On computing HITS ExpertRank via lumping the hub matrix

Yongxin Dong\textsuperscript{a}, Yuehua Feng\textsuperscript{b} and Jianxin You\textsuperscript{a}

\textsuperscript{a}School of Economics and Management, Tongji University, Shanghai, People’s Republic of China; \textsuperscript{b}School of Mathematics, Physics and Statistics, Shanghai University of Engineering Science, Shanghai, People’s Republic of China

\textbf{ABSTRACT}

The dangling nodes are the nodes with no out-links in the web graph. It saves many computational costs if the dangling nodes are lumped into one node. In this paper, motivated by so many dangling nodes in web graphs, we mainly develop theoretical results for the modified hyperlink-induced topic search (MHITS) model by the lumping method, and the results are also suitable for the HITS model. There are three main findings in the theoretical analysis. First, MHITS can be lumped, although the matrix involved is not stochastic. Second, the hub vector of the nondangling nodes can be computed separately from dangling nodes, and the hub vector of the dangling nodes is parallel to a vector of all ones in MHITS. Third, the authority vector of the nondangling nodes is difficult to compute separately from dangling nodes. The numerical results not only show the feasibility and effectiveness of theoretical analyses, but also demonstrate that the lumped MHITS method can produce a better initial authority vector after the iteration of computing the hub vector.

\textbf{ARTICLE HISTORY}

Received 24 December 2021
Accepted 5 September 2022

\textbf{CONTACT}

Yuehua Feng \textsuperscript{b} fyh1001@163.com
School of Mathematics, Physics and Statistics, Shanghai University of Engineering Science, Shanghai 201620, People’s Republic of China

\textbf{ARTICLE HISTORY}

Received 24 December 2021
Accepted 5 September 2022

\textbf{Keywords}

HITS; lumping; nondangling nodes; dangling nodes; similarity transformation

\textbf{2020 Mathematics Subject Classifications}

65F10; 65F50; 15A18; 15A21; 68P20

\section{1. Introduction}

The PageRank of Page and Brin, Hyperlink-Induced Topic Search (HITS) of Kleinberg, Stochastic Approach for Link-Structure Analysis (SALSA) of Lempel and Moran use dominant eigenvectors of nonnegative matrices for ranking web page purpose [1–3]. PageRank is the basis of Google, HITS is used in the ask.com search engine, and SALSA is a combination of PageRank and HITS [3,4]. Since the late twentieth century, HITS is another extremely successful modern web information retrieval application of dominant eigenvectors. The resulting two ranking vectors (authority vector and hub vector) from HITS provide the ExpertRanks. High authority score pages indicate that they have relevant content, while high hub score pages are expected to contain hyperlinks to relevant content. The HITS method has broad applications such as product quality ranking, similarity ranking, and so on. For discussions on HITS and modified HITS (MHITS) models, together with the literature on modifications to overcome its weaknesses, we refer readers to [5].

The eigenproblems related to web information retrieval and data mining can be of huge dimensions. Because of the memory constraint of computers, the power method has
become the dominant method for solving the HITS and PageRank eigenproblems [4,5]. As the web is very enormous, the web may contain more than $10^9$ pages and increase quickly and dynamically. It can even take much time (several hours or days) to compute a large web ranking vector. For the search-dependent HITS, the computation problem, which only involves users’ query-related nodes is relatively small, that is why there is relatively less acceleration work on HITS. While for the search independent HITS (also called MHITS), the matrices involved are usually of tremendous dimension and effective numerical accelerations are very desirable [6].

As we know that the power method will lack efficiency when the eigengap (in magnitude) between the largest eigenvalue ($\lambda_1$) and the second-largest eigenvalue ($\lambda_2$), is close enough to 1. The famous Krylov subspace methods can converge faster than the power method, but they are not suitable for such web information problems due to relatively large storage and subspace dimension and so on [4,5,7]. Therefore, many acceleration methods for information retrieval model calculations include the aggregation methods [5], extrapolation methods [8,9], and two-stage acceleration methods [10,11] and other contributions [4,7] are developed. Among them, most methods consider the difficult computation case that the gap-ratio $|\frac{\lambda_2}{\lambda_1}|$ approximates 1. By lumping the Google matrix, Ipsen and Selee analysed the relationship between rankings of nondangling nodes and rankings of dangling nodes [12]. To improve the computational efficiency of HITS ExpertRank, the filtered power method combining Chebyshev polynomials is proposed [6]. More theoretical and numerical results on the web information retrieval model are available, see Refs [4,5,7,9,13]. One may raise the question of whether the HITS (or MHITS) model can still have similar lumping results, thus the computation cost can be reduced even if the matrix involved is not stochastic?

The rest of this paper is organized as follows. Section 2 describes the HITS model and modified HITS model briefly. Section 3 derives the main theorems and algorithms for lumping the modified HITS model. Numerical results are reported in Section 4. Finally, we give a short conclusion in Section 5.

2. HITS model and modified HITS model

Kleinberg [2] used matrices

$$L^T L \in \mathbb{R}^{n \times n} \quad \text{and} \quad LL^T \in \mathbb{R}^{n \times n},$$

(1)

to describe web link structure graph in HITS model, where $L \in \mathbb{R}^{n \times n}$ is an adjacent matrix given by

$$l_{ij} = \begin{cases} 1, & \text{if page } i \text{ links to page } j, \\ 0, & \text{otherwise}. \end{cases}$$

The eigenvectors of $L^T L$ or $LL^T$ are employed to reveal the relative importance (rank) of corresponding web pages’ authority vectors or hub vectors. In (1), if web page $i$ has no outlinks (i.e. image files, pdf with no links to other pages), it is called a dangling node; otherwise, it is called a nondangling node.
The HITS method updates \( v_a \) and \( v_h \) iteratively from some initial vectors \( v_a^{(0)} \) and \( v_h^{(0)} \),

\[
v_a^{(k)} = L^T v_h^{(k-1)}, \quad v_h^{(k)} = L v_a^{(k)}, \quad k = 1, 2, \ldots
\]  

(2)

Once one of \( v_a \) and \( v_h \) is convergent, the other vector is solved by multiplying \( L \) or \( L^T \).

From (2), we have the following expression. The authority vector \( v_a \) of the authority matrix \( L^T L \) and the hub vector \( v_h \) of the hub matrix \( LL^T \) are defined by computing the principle eigenvector of \( L^T L \) and \( LL^T \)

\[
\lambda_{\text{max}} v_a = L^T L v_a, \quad \lambda_{\text{max}} v_h = LL^T v_h, \quad \text{where} \quad v_a \geq 0, \quad v_h \geq 0, \quad \|v_a\|_1 = 1, \quad \|v_h\| = 1,
\]  

(3)

respectively, where \( \lambda_{\text{max}} \) is the principle eigenvalue of \( LL^T \) (or \( L^T L \)).

The matrix \( L^T L \) or \( LL^T \) is a symmetric positive semi-definite matrix, thus it has nonnegative eigenvalues. However, the expression (3) can’t guarantee the uniqueness of \( \lambda_{\text{max}}, v_a \) or \( v_h \) due to the reducibility of \( L^T L \) (or \( LL^T \)). It would cause the HITS algorithm to converge to nonunique solutions. The Perron-Frobenius theorem can guarantee the uniqueness of the dominant eigenvalue and eigenvector of irreducible nonnegative data matrices. We want to get a unique dominant eigenvector of the link matrix, but the reducibility of \( L^T L \) causes the HITS algorithm to converge to nonunique solutions. In fact, PageRank encounters the same uniqueness problem, and Page and Brin forced the original matrix’s irreducibility (accurately primitivity as well). The Google matrix is defined by

\[
G = \alpha S + (1 - \alpha) E, \quad S = \tilde{H} + d w^T, \quad E = e v^T, \quad G \in \mathbb{R}^{n \times n},
\]  

(4)

where \( w \geq 0, w \in \mathbb{R}^n, \|w\|_1 \equiv w^T e = 1 \) is the dangling node vector, \( \alpha \in (0, 1) \) is the damping factor, \( v \geq 0, v \in \mathbb{R}^n, \|v\|_1 \equiv v^T e = 1 \) is a personalization vector, \( e \) is the vector of all ones with suitable size, the entries of the dangling node indicator vector \( d \) are defined by

\[
d_i = \begin{cases} 1, & \text{if page } i \text{ has no outlinks}, \\ 0, & \text{otherwise}, \end{cases}
\]  

(5)

and the entries of the web link structure matrix \( \tilde{H} \) are given by

\[
\tilde{H}_{ij} = \begin{cases} \frac{1}{n_i}, & \text{a nonzero integer } n_i \text{ stands for the number of outlinks of page } i \text{ to page } j, \\ 0, & \text{otherwise}. \end{cases}
\]  

(6)

The personalization vector \( v \) is usually chosen as \( \frac{1}{n} e \), where \( e \) is the vector of all ones with suitable size. To be specific, replacing zero rows by \( d w^T \) in the web link structure matrix \( \tilde{H} \) is to modify the matrix stochastically and make the maximum eigenvalue 1. But after this adjustment, the linear system still can not meet the condition for a unique solution (due to the reducibility of the matrix), therefore, \( ev^T \) part in (4) is further used to handle the reducibility problem. Furthermore, a technical modification similar to Google’s PageRank primitivity trick can be applied to HITS. That is, the modified authority matrix \( A \) and
modified hub matrix $H$ are defined by

\[
A = \xi L^T L + \frac{(1 - \xi)}{n} ee^T, \\
H = \xi LL^T + \frac{(1 - \xi)}{n} ee^T, \quad \text{where } 0 < \xi < 1, \ e = [1 \ \cdots \ 1]^T,
\]

respectively [5]. Hence, to obtain a unique ExpertRank of a HITS model, the $ee^T$ part is necessary to force the irreducibility of a HITS model matrix. Accordingly, the authority vector $\pi_a$ and hub vector $\pi_h$ are defined by

\[
\pi_a^T A = \tilde{\lambda}_{\text{max}} \pi_a^T, \quad \pi_h^T H = \hat{\lambda}_{\text{max}} \pi_h^T, \quad \text{where } \pi_a \geq 0, \pi_h \geq 0, \ \pi_a^T e = 1, \ \pi_h^T e = 1, \quad (7)
\]

respective, where $\tilde{\lambda}_{\text{max}}$ is the principle eigenvalue of $A$, $\hat{\lambda}_{\text{max}}$ is the principle eigenvalue of $H$ and $e$ is a suitable length vector of all ones. In this paper, we thus mainly discuss the computation problem (7), which is also called the modified HITS (MHITS) model. ExpertRanks are provided by the authority and hub vectors from HITS (or MHITS). In the following section, we try to make theoretical and practical contributions for computing purposes.

3. Theorems and algorithms on lumping

3.1. Permutation matrix

The adjacency matrix is lumpable if all dangling nodes are lumped into a single node [10,12]. According to (1), the adjacency matrix admits the structure

\[
PLP^T = \begin{bmatrix} L_{11} & L_{12} \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad PL^T P^T = \begin{bmatrix} L_{11}^T & 0 \\ L_{12}^T & 0 \end{bmatrix},
\]

where $P$ is a suitable permutation matrix, $L_{11} \in \mathbb{R}^{k \times k}, L_{12} \in \mathbb{R}^{k \times (n-k)}$ and $k$ is the number of nondangling nodes. For example, the adjacency matrix $L$ for the small graph in Figure 1 is

\[
L = \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0
\end{bmatrix}.
\]

and the permutation matrix $P$ can be defined by

\[
P = I([2, 3, 4, 1], :) = \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0
\end{bmatrix}.
\]

For a web adjacency matrix $L$, we have $l = Le$, where $e = [1 \ 1 \ \cdots \ 1]^T$. Then in exact arithmetic, $l$ is a nonnegative vector with entries positive integers or 0. Hence, we denote by a dangling node set $D = \{i|l_i = 0\}$, a nondangling node set $ND = \{i|l_i \neq 0\}$ and a suitable
size identity matrix $I$. The permutation matrix $P$ is defined by $P = I(\{ND, D \}, :)$ (in Matlab notation). In Figure 1, since $Le = [0 \ 1 \ 1 \ 2]^T$, $D = \{1\}$, $ND = \{2, 3, 4\}$ and $I$ is an $4 \times 4$ identity matrix, thus the permutation can be defined by $P = I([2, 3, 4, 1], :)$. Therefore, this permutation matrix $P$ is not difficult to construct for some relatively realistic $L$.

### 3.2. Main lumping results

The lumping part has been described in detail above. It is mainly related to the choice or computation of $P$. In this subsection, we focus on the main lumping results. As

$$PLL^T P^T = PLP^T PL^T P^T = \begin{bmatrix} L_{11} & L_{12} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} L_{11}^T & 0 \\ L_{12}^T & 0 \end{bmatrix} = \begin{bmatrix} L_{11} L_{11}^T + L_{12} L_{12}^T \\ 0 \\ 0 \end{bmatrix},$$

then for the primitive modification, we obtain

$$PHP^T = P \left( \xi LL^T + \frac{1 - \xi}{n} ee^T \right) P^T = \xi \begin{bmatrix} L_{11} L_{11}^T + L_{12} L_{12}^T & 0 \\ 0 & 0 \end{bmatrix} + \frac{1 - \xi}{n} ee^T = \begin{bmatrix} \xi (L_{11} L_{11}^T + L_{12} L_{12}^T) + \frac{1 - \xi}{n} ee^T \\ \frac{1 - \xi}{n} ee^T \\ \frac{1 - \xi}{n} ee^T \end{bmatrix},$$

where $e$ is a suitable length vector of all ones. By using the lumping method and the similarity transformation matrix

$$Y = I_{n-k} - \hat{e} e_1^T,$$

with $\hat{e} = e - e_1 = [0, 1, \ldots, 1]^T \in \mathbb{R}^{n-k}$, where $I_{n-k} = [e_1 \cdots e_{n-k}]$ denotes the identity matrix of order $n-k$, and $e_i, i = 1, 2, \ldots, n-k$ is its $i$th column vector [13]. Define

$$X = \begin{bmatrix} I_k & 0 \\ 0 & Y \end{bmatrix},$$
then we have

\[ XPHPTX^{-1} = \frac{\xi(L_{11}L_{11}^T + L_{12}L_{12}^T) + 1 - \xi\frac{e}{n}ee^T}{1 - \xi\frac{ee^TY}{n}} \]

\[ \frac{1 - \xi\frac{Yee^T}{n}}{1 - \xi\frac{Yee^TY}{n}} \]

\[ \frac{1 - \xi}{n} \]

\[ \frac{1 - \xi e}{n} Yee^T \]

\[ \frac{1 - \xi}{n} e^T Yee^T \]

\[ \frac{1 - \xi}{n} e^T Y_{n-k}^T \]

\[ \frac{1 - \xi}{n} e^T Y_{n-k} \]

\[ \frac{1 - \xi}{n} e_{1} Y_{n-k}^T \]

\[ \frac{1 - \xi}{n} e_{1} Y_{n-k} \]

\[ \frac{1 - \xi}{n} e_{1} \]

\[ \frac{1 - \xi}{n} e_{1} \]

\[ \frac{1 - \xi}{n} e_{1} \]

where we have used the fact that \( Y^{-1}e_1 = e \). Thus, we have proved the following theorem.

**Theorem 3.1:** With the above notation, let

\[ X = \begin{bmatrix} I_k & 0 \\ 0 & Y \end{bmatrix}, \text{ where } Y = I_{n-k} - \hat{e}e_1^T \text{ and } \hat{e} = e - e_1 = \begin{bmatrix} 0 & 1 & \cdots & 1 \end{bmatrix}^T, \]

and \( P \) be a suitable permutation matrix. Then

\[ XPHPTX^{-1} = \begin{bmatrix} H^{(1)} & H^{(2)} \\ 0 & 0 \end{bmatrix}, \]
where
\[
H^{(1)} = \begin{bmatrix}
H^{(1)}_{11} & \frac{1 - \xi}{n} (n - k) e \\
\frac{1 - \xi}{n} e^T & \frac{1}{n} (n - k)
\end{bmatrix}
\]
and
\[
H^{(2)} = \begin{bmatrix}
\frac{1 - \xi}{n} e e^T Y^{-1} [e_2, \cdots e_{n-k}] \\
\frac{1}{n} e^T Y^{-1} [e_2, \cdots e_{n-k}]
\end{bmatrix},
\]
with \(H^{(1)} \in \mathbb{R}^{(k+1) \times (k+1)}\), \(H^{(2)} \in \mathbb{R}^{(k+1) \times (n-k-1)}\) and \(H^{(1)}_{11} = \xi (L_{11}L_{11}^T + L_{12}L_{12}^T) + \frac{1 - \xi}{n} e e^T\). \(H^{(1)}\) has the same nonzero eigenvalues as \(H\).

The following theorem distinguished the relationship between the hub ranking vector \(\pi_h\) of \(H\) and the stationary distribution \(\sigma\) of \(H^{(1)}\). As we can see, the leading \(k\) elements of \(\pi_h\) represent the hub vector due to the nondangling nodes, and the trailing \(n-k\) elements stand for the hub vector associated with the dangling nodes. Thus the relationship between the ranking of dangling nodes and that of nondangling nodes is derived. For ease of the following proof process, we show the structure of the submatrix \(Y\). Separate the first leading row and column,
\[
Y = \begin{bmatrix}
1 & 0 \\
e & I_{n-k-1}
\end{bmatrix}.
\]  
Motivated by validating the analytic relationship between the ranking vector of nondangling nodes and that of dangling nodes in the hub matrix model, we present our main lumping results for MHITS.

**Theorem 3.2:** With the above notation, let
\[
\sigma^T H^{(1)} = \lambda_{\text{max}} \sigma^T, \quad \sigma \in \mathbb{R}^{k+1}, \quad \sigma \geq 0, \quad \|\sigma\|_1 = 1,
\]  
where \(H^{(1)}\) is defined by (14) and partition \(\sigma^T = [\sigma_{1,k}^T \sigma_{k+1}^T]\), where \(\sigma_{k+1}\) is a scalar, and \(\lambda_{\text{max}}\) is the largest eigenvalue of \(H\). Then the hub vector of \(H\) equals
\[
\pi_h^T = \begin{bmatrix}
\sigma_{1,k}^T & \frac{1 - \xi}{n\lambda_{\text{max}}} e^T
\end{bmatrix} P,
\]  
where \(P\) is a suitable permutation matrix satisfying (8).

**Proof:** According to Theorem 3.1, the stochastic matrix \(H^{(1)}\) of order \(k + 1\) has the same nonzero eigenvalues as \(H\). From (14) and (16), we can obtain that \([\sigma^T \frac{1}{\lambda_{\text{max}}} \sigma^T H^{(2)}]\) is an eigenvector for \(XPH^T X^{-1}\) associated with the eigenvalue \(\lambda_{\text{max}}\). Therefore,
\[
\tilde{\pi}^T = \begin{bmatrix}
\sigma^T & \frac{1}{\lambda_{\text{max}}} \sigma^T H^{(2)}
\end{bmatrix} XP
\]  
is an eigenvector of \(H\) associated with \(\lambda_{\text{max}}\). Since \(H\) and \(H^{(1)}\) have the same nonzero eigenvalues, and the principle eigenvalue of \(H\) is distinct, the stationary probability distribution
σ of $H^{(1)}$ is unique. We repartition

$$\tilde{\pi}^T = \left[ \sigma^T_{1:k} \begin{bmatrix} \sigma_{k+1} & \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} \end{bmatrix} \right] \begin{bmatrix} I_k & 0 \\ 0 & Y \end{bmatrix} P.$$ 

Multiplying out $\tilde{\pi}^T = \left[ \sigma^T_{1:k} \begin{bmatrix} \sigma_{k+1} & \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} \end{bmatrix} Y \right] P$. Hence, by (14) and (15), we have

$$\begin{bmatrix} \sigma_{k+1} & \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} \end{bmatrix} = \begin{bmatrix} \sigma_{k+1} - \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} e \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} \\ \frac{1}{\lambda_{\max}} \begin{bmatrix} 1 - \frac{\xi}{n} (n-k) e \\ \frac{1 - \frac{\xi}{n} (n-k)}{n} \end{bmatrix} - \frac{1}{\lambda_{\max}} \sigma^T H^{(2)} e \\ \frac{1}{\lambda_{\max}} \begin{bmatrix} 1 - \frac{\xi}{n} e^T \begin{bmatrix} e \end{bmatrix} \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} - \frac{1}{\lambda_{\max}} \sigma^T \begin{bmatrix} 1 - \frac{\xi}{n} e^T \begin{bmatrix} e \end{bmatrix} \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} \\ \frac{1}{\lambda_{\max}} \begin{bmatrix} 1 - \frac{\xi}{n} e^T \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} \begin{bmatrix} e_2 \cdots e_{n-k} \end{bmatrix} \\ \frac{1}{\lambda_{\max}} \begin{bmatrix} 1 - \frac{\xi}{n} e^T \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} \begin{bmatrix} 1 - \frac{\xi}{n} e^T \begin{bmatrix} e \end{bmatrix} \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} \begin{bmatrix} e_2 \cdots e_{n-k} \end{bmatrix} \\ \frac{1}{\lambda_{\max}} \begin{bmatrix} 1 - \frac{\xi}{n} e^T \\ \frac{1 - \frac{\xi}{n} e^T}{n} \end{bmatrix} \begin{bmatrix} e_2 \cdots e_{n-k} \end{bmatrix} \end{bmatrix}.$$ 

due to the fact that

$$Y^{-1} \begin{bmatrix} e_2 \cdots e_{n-k} \end{bmatrix} = \begin{bmatrix} e_2 \cdots e_{n-k} \end{bmatrix},$$

$$\sigma_{k+1} = \frac{1}{\lambda_{\max}} \sigma^T \begin{bmatrix} 1 - \frac{\xi}{n} (n-k) e \\ \frac{1 - \frac{\xi}{n} (n-k)}{n} \end{bmatrix},$$

and

$$\sigma^T e = 1.$$
Hence,

\[
\tilde{\pi}^T = \begin{bmatrix}
\sigma_{1:k} & \frac{1 - \xi}{n\lambda_{\text{max}}} e^T
\end{bmatrix} P,
\]

where \( P \) is a suitable permutation matrix satisfying (8). As discussed above and \( \pi_h \) is unique, we conclude that \( \tilde{\pi} = \pi_h \) if \( e^T \tilde{\pi} = 1 \).

\[\blacksquare\]

**Remark 3.1:** Theorem 3.2 shows that we can compute the ranking vector of submatrix \( H^{(1)} \) which is derived from the hub matrix, and then recover the hub vector according to (17).

**Remark 3.2:** One can generalize the concrete invertible matrix \( Y \) in (15) in Theorems 3.1 and 3.2 by any invertible similarity transformation matrix satisfying the condition \( \hat{Y} e = e_1 \). For more detailed induction, we refer readers to [13].

**Remark 3.3:** Since

\[
PAP^T = PL^T LP^T = PL^T P^T PL^T = \begin{bmatrix}
L_{11}^T & 0 \\
L_{12}^T & 0
\end{bmatrix} \begin{bmatrix}
L_{11} & L_{12} \\
0 & 0
\end{bmatrix} = \begin{bmatrix}
L_{11}^T L_{11} & L_{11}^T L_{12} \\
L_{12}^T L_{11} & L_{12}^T L_{12}
\end{bmatrix},
\]

we thus remark that it is cheaper to compute (9), rather than (18). This is due to \( PAP^T \) is denser than \( PHP^T \). Hence, it is better to compute the hub matrix \( LL^T \) first, when compared with the authority matrix \( L^T L \). It is also better to compute the modified hub matrix \( \xi LL^T + \frac{(1 - \xi)}{n} ee^T \) first, when compared with the modified authority matrix \( \xi L^T L + \frac{(1 - \xi)}{n} ee^T \), where \( \xi \in (0, 1) \).

**Remark 3.4:** One can permutate \( L \in \mathbb{R}^{n \times n} \) such that (8) holds. However, the web adjacency matrix is very sparse (usually there is only about ten entries per row), one can even permutate \( L \) such that \( \hat{P}L\hat{P}^T = \begin{bmatrix}
\hat{L}_{11} & 0 \\
\hat{L}_{21} & 0
\end{bmatrix} \), where \( \hat{P} \) is a suitable permutation matrix, \( \hat{L}_{11} \in \mathbb{R}^{\hat{k} \times \hat{k}} \) and \( \hat{L}_{21} \in \mathbb{R}^{(n - \hat{k}) \times \hat{k}} \). This phenomenon can be verified by web data matrices from on-line Florida sparse matrix collection. Particularly, the number of all zero columns may be more than the number of all zero rows. In this HITS or MHITS model case, the authoritative ranking vector is recommended to have priority in computing. But note that this case is very rare.

### 3.3. Algorithm derivation

Based on Theorem 3.2, since \( e^T \pi_h = 1 \), the Equation (17) become

\[
1 = e^T \pi_h = e^T \sigma_{1:k} + \frac{1 - \xi}{n\lambda_{\text{max}}} (n - k),
\]

then

\[
\frac{1 - \xi}{n\lambda_{\text{max}}} = \frac{1 - e^T \sigma_{1:k}}{n - k},
\]
therefore

\[
\pi_h^T = \begin{bmatrix} \sigma_{1:k}^T & 1 - e^T \sigma_{1:k} e^T \end{bmatrix} P. \quad (19)
\]

Moreover, we can know that the computation of the hub vector \( \pi_h \) is only related to \( \sigma \) (dominant eigenvector of \( H(1) \)) and permutation matrix \( P \), and independent of the dominant eigenvalue \( \lambda_{\text{max}} \) of \( H(1) \) in (19), so we just use the power method to compute the stationary distribution \( \sigma \) of the lumped matrix \( H(1) \). The iteration process of power method on matrix \( H(1) \) can be simplified by direct computation, that is

\[
\sigma_{1:k} = \xi \tilde{L} \tilde{L}^T \sigma_{1:k} + \frac{1 - \xi}{n} e, \quad \text{with} \quad \tilde{L} = \begin{pmatrix} L_{11} & L_{12} \end{pmatrix} \in \mathbb{R}^{k \times n},
\]

\[
\sigma_{k+1} = \frac{1 - \xi}{n} (n - k),
\]

therefore, we only need to update \( \sigma_{1:k} \) at each iteration process.

In the lumped MHITS (LMHITS) and lumped HITS (LHITS) methods, which are described in Algorithm 1, we use the above process to compute the stationary distribution \( \sigma \) first, then recover the hub vector \( \pi_h \) by (19). Furthermore, we compute the authority vector \( \pi_a \). Since LMHITS and LHITS methods need to be compared with the MHITS and HITS algorithms in numerical experiments, we list MHITS and HITS methods in Algorithm 2. When \( \xi = 1 \) in Algorithms 1 and 2, MHITS and LMHITS reduce to HITS and LHITS, respectively. \( \tilde{\sigma} \) and \( \tilde{\pi}_h \) are chosen as the initial values for computing authority vector \( \pi_a \) in LMHITS and MHITS, respectively. We will describe the reason for this kind of choice in the numerical experiment. The residual norm vectors \( \text{resvec}_a \) and \( \text{resvec}_h \), output by Algorithms 1 and 2, represent the residual norm at each iteration for computing authority vector and hub vector, respectively.

Since \( H(1) \) has the same nonzero eigenvalues as \( H \) (see Theorem 3.2), Algorithm 1 has the same convergence rate as the convergence rate of the power method applied to the full hub matrix \( H \). But it is faster due to operating on a smaller matrix. All advantages of the famous power method are retained by Algorithms 1–2. These algorithms require minimal storage and are easy to implement.

### 4. Numerical experiments

To show the value of the theoretical result, numerical experiments are reported in this section. Numerical experiments are run on Matlab 2016b, with a 2.9 GHz Intel Core i5 processor and 8 GB RAM. Some numerical tests are executed on sparse web matrices \(^1\) with some basic information shown in Table 1. The number of none zero (NNZ) elements in each matrix and dangling node ratio are also summarized. The parameter \( \xi \) in MHITS and LMHITS is set as 0.95, and the overall termination criterion is triggered once the residual norm is below \( 10^{-10} \). Besides, we denote by ‘Iter’ the number of iterations, and by ‘CPU’ the CPU time used in seconds. The symbols \( \tilde{\pi}_h, \tilde{\sigma}, \pi_a, \pi_h \) should be referred in Algorithms 1 and 2.

Figure 2 displays the run time to find the corresponding permutation matrix for eight different matrices, respectively. As we can see, for the test matrices, the time cost is relatively
Algorithm 1 Lumped MHITS and lumped HITS methods

Input: A web adjacency matrix $L$, a positive parameter $\xi \in (0, 1]$;
Output: An authority vector $\pi_a$, a hub vector $\pi_h$, residual vectors $\text{resvec}_a$ and $\text{resvec}_h$.

% Obtain permutation matrix $P$
\begin{align*}
n &= \text{size}(L, 1);
\text{rowsumvector} &= L \ast \text{ones}(n, 1);
\text{ND} &= \text{find}(\text{rowsumvector}) ;
\text{D} &= \text{find}(\text{rowsumvector} == 0);
k &= \text{length}(\text{ND}) ;
P &= \text{I}([\text{ND}; \text{D}], :);
\tilde{L} &= PLP^T ;
\tilde{L} &= \tilde{L}(1 : k, :);
\end{align*}

Initialize: $\sigma = 1/(k + 1) \ast \text{ones}(k, 1); \text{iter} = 0$;
while not convergent do % The power method applied to $H^{(1)}$.
\begin{align*}
\tilde{\sigma} &= \sigma; \text{iter} = \text{iter} + 1;
\tilde{\sigma} &= \tilde{L}^T \sigma ;
\sigma &= \xi \tilde{\sigma} + (1 - \xi)/n \ast \text{ones}(k, 1) ;
\sigma &= \sigma/(|\sigma|_1 + (1 - \xi)(n - k)/n) ;
\text{resvec}_{\pi_a}(\text{iter}) &= |\sigma - \tilde{\sigma}|_1 ;
\end{align*}
end while
$\pi_h = [\sigma; (1 - |\sigma|_1)/(n - k) \ast \text{ones}(n - k, 1)];$ % Recover hub vector $\pi_h$. if $\xi < 1$ then % Compute authority vector by the LMHITS method.
Initialize: $\pi_a = \tilde{\sigma}; \text{iter} = 0$;
while not convergent do
\begin{align*}
\tilde{\pi}_a &= \pi_a; \text{iter} = \text{iter} + 1;
\pi_a &= \xi \tilde{L}^T \tilde{\pi}_a + (1 - \xi)/n \ast \text{ones}(n, 1) ;
\pi_a &= \pi_a/|\pi_a|_1 ;
\text{resvec}_{\pi_a}(\text{iter}) &= |\pi_a - \tilde{\pi}_a|_1 ;
\end{align*}
end while
else % Compute authority vector by the LHITS method when $\xi = 1$.
$\pi_a = \tilde{L}^T \sigma ;$
$\pi_a = \pi_a/|\pi_a|_1 ;$
end if
$\pi_a = P^T \pi_a ;$
$\pi_h = P^T \pi_h ;$

small, and the largest time cost is around 0.25 seconds in practice. It is simple to implement and requires minimal cost. As a consequence, this permutation $P$ is not expensive to construct.

Table 2 displays the running time for eight matrices by HITS, LHITS, MHITS, and LMHITS methods (since the iteration numbers of HITS and LHITS methods are the same, as are MHITS and LMHITS methods, hence we don’t list them in Table 2). We observe that LMHITS is superior to LHITS in terms of CPU, and LHITS is faster than HITS. Overall, for the numerical results presented, the running time in the proposed algorithm is comparable...
Algorithm 2 Modified HITS and HITS methods

Input: A web adjacency matrix $L$, a positive parameter $\xi \in (0, 1]$;
Output: An authority vector $\pi_a$, a hub vector $\pi_h$, residual vectors $\text{resvec}_a$ and $\text{resvec}_h$.

1. $n = \text{size}(L, 1)$;
2. Initialize: $\pi_h = 1/n \ast \text{ones}(n, 1)$; $\text{iter} = 0$;
3. while not convergent do
   1. $\tilde{\pi}_h = \pi_h$; $\text{iter} = \text{iter} + 1$;
   2. $\hat{\pi}_h = L^T \pi_h$;
   3. $\pi_h = \xi L \tilde{\pi}_h + (1 - \xi)/n \ast \text{ones}(n, 1)$;
   4. $\pi_h = \pi_h/\|\pi_h\|_1$;
   5. $\text{resvec}_h(\text{iter}) = \|\pi_h - \tilde{\pi}_h\|_1$;
4. end while
5. if $\xi < 1$ then % Compute authority vector by the MHITS method.
   1. Initialize: $\pi_a = \hat{\pi}_h$; $\text{iter} = 0$;
   2. while not convergent do
      1. $\tilde{\pi}_a = \pi_a$; $\text{iter} = \text{iter} + 1$;
      2. $\hat{\pi}_a = \xi L^T L \pi_a + (1 - \xi)/n \ast \text{ones}(n, 1)$;
      3. $\pi_a = \pi_a/\|\pi_a\|_1$;
      4. $\text{resvec}_a(\text{iter}) = \|\pi_a - \tilde{\pi}_a\|_1$;
   3. end while
6. else % Compute authority vector by the HITS method when $\xi = 1$.
   1. $\pi_a = L^T \pi_h$; $\pi_a = \pi_a/\|\pi_a\|_1$;
7. end if

Table 1. Web adjacency test matrices used in numerical tests.

| Matrix             | Order | NNZ   | Dangling node ratio |
|--------------------|-------|-------|---------------------|
| California         | 9664  | 16150 | 47.98%              |
| email-EuAll        | 265214| 420045| 15.01%              |
| p2p-Gnutella31     | 62586 | 147892| 73.82%              |
| wb-cs-stanford     | 9914  | 36854 | 28.86%              |
| web-BerkStan       | 685230| 760059| 0.69%               |
| web-Google         | 916428| 5105039| 19.31%             |
| web-NotreDame      | 325729| 1497134| 57.65%            |
| wiki-Talk          | 2394385| 5021410| 93.84%           |

to those obtained by HITS and MHITS. Therefore, we can conclude that these lumped-type algorithms are cheaper to compute expert ranking vectors. The wiki-talk matrix has over 90% dangling nodes, while the web-BerkStan has very few dangling nodes, we do not see more improvements for wiki-talk over web-BerkStan in Table 2. One reason for this result is that the iteration number of wiki-talk is much smaller than that of web-BerkStan. The iteration numbers of MHITS and LMHITS can be partly found in Table 3.

Table 3 lists the efficiency of LMHITS and MHITS for computing ExpertRank vectors with different initial authority vectors in Algorithms 1 and 2. For MHITS and LMHITS, the initial values of the authority vector are chosen as $\tilde{\pi}_h$ and $\tilde{\sigma}$, respectively; for MHITS2 and LMHITS2, the initial values are both chosen as $\pi_h$; for MHITS3 and LMHITS3, the
Figure 2. Time cost (in seconds) to find the permutation matrix for each web adjacency matrix.

Table 2. Comparison of CPU time for test matrices with four methods.

| Matrix          | HITS   | LHITS  | MHITS  | LMHITS |
|-----------------|--------|--------|--------|--------|
| California      | 0.024  | 0.021  | 0.036  | 0.034  |
| email-EuAll     | 3.745  | 2.507  | 4.801  | 3.792  |
| p2p-Gnutella31  | 0.256  | 0.204  | 0.344  | 0.266  |
| wb-cs-stanford  | 0.022  | 0.016  | 0.025  | 0.023  |
| web-BerkStan    | 30.318 | 27.719 | 41.186 | 35.805 |
| web-Googe       | 29.962 | 22.632 | 37.431 | 30.203 |
| web-NotreDame   | 0.785  | 0.642  | 0.840  | 0.667  |
| wiki-Talk       | 3.392  | 2.371  | 5.015  | 3.111  |

initial values are both chosen as $e/n$. In the 'Iter' result, the numbers inside and outside parentheses represent the number of iterations for the hub vector and authority vector, respectively.

From numerical results in Table 3, we observe that only the iteration numbers of authority vectors vary while those of hub vectors are the same for each test matrix, since the initial authority vector is different and the initial hub vector is the same. Table 3 also demonstrates that the efficiency of algorithms depends on how to choose an initial authority vector suitably. By comparing the above three initial value strategies for authority vectors, one can observe that the initial value choice of MHITS is superior to that of MHITS2 and MHITS3; and the initial value choice of LMHITS is also superior to that of LMHITS2 and LMHITS3.

In the process of obtaining the main results in Table 3, we observe that the residual norms of the hub vector $\pi_h$ (or the authority vector $\pi_a$) between MHITS2 and LMHITS2 in each iteration are almost the same. The same phenomenon happens between MHITS3 and LMHITS3 in each iteration. Besides, the residual norms of the hub vector $\pi_h$ between MHITS and LMHITS in each iteration are also almost the same. Exceptions are made by
Table 3. The performance of MHITS, MHITS2, MHITS3 and their lumped types in Algorithms 1–2 with different initial authority vectors.

| Matrix        | CPU  | MHITS | MHITS2 | MHITS3 | LMHITS | LMHITS2 | LMHITS3 |
|---------------|------|-------|--------|--------|--------|---------|---------|
| California    | iter | 0.036 | 0.131  | 0.091  | 0.034  | 0.127   | 0.081   |
| email-EuAll   | iter | 4.801 | 7.404  | 7.224  | 3.792  | 5.956   | 5.466   |
| p2p-Gnutella31 | iter  | 0.344 | 0.513  | 0.459  | 0.266  | 0.377   | 0.417   |
| wb-cs-stanford | iter | 0.025 | 0.045  | 0.051  | 0.023  | 0.038   | 0.043   |
| web-BerkStan  | iter | 41.186| 42.223 | 59.731 | 35.805 | 41.287  | 57.386  |
| web-NotreDame| iter | 0.840 | 1.440  | 1.529  | 0.667  | 1.229   | 1.274   |
| wiki-Talk     | iter | 5.015 | 7.980  | 7.532  | 3.111  | 5.637   | 5.608   |

the residual norms of the authority vector \( \pi_a \) in MHITS and LMHITS in each iteration, which are shown in Figure 3. We can see that LMHITS is superior to MHITS by offering better initial values of the authority vector for the next iterations. The residual norm of the authority vector in LMHITS is smaller, especially at the beginning of iterations. In conclusion, \( \hat{\sigma} \) produced in LMHITS is closer to the authority vector \( \pi_a \). Consequently, choosing \( \hat{\sigma} \) as the initial value for computing the authority vector in Algorithm 1 is applicable and wise.

Figure 3. Residual norms of the authority vector for each web adjacency matrix.
Table 4. Rank comparison obtained from the HITS and MHITS algorithms.

| Matrix          | \(dr_a\)  | \(dr_h\)  |
|-----------------|-----------|-----------|
| California      | 7735(80.04%) | 5067(52.43%) |
| email-EuAll     | 210529(79.38%) | 97164(36.64%) |
| p2p-Gnutella31  | 51207(81.82%) | 7375(11.78%) |
| wb-cs-standford | 8560(86.34%) | 6693(67.51%) |
| web-BerkStan    | 649505(94.79%) | 606670(88.54%) |
| web-Google      | 574782(62.72%) | 745217(81.32%) |
| web-NotreDame   | 311386(95.60%) | 311305(95.57%) |
| wiki-Talk       | 54309(2.27%) | 642(0.03%) |

Table 5. The corresponding index when the element ranking of sorted authority (or hub) vectors start to make a difference.

| Matrix          | \(ci_a\) | \(ci_h\) |
|-----------------|---------|---------|
| California      | 1223    | 807     |
| email-EuAll     | 5342    | 8399    |
| p2p-Gnutella31  | 369     | 268     |
| wb-cs-standford | 19      | 35      |
| web-BerkStan    | 2       | 5806    |
| web-Google      | 136     | 666     |
| web-NotreDame   | 11354   | 10834   |
| wiki-Talk       | 3795    | 136373  |

Table 4 shows rank comparison obtained from the HITS and MHITS algorithms. We sort the authority vectors and the hub vectors output by HITS and MHITS (Algorithm 2) from most important to least important. \(dr_a\) (or \(dr_h\)) represents the numbers of elements in these two sorted authority (or hub) vectors that differ in their ranking. In Table 4, the numbers inside parentheses represent the ratio of \(dr_a\) (or \(dr_h\)) to the order of the corresponding adjacent matrix. We can see that the results obtained from the HITS and the MHITS algorithms sometimes become relatively small, but sometimes become relatively different. Furthermore, we also use \(ci_a\) (or \(ci_h\)) to denote the corresponding index when the element ranking of sorted authority (or hub) vectors start to make a difference in Table 5. It is shown that the top ranks of ExpertRank are almost the same for most test matrices. Therefore, it makes sense to use the MHITS algorithm to obtain ExpertRank.

5. Conclusion

In this paper, we have studied a HITS (or MHITS) computation approach via lumping dangling nodes of a hub matrix \(LL^T\) (or a modified hub matrix \(\xi LL^T + \frac{(1-\xi)}{n} ee^T, 0 < \xi < 1\)) into a single node. Thus we have answered the model’s computation question left in the introduction.

The HITS or MHITS can be computed by lumping approach despite the involved matrices are not stochastic. The approach which we discuss is useful whether the HITS model is search-dependent or not. For the hub vector, Theorem 3.2 shows us that the rankings of nondangling nodes can be computed independently from that of dangling nodes; while rankings of dangling nodes are constant vectors \((ke, \text{where } k\text{ is a constant and } e\text{ is a vector...})\)
of all ones). According to Remark 3.3, the authority vector is relatively difficult to compute when compared with the hub vector. Thus we suggest that it is better to compute the hub vector in priority rather than the authority vector or both of them simultaneously. The experimental results also verify our theory, and the results show that LMHITS not only accelerates the iterative process of MHITS, but also generates a better initial value of the authority vector after the computation of the hub vector.

Further research may include how to compute SALSA like HITS. Questions like the computation effects of dangling nodes in the SALSA model are also worthy of study.

**Note**

1. Available at the SuiteSparse Matrix Collection (https://sparse.tamu.edu/).

**Acknowledgments**

The authors would like to thank the editor and referees for their constructive and insightful comments and suggestions which are very helpful in improving the quality of the earlier version of this manuscript.

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**Funding**

This work was supported by National Natural Science Foundation of China [grant numbers 12001363, 71671125].

**References**

[1] Page L, Brin S, Motwani R, et al. The PageRank citation ranking: bringing order to the web. Stanford Digital Libraries. 1999. Available from: http://dbpubs.stanford.edu:8090/pub/1999-66).

[2] Kleinberg JM. Authoritative sources in a hyperlinked environment. J ACM. 1999;46:604–632.

[3] Lempel R, Moran S. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. Comput Netw. 2000;33:387–401.

[4] Langville AN, Meyer CD. A survey of eigenvector methods for web information retrieval. SIAM Rev. 2005;47:135–161.

[5] Langville AN, Meyer CD. Google’s PageRank and beyond: the science of search engine rankings. Princeton: Princeton University Press; 2006.

[6] Zhou Y-K. Practical acceleration for computing the HITS ExpertRank vectors. J Comput Appl Math. 2012;236:4398–4409.

[7] Eldén L. Matrix methods in data mining and pattern recognition. Philadelphia: SIAM; 2007.

[8] Brezinski C, Redivo-Zaglia M. The PageRank vector: properties, computation, approximation, and acceleration. SIAM J Matrix Anal Appl. 2006;28:551–575.

[9] Feng Y-H, You J-X, Dong Y-X. An extrapolation iteration and its lumped type iteration for computing PageRank. Bull Iran Math Soc. 2022;48:2603–2620.

[10] Lee CP, Golub GH, Zenios SA. A two-stage algorithm for computing PageRank and multistage generalizations. Internet Math. 2007;4:299–327.

[11] Dong Y-X, Gu C-Q, Chen Z-B. An arnoldi-inout method accelerated with a two-stage matrix splitting iteration for computing PageRank. Calcolo. 2017;54:1–23.
[12] Ipsen ICF, Selee TM. PageRank computation with special attention to dangling nodes. SIAM J Matrix Anal Appl. 2007;29:1281–1296.

[13] Dong Y-X, Feng Y-H, You J-X, et al. Comments on lumping the Google matrix. 2021. Available from: arXiv:2107.11080.