of main memory.

key words: *scoring algorithm, Web pages, HITS, BHITS, PageRank, search engine, Web graph, spam page, spam links*

1. Introduction

Search engines are widely used as a tool for obtaining information on a topic from the Web [1], [10], [20]. Given keywords specifying a topic, search engines score Web pages containing the keywords by using several scoring algorithms, and output the pages in decending order of the score. For example, PageRank proposed by Brin and Page [5] has been used as a scoring algorithm by Google [10]. Another scoring algorithm "Hyperlink Induced Topic Search," abbreviated to HITS and proposed by Kleinberg [13], has the following three significant features (1)-(3).

(1) The HITS algorithm gives high scores to pages related to the topic specified by given keywords even if the pages do not contain the keywords. On the other hand, most of the search engines output only pages containing given keywords. Thus, the HITS algorithm can find some of the pages which cannot be found by other search engines.

(2) The HITS algorithm can be executed on a PC with a small amount of main memory, because it needs data of a quite small number of pages, compared with the PageRank algorithm and most of the scoring algorithms used by search engines.

(3) The HITS algorithm can be executed on demand, because it needs data small enough to be collected through the network on demand. On the other hand, the PageRank algorithm takes several weeks to collect data, and hence cannot be executed on demand.

The HITS algorithm worked well in the days when it was proposed. Several HITS-based algorithms have been proposed since [2], [14], [15], [17]. However, the original HITS algorithm and the HITS-based algorithms no longer work well on today's Web due to an increase of spam links. Several methods of finding spam links have recently been developed [7]–[9], but they require too large data of pages to be executed on demand with a PC. For example, the methods proposed by Fetterly *et al.* [8], [9] require the data of the contents of pages, which are much larger than the data of the links of the pages used by the HITS algorithm.

In this paper, we first propose three methods to find linkfarms by using network information; a *linkfarm* is a set of spam links which form a densely connected subgraph of a Web graph; a *Web graph* is a directed graph whose vertex set is a set of Web pages, and whose edge set is a set of links between pages. Our methods find more linkfarms than the method proposed by Wu and Davison [19]. We then propose a *trust-score* algorithm to give high scores to pages which are not spam pages with a high probability, by extending the ideas used by the TrustRank algorithm [11]. We then construct four scoring algorithms; the first one is obtained by combining our trust-score algorithm with BHITS algorithm proposed by Bharat and Henzinger [3]; the remaining three are obtained by combining each of our three methods of finding linkfarms with the trust-score algorithm and BHITS. We finally evaluate our algorithms and several HITS-based algorithms by experiments.

In order to evaluate various scoring algorithms, we use the "quality of top ten authorities" found by an algorithm for a given topic; the *top ten authorities* are pages of the top ten high score given by the algorithm, the *quality* of top ten authorities is measured by the number of pages related to the topic among the top ten authorities, and hence the quality of

Manuscript received January 25, 2007.
Manuscript revised June 3, 2007.

[†]The author is with the Department of Information Sciences, Faculty of Science and Engineering, Tokyo Denki University, Saitama-ken, 350–0394 Japan.
[††]The author is with Mobile Entertainment Category, Victor Company of Japan, Limited, Maebashi-shi, 371–8543 Japan.
[†††]The author is with the Graduate School of Information Sciences, Tohoku University, Sendai-shi, 980–8579 Japan.
a) E-mail: asano@j.dendai.ac.jp

DOI: 10.1093/ietisy/e91–d.2.200

top ten authorities is at most ten. We examine the quality of top ten authorities by computational experiments using fourteen topics. For almost all the topics, our algorithms find top ten authorities of higher quality than those found by the existing algorithms. Particularly, one of our algorithms, called TaN+BHITS and abbreviated to TaN, employing the trust-score algorithm and a method of finding linkfarm by using name servers, finds top ten authorities of the best average quality 8.79, while the existing algorithms find top ten authorities of average quality at most 3.07 (see Table 1 in Sect. 4).

Our TaN+BHITS algorithm can be used to find pages related to a given topic on today's Web; most of the pages given high score by the algorithm are truly related to a given topic for almost all the topics used in our experiments. Our four algorithms including TaN+BHITS require no data of pages other than the data collected by the original HITS algorithm, and hence can be executed on demand for a given topic on a PC with a small amount of main memory.

The rest of this paper is organized as follows. In Sect. 2, we outline the original HITS algorithm proposed by Kleinberg [13], and introduce related works. In Sect. 3, we first propose three methods of finding linkfarms, then propose the trust-score algorithm, and finally obtain four improved variants of the HITS algorithm by combining the BHITS algortihm with the three methods of finding linkfarms and the trust-score algorithm. In Sect. 4, we analyze the results obtained by computational experiments. In Sect. 5, we present our concluding remarks. A preliminary version of the paper was presented in [2].

2. Preliminaries

We first define a host name and a domain name in Sect. 2.1, then present the original HITS algorithm in Sect. 2.2, and finally outline the BHITS algorithm proposed by Bharat and Henzinger in Sect. 2.3.

2.1 Terms

Since a term "host name" is sometimes confused with a term "domain name," we use the following definitions throughout the paper.

The *host name* of a Web page p is the name of the host containing p. As a host name of p, we use a substring of p's URI between `http://` and the next slash symbol. For example, if a page p has URI `http://www.geocities.jp/ken/index.html`, then the host name of p is `www.geocities.jp`.

Let $domlevel(p)$ be one plus the number of dot symbols in the host name of page p. Thus, $domlevel(p) = 3$ for the page p above. Divide a host name by dot symbols into a number $domlevel(p)$ of substrings, then the i-th substring from the right is called the *i-th level domain*. For example, if the host name of a page p is `www.geocities.jp`, then `geocities` is the second level domain of p. We say that two pages p and q have the *same domain name* if either p and q have the same host name or $domlevel(p) = domlevel(q) \geq 3$ and the i-th level domain of p is equal to the i-th level domain of q for each i, $1 \leq i \leq domlevel(p) - 1$. For example, if page p has URI `http://news.www.infoseek.co.jp/` and page q has URI `http://music.www.infoseek.co.jp/`, then p and q have the same domain name, because $domlevel(p) = domlevel(q) = 5$ and p and q have the same first, second, third, and fourth level domains, `jp`, `co`, `infoseek`, and `www`, respectively. On the other hand, if page p has URI `http://ask.jp` and page q has URI `http://slashdot.jp`, then p and q do not have the same domain, because they do not have the same host name and $domlevel(p) = domlevel(q) = 2$.

2.2 Original HITS Algorithm

The HITS algorithm proposed by Kleinberg [13] finds *authorities* and *hubs* for a topic specified by given keywords. The algorithm regards a page linked from many pages as an *authority*, and regards a page having links to many authorities as a *hub*. More precisely, the algorithm computes an "authority score" and a "hub score" of each page, as outlined as follows.

The original HITS algorithm

1. Let the *root set* R be a set of top x pages of the result of some search engine for given keywords, where x is an integer parameter. Kleinberg [13] usually set $x = 200$.
2. Construct a Web graph $G = (V, E)$, where the vertex set V is a union of R and the set of all pages that either link to or are linked from pages in R, and the edge set E consists of all the links between pages in V except links between pages having the same hostname.
3. For each vertex $v_i \in V$, $1 \leq i \leq |V|$, set an *authority score* a_i to 1 and set a *hub score* h_i to 1.
4. Repeat the following procedures (a)-(c) ℓ times, where ℓ is a given integer parameter.
   (a) For each $v_i \in V$, $a_i := \sum_{(v_j, v_i) \in E} h_j$;
   (b) For each $v_i \in V$, $h_i := \sum_{(v_i, v_j) \in E} a_j$; and
   (c) Normalize vectors $\mathbf{a} = (a_1, a_2, \ldots, a_{|V|})$ and $\mathbf{h} = (h_1, h_2, \ldots, h_{|V|})$ in the L_2-norm so that $\sum_{i=1}^{|V|} a_i = 1$ and $\sum_{i=1}^{|V|} h_i = 1$.
5. Output vectors $\mathbf{a}$ and $\mathbf{h}$.

Kleinberg usually set $\ell = 50$, because the vectors $\mathbf{a}$ and $\mathbf{h}$ almost converge after repeating (a)-(c) in Step 4 fifty times.

Throughout the paper, the "top ten authorities" found by a HITS-based algorithm mean the ten pages of the heighest authority score among the pages found by the algorithm, and we measure the "quality" of the top ten authorities by the number of pages related to a given topic among the top ten authorities. For example, if the top ten authorities found by a HITS-based algorithm contain seven pages related to a given topic, then the quality of the top ten authorities

is seven. Similar measures have been used by several researchers on HITS-based algorithms [3], [15].

The original HITS algorithm could find top ten authorities of practically good quality and could be used to find pages related to a given topic in the days when it was proposed. Subsequently, several researchers pointed out a *mutual reinforcing* problem and a *topic drift* problem of the original HITS, and proposed various HITS-based algorithms with effective solutions to these problems [3], [4], [6], [12], [14]–[18], [21]. We will describe the mutual reinforcing problem in Sect. 2.3. The original HITS algorithm sometimes incorrectly gives high scores to pages not related to a given topic. This is called a topic drift problem.

The number of pages on the Web has been exponentially increasing since the Web was born, and the so-called spam links have been increasing, too. On today's Web, the HITS algorithm and the variants cannot find top ten authorities of good quality any more due to the increase of spam links.

Some authors [7]–[9] define a spam link as follows: a link from a page p to a page q is a *spam link* if the content of q is irrelevant to the content of p and the link is created to force a scoring method to give a high score to the site containing the page p. It is difficult to find spam links according to this definition, and hence the authors try to find spam links approximately by using heuristics. For example, Fetterly *et al.* [8], [9] proposed several heuristics to find spam links. The heuristics use the content information of pages, although the original HITS uses only the link information of pages. Costa Carvalho *et al.* [7] proposed several heuristics to find spam links by using characteristic graph structures of *inter-host links*, each between a page in a host and a page in another host. All these heuristics require data of many pages other than the pages used by the original HITS. Thus, the existing methods of finding spam links require much larger data than those required by the original HITS, and hence they are not suitable for our objective of establishing a scoring algorithm which can be executed on demand with a PC.

2.3 BHITS Algorithm

The orignal HITS algorithm suffers from the following problem. If a malicious person creates a host containing a number of pages linking to the same page p in another site, then the authority score of the page p becomes unfairly high. One can easily create such a host. Thus, the top ten authorities found by the original HITS algorithm could be made unreliable by a malicious person. This problem is called a *mutually reinforcing problem*.

Slightly modifying the original HITS algorithm, Bharat and Henzinger [3] proposed the BHITS algorithm, which almost resolves the mutually reinforcing problem. We now introduce several notations to explain their modifications. For each vertex v_i in the base set V used in the original HITS, let $\Gamma^-(v_i)$ be the set of vertices $\{v_j \in V \mid (v_j, v_i) \in E\}$, and let $\Gamma^+(v_i)$ be the set of vertices $\{v_j \in V \mid (v_i, v_j) \in E\}$. Let $H^-(v_i)$ (or $H^+(v_i)$) be the set of all hosts containing pages corresponding to vertices in $\Gamma^-(v_i)$ (or $\Gamma^+(v_i)$, respectively). For each host $host_k \in H^-(v_i)$, $1 \le k \le |H^-(v_i)|$, let $r^-(host_k, v_i)$ be the number of links going from pages in $host_k$ to the page v_i. Similarly, for each host $host_k \in H^+(v_i)$, $1 \le k \le |H^+(v_i)|$, let $r^+(host_k, v_i)$ be the number of links going from page v_i to those in $host_k$. The BHITS algorithm replaces (a) and (b) in Step 4 of the original HITS algorithm by the following (a)′ and (b)′, respectively.

(a)′ For each $v_i \in V$, $a_i := \sum_{(v_j,v_i)\in E} \frac{h_j}{r^-(host(v_j),v_i)}$.

(b)′ For each $v_i \in V$, $h_i := \sum_{(v_i,v_j)\in E} \frac{a_j}{r^+(host(v_j),v_i)}$.

Thus, even if a number of pages in a host link to the same page p, the authority score of the page p computed by the BHITS algorithm does not become unfairly high, and hence the mutually reinforcing problem is almost resolved.

For most of the topics used in their experiments, the BHITS algorithm found top ten authorities of better quality than those obtained by the original HITS algorithm [3]. We will hence construct several variants of the BHITS algorithm instead of the original HITS algorithm.

3. Improvements

In Sect. 3.1, we first propose three methods to find linkfarms, then in Sect. 3.2 we propose a trust-score algorithm, and finally in Sect. 3.3 we construct four new scoring algorithms by combining the BHITS algorithm with the three methods of finding linkfarms and the trust-score algorithm.

3.1 Removing Linkfarms

Wu and Davison [19] define a linkfarm as a set of spam links which form a densely connected subgraph on a Web graph, and say that a page belongs to a linkfarm if the page is an end of a spam link in the linkfarm. The HITS algorithm gives high authority scores and high hub scores to pages belonging to a densely connected subgraph, and hence pages belonging to a linkfarm would get high authority score and high hub score. Consequently, the top ten authorities obtained by the HITS algorithm would contain a number of pages belonging to the linkfarm.

Wu and Davison proposed an algorithm for finding linkfarms, and evaluated how effective their algorithm is for improving scoring algorithms. For the evaluation, they used a simple scoring algorithm which gives each page a score equal to the number of links entering to the page. For most of the topics used in their experiments, the simple scoring algorithm could obtain top ten authorities of good quality if it ignores all the links in the linkfarms found by their algorithm. Thus, their algorithm is effective for improving the simple scoring algorithm. However, they did not confirm whether their algorithm is effective or not for improving the HITS-based algorithms, which are more sophisticated than the simple scoring algorithm.

We evaluate how effective the algorithm of Wu and Davison is for improving the BHITS algorithm. Let **LF+BHITS** be the BHITS algorithm which ignores all the links in the linkfarms found by the algorithm of Wu and Davison. We will evaluate the quality of top ten authorities found by LF+BHITS in Sect. 4.

We discover that a typical linkfarm consists of pages sharing some kinds of network information, such as a host name, a domain name, an IP address, and a name server. (Both the IP address of the host containing a given page and the name of the name server used by the host can be easily obtained by several methods including `nslookup` UNIX command.) For example, there is a linkfarm consisting of the following five pages sharing the same domain name. (A domain name is defined in Sect. 2.)

- http://news.www.infoseek.co.jp/
- http://music.www.infoseek.co.jp/
- http://movie.www.infoseek.co.jp/
- http://health.www.infoseek.co.jp/
- http://map.www.infoseek.co.jp/

There is another linkfarm consisting of the following five pages sharing the same IP address.

- http://niwatori.bufsiz.jp/shop18.htm
- http://healthnet.client.jp/
- http://tsutaya.healthnet.client.jp/
- http://onkyo.aidol.nobody.jp/
- http://boople.kazokukai.gozaru.jp/

All the pages in each of the two linkfarms above share the same name server, too.

The original HITS algorithm ignores every link between two pages belonging to the same host, but it still suffers from a number of linkfarms. Authors of Refs. [7]–[9] mentioned that a set of spam pages created by a marlicious person frequently shares some kinds of network information even if the pages do not have the same host name. Investigating a number of pages sharing the same host name, domain name, or IP address, we found that they usually share the same name server, too.

We thus propose an algorithm Nameserver+BHITS, abbreviated to **N+BHITS**, which is the BHITS algorithm with the following two modifications:

(1) In Step 2 of the original HITS algorithm, remove every link between two pages sharing the same name server from the Web graph G; and

(2) Use a name server instead of a host name in Step 4 (a)′ and (b)′ of the BHITS algorithm.

Let Algorithm IP+BHITS, abbreviated to **I+BHITS**, be the same as Algorithm N+BHITS except that it uses an IP address instead of a name server. Similarly, let Algorithm Domain+BHITS, abbreviated to **D+BHITS**, be the same as N+BHITS except that it uses a domain name instead of a name server. It is not new ideas to ignore links between two pages sharing the same IP address or domain name [21], but we compare effectiveness of these ideas with that of the method proposed by Wu and Davison. We will evaluate the quality of top ten authorities found by N+BHITS, I+BHITS, and D+BHITS in Sect. 4.

3.2 Trust-Score

Gyongyi *et al.* [11] proposed the TrustRank algorithm which approximately finds *reliable pages*, which are not created with malicious motives. The algorithm gives a *reliability score* to a page on the Web. The main idea used by the algorithm is to note that "a page linked from reliable pages would be reliable." This idea is similar to the main idea used by the PageRank algorithm [5]: "a page linked from important pages would be important."

The TrustRank algorithm requires a huge number of pages on the Web similarly as the PageRank algorithm, and it cannot be directly used to score the pages on a small Web graph used by the HITS algorithm. Thus, we cannot directly use the TrustRank algorithm for improving the HITS algorithm.

We propose an algorithm to give a *trust-score* to every page contained in the base set used by HITS. Employing the ideas of the HITS algorithm and the TrustRank algorithm, we say that a page is reliable if it has links to many reliable pages related to a given topic, and that it would be a good hub page for the topic. For most of the topics used in our preliminary experiments, we found that more than half of the pages in the root set are reliable and related to the given topic. Our trust-score algorithm thus regards a page u as a *reliable hub page* if the page u links to many pages in the root set, and regards a page v as a *reliable authority page* if the page v is linked from many reliable hub pages. More precisely, if there are two or more hosts, each containing a page which belongs to the root set and is linked from a page u, then the trust-score algorithm gives u a *trust hub score* $T_{hub}(u)$ equal to the number of these hosts; otherwise, the algorithm gives u a trust hub score $T_{hub}(u) = 0$. The algorithm also gives a page v a *trust authority score* $T_{auth}(v)$ equal to the sum of trust hub scores (divided by $|H^+(u)|$) of the pages u linking to v. The *trust-score* of v is its trust authority score normalized by the sum of all trust authority scores. The trust hub score of a page u is a measure to indicate how reliable u is as a hub page, and the trust-score of a page v is a measure to indicate how reliable v is as an authority page. Thus our trust-score algorithm is as follows.

Trust-Score Algorithm

Let R, V and E be the root set, the vertex set and the edge set, respectively, used in the HITS algorithm.

1. For each page $u \in V$, let $\{s_1, s_2, \ldots, s_k\}$ be the set of all pages that are linked from u and belong to the root set R, that is, $(u, s_i) \in E$ and $s_i \in R$, $1 \le i \le k$. Let $host(u)$ be the number of distinct host names of $s_1, s_2, \ldots, s_k$. If $host(u) \ge 2$, then set a trust hub score $T_{hub}(u) = host(u)$. Otherwise, set $T_{hub}(u) = 0$.

2. For each page $v \in V$, set a trust authority score $T_{auth}(v)$ to $\sum_{(u,v)\in E} \frac{T_{hub}(u)}{|H^+(u)|}$.
3. For each page $v \in V$, output $\frac{T_{auth}(v)}{\sum_{w\in V} T_{auth}(w)}$ as the trust-score of page v.

3.3 Our Four Scoring Algorithms

We propose the following four algorithms using the trust-score algorithm. Let TrustScore+BHITS, abbreviated to **T+BHITS**, be the algorithm which gives a page p a score $t(p) + a(p)$, where $t(p)$ is the trust-score of the page p, and $a(p)$ is the authority score of p computed by BHITS. Similarly, let **TaI+BHITS**, **TaD+BHITS**, and **TaN+BHITS** be the algorithms which give a page p a score $t(p) + a(p)$, where $a(p)$ is the authority score of p computed by I+BHITS, D+BHITS, and N+BHITS, respectively. TaI+BHITS, TaD+BHITS and TaN+BHITS are abbreviations of "TrustScore and IP + BHITS," "TrustScore and Domain + BHITS" and "TrustScore and Nameserver + BHITS," respectively. Using two weight factors x and y, one can construct an algorithm which gives a page p a score $x \cdot t(p) + y \cdot a(p)$. We have made some preliminary experiments varying the weights x and y, and confirmed that, for most of the used topics, top ten authorities of the best quality are obtained when $x = y = 1$. We will evaluate the quality of top ten authorities found by T+BHITS, TaI+BHITS, TaD+BHITS, and TaN+BHITS in Sect. 4.

4. Experimental Results

Table 1 depicts the results of our experiments conducted from December 2005 through January 2006 for the following eleven HITS-based algorithms. The first three are existing HITS-based algorithms: HITS, BHITS, and WBHITS proposed by Li *et al.* [15]; in Table 1 they are abbreviated as H, B and WB, respectively. The next four algorithms LF+BHITS, D+BHITS, I+BHITS and N+BHITS, abbreviated as LB, DB, IB and NB respectively, are presented in Sect. 3.1, and are variants of the BHITS algorithm using methods of finding linkfarms; LF+BHITS uses Wu and Davison's method of finding linkfarms, D+BHITS uses a domain name, I+BHITS uses an IP address, and N+BHITS uses a name server. The last four algorithms T+BHITS, TaD+BHITS, TaI+BHITS and TaN+BHITS, abbreviated as T, TaD, TaI and TaN respectively, are our algorithms using the trust-score in Sect. 3.2; T+BHITS combines the trust-score algorithm with BHITS, and similarly, the three algorithms TaD+BHITS, TaI+BHITS and TaN+BHITS combine the trust-score algorithm with D+BHITS, I+BHITS and N+BHITS, respectively.

Each of the given fourteen topics Topic 1, Topic 2, ..., Topic 14 is specified by one of the following fourteen keywords: "Canoe," "Cheese," "F1 (Formula One)," "Gardening," "Iriomote-Cat (one of the endangered species)," "Kabuki (Japanese traditional performance)," "Monochrome-Photograph," "Movie," "Olympic," "Pipe-Organ," "Railway," "Rock-Climbing," "Tennis," and "Wine." For each topic, the root set used in our experiments consists of top two hundred pages of the result of Google for the topic. HITS-based algorithms including our algorithms can receive multiple keywords as an input as well as a single keyword. The results for other topics represented by a single or multiple keywords are similar to those for the topics used in the experiments, and the results obtained by using search engines other than Google are also similar to those obtained by using Google.

Each cell in the rows from "Topic 1" to "Topic 14" of

Table 1 The experimental results.

	Existing Algorithms			Variants of BHITS				Ours			
Algorithm	**H**	**B**	**WB**	**LB**	**DB**	**IB**	**NB**	**T**	**TaD**	**TaI**	**TaN**
Topic 1	0	7	9	7	8	8	8	10	10	10	10
Topic 2	0	0	0	0	0	0	0	10	0	0	0
Topic 3	0	0	0	0	0	10	10	10	10	10	10
Topic 4	0	0	0	0	0	0	0	7	9	5	9
Topic 5	0	5	10	5	10	10	10	7	10	10	10
Topic 6	0	1	1	1	1	1	1	10	9	10	9
Topic 7	0	2	2	2	10	10	10	6	9	9	9
Topic 8	4	10	10	10	10	2	2	10	10	9	9
Topic 9	10	0	0	0	10	10	10	8	10	10	10
Topic 10	0	0	1	0	10	0	10	7	10	7	10
Topic 11	10	10	10	10	10	10	10	10	10	10	10
Topic 12	0	0	0	0	0	0	10	2	5	2	10
Topic 13	0	0	0	10	0	10	10	3	3	10	10
Topic 14	0	0	0	5	0	0	6	8	7	5	7
Average	**1.71**	**2.50**	**3.07**	**3.57**	**4.93**	**5.07**	**6.93**	**7.71**	**8.00**	**7.64**	**8.79**
Sufficient	**2**	**2**	**4**	**3**	**6**	**6**	**8**	**6**	**10**	**9**	**12**
Non-Root	**10**	**16**	**12**	**21**	**30**	**19**	**39**	**13**	**20**	**17**	**25**
Non-Google	**9.36**	**9.43**	**9.14**	**9.14**	**9.14**	**8.36**	**8.07**	**5.86**	**6.43**	**6.29**	**5.93**

H: HITS, B: BHITS, WB: WBHITS, LB: LF+BHITS, DB: D+BHITS, IB: I+BHITS, NB: N+BHITS, T: T+BHITS, TaD: TaD+BHITS, TaI: TaI+BHITS, TaN: TaN+BHITS.

Table 1 shows the quality of the top ten authorities found by each algorithm for the given topic. As we have already described, the quality of top ten authorities is defined as the number of pages which are related to the given topic and belong to the top ten authorities found by each algorithm. It is judged by manual inspection of human subjects whether a page is truly related to a given topic. Many authors use similar methods to evaluate HITS-based methods [3], [4], [6], [12], [14]–[18]. For each algorithm, "Average" row in Table 1 shows the average quality of the top ten authorities obtained for the fourteen topics. For each algorithm, "Sufficient" row shows the number of topics for which the obtained top ten authorities have sufficient quality; we say that the quality of top ten authorities is *sufficient* if the top ten authorities contain only at most one page not related to the given topic. For example, the original HITS finds top ten authorities of sufficient quality for only two topics among the fourteen topics, and our TaN+BHITS finds top ten authorities of sufficient quality for twelve topics.

The original HITS algorithm and the BHITS algorithm find top ten authorities containing few pages related to the given topics; the top ten authorities found by HITS contain no related pages for eleven topics among the fourteen topics, and the top ten authorities found by BHITS contain no related pages for eight topics. It was reported that the WBHITS algorithm is one of the best HITS-based algorithms [15], but WBHITS finds top ten authorities containing no related pages for seven topics among the fourteen topics. These three algorithms cannot be used for finding related pages on today's Web.

We say that two hosts $host_1$ and $host_2$ are *mutually linked* if there are both a link from a page in $host_1$ to a page in $host_2$ and a link from a page in $host_2$ to a page in $host_1$, and say that a link from a page in a host to a page in another host is a *mutual inter-host link* if these hosts are mutually linked. For four topics "F1 (Topic 3)," "Olympic (Topic 9)," "Pipe-Organ (Topic 10)," and "Rock-Climbing (Topic 12)," we found a number of linkfarms, each containing few mutual inter-host links. These linkfarms cannot be found by LF+BHITS using Wu and Davison's method of finding linkfarms, because their method can find only linkfarms containing a number of mutual inter-host links [19]. On the other hand, each of the linkfarms consists of links between pages sharing some kind of network information, and hence at least one of our algorithms D+BHITS, I+BHITS, and N+BHITS finds such linkfarms. This is the reason why our proposed algorithms D+BHITS, I+BHITS, and N+BHITS find top ten authorities of better quality than those found by LF+BHITS. Thus, our proposed methods of finding linkfarms are more effective for improving the BHITS algorithm than Wu and Davison's method. For most of the fourteen topics, N+BHITS finds top ten authorities of better quality than those found by D+BHITS or I+BHITS, and hence the method of finding linkfarms by utilizing the name server is better than the other two methods for improving the BHITS algorithm.

For most of the fourteen topics, our four algorithms using the trust-score algorithm, T+BHITS, TaD+BHITS, TaI+BHITS and TaN+BHITS, find top ten authorities of much better quality than those found by the other algorithms, hence the trust-score algorithm is effective for improving the BHITS algorithm. Particularly, for most of the topics, TaN+BHITS obtains top ten authorities of better quality than those obtained by the other algorithms.

The average quality 8.79 of the top ten authorities found by TaN+BHITS is significantly better than the average quality 1.71, 2.50, and 3.07 of the top ten authorities found by HITS, BHITS, and WBHITS, respectively. TaN+BHITS also finds top ten authorities of sufficient quality for most of the topics, twelve topics among the fourteen, although the existing HITS-based algorithms, HITS, BHITS and WBHITS, find top ten authorities of sufficient quality for very few topics. For each of the four topics, "Kabuki (Topic 6)," "Olympic (Topic 9)," "Pipe-Organ (Topic 10)," "Tennis (Topic 13)," and "Rock-Climbing (Topic 12)," the top ten results of Google contain some pages not related to the topic, but all the pages in the top ten authorities found by TaN+BHITS are related to the topic.

All the eleven algorithms use a root set consisting of top two hundred pages given high scores by Google, and Google would use some techniques to cope with spam links like linkfarms. Therefore, the root set would have fewer spam links than a set of two hundred pages, containing a given keyword and arbitrarily chosen from the Web. However, the TaN+BHITS algorithm finds top ten authorities containing many pages related to the given topics although the original HITS algorithm finds top ten authorities containing few related pages, as described above. Hence, the proposed method of finding linkfarms using a name server and the trust-score algorithm are very effective to obtain top ten authorities with good quality.

The HITS algorithm finds some of the related pages which cannot be found by Google, as we described in Sect. 1. We now evaluate how many such pages the eleven algorithms find. For each algorithm, "Non-Root" row of Table 1 denotes the total number of pages such that

(1) they are related to the given topic;

(2) they belong to the top ten authorities found by the algorithm; and

(3) they do not belong to the root set.

If an algorithm has a large value in the row, then the algorithm finds a number of related pages which could not be found by Google, because the root set consists of the top two hundred pages obtained by Google.

One can observe the following three facts (a)-(c):

(a) The algorithms using our methods of finding linkfarms, D+BHITS, I+BHITS, N+BHITS, TaD+BHITS, TaI+BHITS, and TaN+BHITS, have a larger value in the row than the existing algorithms, HITS, BHITS, and WBHITS;

(b) D+BHITS has a larger value in the row than TaD+BHITS which is the algorithm obtained by combining D+BHITS with the trust-score algorithm. Similarly, I+BHITS (or N+BHITS) has a larger value than TaI+BHITS (or TaN+BHITS) which is the algorithm combining I+BHITS (or N+BHITS, respectively) with the trust-score algorithm.

(c) TaN+BHITS has a larger value in the row than the other algorithms using the trust-score.

The fact (a) implies that our methods of finding linkfarms are effective for finding related pages which cannot be found by Google. The fact (b) can be explained as follows: the trust-score algorithm frequently gives high scores to pages belonging to the root set, and hence all the algorithms using the trust-score frequently find top ten authorities containing a number of pages belonging to the root set; consequently these top ten authorities contain few pages which do not belong to the root set; hence TaD+BHITS (or TaI+BHITS, TaN+BHITS) using the trust-score has a smaller value in the row than D+BHITS (or I+BHITS, N+BHITS, respectively) without the trust-score. As we described above, the algorithms with the trust-score find top ten authorities of better quality than those found by the algorithms without the trust-score, and thus the fact (b) implies that there is a trade-off between the quality of top ten authorities and the value in the row. The quality of top ten authorities is generally more important than the value in the row, and hence the fact (c) implies that TaN+BHITS is the best choice for finding both top ten authorities of sufficient quality and a number of related pages which cannot be found by Google.

For each algorithm, the bottom row "Non-Google" of Table 1 denotes the average number of pages such that they belong to the top ten authorities found by the algorithm but do not belong to the top ten pages obtained by Google. For example, the top ten authorities found by TaN+BHITS algorithm contain 5.93 pages on average which do not belong to the top ten pages obtained by Google. One can observe from the row that our algorithms T, TaD, TaI and TaN find the top ten authorities such that five or six of them are not ranked as the top ten pages by Google. Hence our algorithms are useful for finding related pages different from the results of Google.

We now discuss some remaining problems of our HITS-based algorithms. For topic "Cheese (Topic 2)," all the algorithms except T+BHITS find top ten authorities containing no related pages. On the Web graph used by T+BHITS for Topic 2, there are several links emanating from pages with large trust hub scores and entering to some related pages in the top ten authorities, and hence these related pages are given high trust scores. Most of these links join pages sharing some kind of network information other than a host name, such as a domain name, an IP address, and a name server. These links are ignored by D+BHITS, I+BHITS, N+BHITS, TaD+BHITS, TaI+BHITS, or TaN+BHITS, and hence the top ten authorities found by these algorithms do not contain such related pages. In other words, these algorithms incorrectly regard the links as linkfarms and remove them. It is one of the remaining problems to distinguish such links from actual linkfarms.

Our methods of finding linkfarms might create another problem. There is a number of blog sites or BBS sites sharing the same host, domain, IP address, or name server, and all the links between these sites would be regarded as a linkfarm by our algorithms. If a blog site is linked from a number of sites, then the blog site should be given a high authority score according to the main idea of HITS. However, if the blog site is linked from a number of blog sites sharing the same name server but is not linked from a number of other sites, then all the links between these blog sites are regarded as a linkfarm by N+BHITS and TaN+BHITS; hence, there might arise a problem that the blog site would not be given a high authority score by N+BHITS and TaN+BHITS. In order to correctly regard such a blog site as an authority, one needs a method of classifying links between pages sharing the same name server into spam links and non-spam links; obtaining such a method would be one of the future works.

For each of the fourteen topics in Table 1, all the algorithms use the same vertex set. Table 2 depicts the numbers of vertices used by the algorithms for the fourteen topics. On the other hand, each of the algorithms uses an edge set slightly different from the edge sets used by other algorithms, because each algorithm ignores several links in its own manner. The original HITS algorithm, BHITS, WBHITS, and T+BHITS use the same edge set for each topic, because all of them ignores only links between pages having the same host name. The numbers of edges used by these four algorithms are shown in column "Edge(h)" in Table 2. Similarly, each of the three pairs of algorithms (D+BHITS, TaD+BHITS), (I+BHITS, TaI+BHITS), and (N+BHITS, TaN+BHITS) uses the same edge set, which excludes links between pages having the same domain name, IP address, and name server, respectively. The numbers of edges used by these three pairs of algorithms are shown in columns "Edges(d)," "Edges(i)," and "Edges(n)," respectively.

Table 2 The numbers of vertices and edges.

Topic	Vertices	Edges(h)	Edges(d)	Edges(i)	Edges(n)
1	6319	16892	15609	15891	12437
2	6706	91914	78596	82618	65589
3	9313	105929	69253	74911	53382
4	10652	156082	114086	123768	83819
5	1920	5602	4227	4493	4268
6	5193	19395	13889	18122	13561
7	5682	88228	30116	33541	25536
8	12964	72070	52860	65971	45717
9	6904	19742	15030	16677	14729
10	2630	14445	7012	9877	7044
11	8298	15928	14148	14943	13881
12	3127	16485	7139	9335	6590
13	6202	30314	24625	28736	25172
14	8924	108364	89634	98552	88660

The amount of memory used by each of our proposed algorithms is almost equal to that used by the original HITS algorithm. Most part of the running time of the HITS-based algorithms is spent to collect data of pages through the Internet. Every algorithm uses data of the same set of pages, and hence the running time of our propsed algorithms are almost equal to that of the original HITS algorithms. In our experiments, all the algorithms output top ten authorities in a few seconds once the required data is collected through the Internet. The algorithms use about 10 MB of memory for the Topic 8, which is larger than the amount of memory used for any of other topics. All the algorithms are implemented with C++ and executed on a PC with Pentium4 1.8 GHz and 512 MB main memory. Our scoring algorithm TaN+BHITS can find top ten authorities of good quality on today's Web and can be executed on demand with a PC. The algorithm particularly finds top ten authorities of the best quality for most of the used topics.

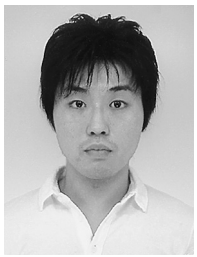
5. Concluding Remarks

We have proposed several improved variants of the HITS algorithm, by proposing the trust-score algorithm and three methods of finding linkfarms. One of our algorithms, named TaN+BHITS, finds top ten authorities of much better quality than those found by the existing algorithms, and the top ten authorities found by TaN+BHITS contain a number of related pages which cannot be found by Google. We have hence succeeded in developing a HITS-based algorithm which can find top ten authorities of sufficiently good quality on today's Web. Our algorithms take almost the same amount of memory and running time as those taken by the original HITS algorithm, and hence our algorithms can be executed on demand with a PC having a small amount of main memory. The purpose of this work is to improve the HITS algorithms using only the same link information as that used by the original HITS algorithm. One of the future works would be to combine our algorithms with several techniques utilizing text information like the title of a page.

References

[1] Alta Vista, Digital Equipment Corporation, http://altavista.digital.com/
[2] Y. Asano, Y. Tezuka, and T. Nishizeki, "HITS on today's Web," IEICE Technical Report, COMP2005-62, 2006.
[3] K. Bharat and M.R. Henzinger, "Improved algorithms for topic distillation in a hyperlinked environment," Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.104–111, 1998.
[4] A. Borodin, G.O. Roberts, J.S. Rosenthal, and P. Tsaparas, "Finding authorities and hubs from link structures on the World Wide Web," Proc. 10th International World Wide Web Conference, pp.415–429, 2001.
[5] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," Proc. 7th International World Wide Web Conference, pp.14–18, 1998.
[6] S. Chakrabarti, B. Dom, D. Gibson, J. Keinberg, P. Raghavan, and S. Rajagopalan, "Automatic resource compilation by analyzing hyperlink structure and associated text," Proc. 7th International World Wide Web Conference, pp.65–74, 1998.
[7] A.C. Carvalho, P. Chirita, E. Moura, P. Calado, and W. Nejdl, "Site level noise removal for search engines," Proc. 15th International World Wide Web Conference, pp.73–82, 2006.
[8] D. Fetterly, M. Manasse, and M. Najork, "Spam, damn spam, and statistics: Using statistical analysis to locate spam Web pages," Proc. 7th International Workshop on the Web and Databases, pp.1–6, 2004.
[9] D. Fetterly, M. Manasse, M. Najork, and A. Ntoulas, "Detecting spam Web pages through content analysis," Proc. 15th International World Wide Web Conference, pp.83–92, 2006.
[10] Google, Google Inc., http://www.google.com/
[11] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen, "Combating Web spam with TrustRank," Proc. 30th International Conference on Very Large Databases, pp.576–587, 2004.
[12] G. Jeh and J. Widom, "SimRank: A measure of structual-context similarity," Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.538–543, 2002.
[13] J. Kleinberg, "Authoritative sources in a hyperlinked environment," Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.668–677, 1998.
[14] R. Lempel and S. Moran, "The stochastic approach for link-structure analysis (SALSA) and the tkc effect," Proc. 9th International World Wide Web Conference, pp.387–401, 2000.
[15] L. Li, Y. Shang, and W. Zhang, "Improvement of HITS-based algorithms on Web documents," Proc. 11th International World Wide Web Conference, pp.527–535, 2002.
[16] S. Nomura, S. Oyama, T. Hayamizu, and T. Ishida, "Analysis and improvement of HITS algorithm for detecting Web communities," Proc. 2002 Symposium on Applications and the Internet, pp.132–140, 2002.
[17] M. Wang, "A significant improvement to Clever algorithm in hyperlinked environment," Poster Proc. 11th International World Wide Web Conference, 2002.
[18] X. Wang, Z. Lu, and A. Zhou, "Topic exploration and distillation for web search by a similarity-based analysis," Proc. 3rd International Conference on Web-Age Information Management, pp.316–327, 2002.
[19] B. Wu and B.D. Davison, "Identifying link farm spam pages," Proc. 14th International World Wide Web Conference, pp.820–829, 2005.
[20] Yahoo!, Yahoo! Corporation, http://www.yahoo.com/
[21] Y. Zhang, J.X. Yu, and J. Hou, Web Communities: Analysis and Construction, Springer, Berlin, 2006.

Yasuhito Asano received B.S., M.S. and D.S. in Information Science, the University of Tokyo in 1998, 2000, and 2003, respectively. In 2003–2005, he was a research associate of Graduate School of Information Sciences, Tohoku University. He joined Department of Information Sciences, Tokyo Denki University, in 2006, and he is currently an assistant professor at Tokyo Denki University. His research interests include web mining, network algorithms. He is a member of IPSJ, DBSJ, OR Soc. Japan.



Yu Tezuka received B.E., M.E. degrees from Tohoku University, Japan, in 2004 and 2006, respectively, all in information sciences. He joined Victor Company of Japan, Limited, in 2006, and he is currently a system engineer of Car Navigation Products Development Group, Engineering Department, Mobile Entertainment Category.

Takao Nishizeki received the B.E., M.E., and Dr.Eng. degrees from Tohoku University, Japan, in 1969, 1971, and 1974, respectively, all in electrical communication engineering. He joined Tohoku University in 1974, and is currently Professor of Graduate School of Information Sciences. His areas of research interest are combinatorial algorithms, graph theory, and cryptology. Dr. Nishizeki is a member of the Japan Society for Industrial and Applied Mathematics, and is a Fellow of ACM, IEEE and the Information Processing Society of Japan.