Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 108C (2017) 68–78

Procedia Computer Science

International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

Graph Ranking Guarantees for Numerical Approximations to Katz Centrality

Eisha Nathan[1], Geoffrey Sanders[2], James Fairbanks[3], Van Emden Henson[2], and David A. Bader[1]

[1] School of Computational Science and Engineering, Georgia Institute of Technology
{enathan3,bader}@gatech.edu
[2] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
{sanders29,henson5}@llnl.gov
[3] Georgia Tech Research Institute
james.fairbanks@gtri.gatech.edu

Abstract

Graphs and networks are prevalent in modeling relational datasets from many fields of research. By using iterative solvers to approximate graph measures (specifically Katz Centrality), we can obtain a ranking vector consisting of a number for each vertex in the graph identifying its relative importance. We use the residual to accurately estimate how much of the ranking from an approximate solution matches the ranking given by the exact solution. Using probabilistic matrix norms and applying numerical analysis to the computation of Katz Centrality, we obtain bounds on the accuracy of the approximation compared to the exact solution with respect to the highly ranked nodes. This relates the numerical accuracy of the linear solver to the data analysis accuracy of finding the correct ranking. In particular, we answer the question of which pairwise rankings are reliable given an approximate solution to the linear system. Experiments on many real-world networks up to several million vertices and several hundred million edges validate our theory and show that we are able to accurately estimate large portions of the approximation. By analyzing convergence error, we develop confidence in the ranking schemes of data mining.

Keywords: graphs, data analysis, numerical accuracy, katz centrality, ranking

1 Introduction

Graphs are a very popular means of representing massive amounts of relational data. One of the most popular questions arising from the analysis of large graphs is to determine the most important vertices in a graph. Vertex importance is referred to as *centrality*, and centrality scores can be used to provide *rankings* on the vertices of a graph. While there exist many such centrality measures, in this work we focus on Katz Centrality because of its analytical

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science
10.1016/j.procs.2017.05.021

tractability. Efficiently solving for the Katz centrality in a graph involves solving a linear system. Obtaining an exact solution via direct methods is prohibitively computationally expensive, since we are required to take the inverse of a matrix. The most accurate way to obtain the exact solution would be by Cholesky decomposition, which costs $\mathcal{O}(n^2)$, where n is the number of vertices in the graph. In many real networks the amount of data is massive and n can be as large as millions or billions of vertices, so direct methods such as these do not scale and are impractical. Moreover, there is no technique to compute an exact solution for a general graph in finite precision arithmetic, so in practice, iterative methods are often used to obtain an approximate solution. Iterative methods optimally cost $\mathcal{O}(m)$, where m is the number of edges in the graph, but to achieve this optimal complexity the number of iterations must be limited. Many real-world graphs are sparse and $m \ll n^2$ [1]. In this paper we provide theoretical guarantees (Theorem 1) on the accuracy of an approximate solution compared to the exact solution to certify rankings in the approximation, and explain how they can be used to limit the number of iterations in the iterative solver.

This work bridges the gap between the fields of numerical analysis and data mining by understanding the effect that the error in a numerical problem has on the confidence of the data analysis problem of ranking. We solve the data mining problem of ranking by solving the numerical problem of obtaining a solution to a linear system. Using iterative methods to obtain an approximate solution to this linear system inherently gives rise to some error in the approximation. We explain how this error affects how much of the approximate solution is accurate with respect to the unknown exact solution to the ranking problem. Additionally, we develop practical algorithms that leverage our theory. For many application purposes it is primarily the highly-ranked vertices that are of interest. Consider performing a web search with Google. Typically anyone running a web search has enough human resources to examine the top. In a Twitter graph, we might wish to identify the most influential voices in a subset of Twitter users, or in a network modeling disease spread an analyst would be interested in finding sites of disease origin. These queries are answered by examining the highly ranked vertices in the graph.

In this paper, we obtain bounds on the accuracy of the approximation compared to the exact solution using properties of error analysis on linear solvers. We validate our theoretical guarantees of certifying the accuracy of the top ranked vertices across several real-world networks and show that our method is able to find these vertices in a fraction of the time compared to the standard approach of running to machine precision. The main goal of the work is to improve our understanding of how numerical accuracy affects data analysis accuracy.

1.1 Contributions

This paper makes the following contributions:

- A new error bound (Theorem 1) on elements of a ranking vector to provide graph ranking guarantees to the computation of Katz Centrality.
- A new stopping criterion for iterative solvers to identify top ranked vertices in a graph that reduces runtime compared to running a solver to machine precision.
- Empirical evidence of a tighter probabilistic upper bound on $\|A\|_2$ compared to deterministic Gershgorin bounds for real-world graphs.
- Demonstrations that these bounds provide practical results in real datasets.

1.2 Related Work

Many data analysis problems are answered by solving an induced numerical problem. We present how numerical error in the approximation to the solution of a linear system affects the solution to the original ranking problem. Several centrality measures can be expressed as functions of the adjacency matrix of a graph [2]. Pagerank [13] is a common method for ranking vertices in graphs, where a high score means random walks through the graph tend to visit the highly ranked vertices. Similarly the exponential-based centrality measure weights walks of length k by a factor of $\frac{1}{k!}$ [7]. Here, we address the ranking problem for Katz Centrality [9], a centrality metric that measures the affinity between vertices as a weighted sum of the paths between them.

Solving for many linear algebra based centrality measures directly is generally intractable so iterative solvers are used to approximate them [4]. Understanding the error in the approximate solution to the numerical problem is key to understanding the error in the data mining problem. Ranking vertices in graphs and finding the top ranked vertices is of very practical relevance to data analysts [6]. We focus on approximating the Katz score of the vertices in the graph to a high enough accuracy to certify that the top of the ranking vector is accurate compared to the exact solution. Several other methods for approximating Katz scores across the network only examine paths up to a certain length [5] or employ low-rank approximation [12]. In [3], the authors provide theoretical guarantees for pairwise Katz scores and provide an algorithm to find the Katz scores from one vertex to the rest of the graph with reduced complexity. Our work differs in that we provide confidence as to which portion of the global ranking is correct and use the size of the residual to provide an accurate estimation of the ranking.

The main contribution of this paper is bounding the error between the approximate and exact solutions to accurately certify top portions of the ranking with thorough experimentation to validate our results. We derive the bound and provide error analysis in Section 2. Numerical experiments validating the bound including analysis of both precision and performance of our method are presented in Section 3. Finally, in Section 4, we conclude and discuss further uses of this work.

2 Definitions and Theory

Let $G = (V, E)$ be a graph, where V is the set of n vertices and E the set of m edges. Denote the $n \times n$ adjacency matrix A of G with entries $A(i, j) = 1$ if there exists an edge from vertex i to j, 0 otherwise. In this work we deal with undirected, unweighted graphs so $\forall i, j, A(i, j) = A(j, i)$ and all edge weights are 1, although all the theory presented in this paper can easily be generalized to the weighted case. The matrix 2-norm $\|A\|_2$ is given by the largest singular value, σ_{max}.

Katz centrality rankings quantify the ability of a vertex to initiate walks around the network. The number of walks of length k from vertex i to j is $A^k(i, j)$. The Katz score of vertex i counts the number of closed walks ending at vertex i, while penalizing long walks through the network by multiplying by a fixed user-chosen factor α for each edge used, where $\alpha \in [0, 1/\|A\|_2)$. The Katz centrality of vertex i is given by $\mathbf{e}_i^T \sum_{k=1}^{\infty} \alpha^{k-1} A^k \mathbf{1}$, where $\mathbf{e}_i$ is the ith canonical basis vector and $\mathbf{1}$ is the $n \times 1$ vector of all ones. In practice the Neumann formula [16] is employed to turn this series into a linear solver and we compute the Katz Centrality of all vertices in the graph as $\mathbf{c} = \sum_{k=1}^{\infty} \alpha^{k-1} A^k \mathbf{1} = A(I - \alpha A)^{-1}\mathbf{1}$.

The iterative method we use is conjugate gradient (without a preconditioner) [14], although the theory applies to other iterative techniques. Conjugate gradient is a popular technique to approximate the solution $\mathbf{x}$ in a linear system $M\mathbf{x} = \mathbf{b}$, given M and $\mathbf{b}$. Let $M = I - \alpha A$ so that we solve the system $M\mathbf{x} = \mathbf{1}$ for $\mathbf{x}$ and $\mathbf{c}$ is obtained with a simple matrix vector multiplication as $\mathbf{c} = A\mathbf{x}$ in $\mathcal{O}(m)$. At each step k of the iterative solver we obtain new approximations $\mathbf{x}^{(k)}$ and $\mathbf{c}^{(k)}$ to the exact solutions $\mathbf{x}^*$ and $\mathbf{c}^*$ respectively. The error at each step is denoted as the difference between the exact and approximation, $\|\mathbf{x}^* - \mathbf{x}^{(k)}\|_2$ and the residual norm as $r_k = \|\mathbf{1} - M\mathbf{x}^{(k)}\|_2$, where $\|\cdot\|_2$ denotes the 2-norm. In practice as the exact solution is not known, typical stopping criteria for the iterative solver use the residual norm, terminating when it hits machine precision, $r_k \approx 10^{-15}$. The problem is more ill-conditioned and harder as $\alpha \to 1/\|A\|_2$ and typically requires more iterations to terminate and converge to machine precision.

2.1 Error Analysis

We make the observation that if our goal is identification of the highly ranked vertices in a graph, we ought to focus on *ranking accuracy* not *numerical accuracy*. This is because the error in the data analysis problem of ranking is dfferent than the error in numerical problem of solving the linear system: the relative ranking of vertices can be correct even without a fully correct centrality vector. We theoretically guarantee the accuracy of the solution to numerical problem needed to successfully answer the data mining question of ranking.

When $M^{-1}\mathbf{1}$ is approximated, there will be differences between the approximate solution and the exact solution. We prove that these differences along with the ranking values can indicate how far down the ranking we can go before the approximation error makes it unreliable. Define $\mathbf{b}^{(k)} = \pi^{(k)}\mathbf{c}^{(k)}$, where $\pi^{(k)}$ is the permutation such that $\mathbf{b}^{(k)}$ is the vector $\mathbf{c}^{(k)}$ ordered in decreasing order so that $b_i^{(k)} \geq b_{i+1}^{(k)}$.

Theorem 1. *Let A, M, $\mathbf{x}^{(k)}$, $\mathbf{c}^{(k)}$, $\mathbf{x}^*$, $\mathbf{c}^*$, $\mathbf{b}^{(k)}$, $\mathbf{b}^*$, and r_k be as previously defined. Define $\lambda_{min}(M)$ to be the smallest eigenvalue of the matrix M. Let σ_{up} be any upper bound on $\|A\|_2$. Then for any $i < j$, the ranking of vertex i above j is correct if $|b_i^{(k)} - b_j^{(k)}| > 2\epsilon_k$ for $\epsilon_k = \frac{\sigma_{up}}{\lambda_{min}(M)} r_k$.*

Proof. Using foundations of error analysis in linear solvers, we can bound the point-wise error in the ranking, which will then provide a sufficient error gap in the elements of the approximation to the ranking vector.

$$\begin{aligned}
\|\mathbf{b}^* - \mathbf{b}^{(k)}\|_\infty &= \|\mathbf{c}^* - \mathbf{c}^{(k)}\|_\infty \leq \|\mathbf{c}^* - \mathbf{c}^{(k)}\|_2 \\
&= \|A\mathbf{x}^* - A\mathbf{x}^{(k)}\|_2 \leq \|A\|_2 \|\mathbf{x}^* - \mathbf{x}^{(k)}\|_2 \\
&= \|A\|_2 \|M^{-1}\mathbf{1} - \mathbf{x}^{(k)}\|_2 \leq \|A\|_2 \|M^{-1}\|_2 \|\mathbf{1} - M\mathbf{x}^{(k)}\|_2 \\
&\leq \frac{\|A\|_2}{\lambda_{min}(M)} \|\mathbf{1} - M\mathbf{x}^{(k)}\|_2 \leq \frac{\sigma_{up}}{\lambda_{min}(M)} r_k \\
&=: \epsilon_k
\end{aligned}$$

Since $b_i^{(k)} - b_i^* < \epsilon_k$ and $b_j^* - b_j^{(k)} < \epsilon_k$, this means that $b_i^* - b_j^* > b_i^{(k)} - b_j^{(k)} - 2\epsilon_k$. If $b_i^{(k)} - b_j^{(k)} > 2\epsilon_k$, then $b_i^* - b_j^* > 0$ meaning that the ranking of vertex i above j is correct. □

We observe in practice that this bound is tight enough to produce relevant results in many practical applications and lends itself to the development of a new stopping criterion for iterative solvers when identifying the highly ranked vertices in a graph.

2.2 New Stopping Criterion

Current methods for identifying the top vertices in a graph involve running an iterative solver to machine precision to obtain an approximation of $\mathbf{c}^*$. We introduce a new stopping criterion to find these top vertices that typically provides results much faster than existing methods, based off of the theory developed in Theorem 1. Furthermore, our method provides theoretically sound guarantees as to the correctness of the top vertices, unlike the common method of simply running a solver to machine precision and blindly hoping the resulting vector is good enough for the desired data mining task.

Suppose a user desires a set of j vertices containing the top R highly ranked vertices in a graph, with precision ϕ_0. How large does j need to be before we can accurately certify that the top vertices are in the set? We are not concerned with the internal ordering of this set, but rather that the top R vertices are contained somewhere within the set of j vertices. We answer this question using our theory. At each iteration of conjugate gradient, the current solution $\mathbf{c}^{(k)}$ is ordered in decreasing order to produce the vector $\mathbf{b}^{(k)}$ as described earlier. We find the first position $j > R$ in $\mathbf{b}^{(k)}$ where we find the necessary gap of $|b_R^{(k)} - b_j^{(k)}| > 2\epsilon_k$. The precision for these values of R and j is defined as $\phi = \frac{R}{j-1}$. If for this value of j we have the desired precision ϕ_0 then we terminate, else we iterate again using conjugate gradient to obtain a more accurate approximation. This procedure is given in Algorithm 1, for an adjacency matrix A, upper bound σ_{up} on $\|A\|_2$, number of top vertices R, desired precision ϕ_0, and maximum number of iterations k_{max}. Intuitively the precision shows how far past position R we must travel down the vector to find the necessary gap to ensure we are returning the top R vertices in the graph. Conjugate gradient can be organized to return $\mathbf{x}^{(k)}$, $\mathbf{c}^{(k)}$, and the residual norm r_k at each iteration (denoted `CGiteration` in Algorithm 1).

Algorithm 1: Obtain top R vertices in network with precision ϕ_0

```
1 Function Top_R
      Data: $A, \sigma_{up}, R, \phi_0, k_{max}$
      Result: Set of $j$ vertices s.t. top $R$ vertices are contained within this set
2     $k = 0$; $j = \infty$
3     $M = I - \alpha A$
4     while $\frac{R}{j-1} < \phi_0$ and $k < k_{max}$ do
5         $\mathbf{x}^{(k)}, \mathbf{c}^{(k)}, r_k =$ CGiteration($M, \mathbf{x}^{(k-1)}$)
6         $\mathbf{b}^{(k)} = \pi^{(k)}\mathbf{c}^{(k)}$
7         $\epsilon_k = \frac{\sigma_{up}}{\lambda_{min}(M)} r_k$
8         $j = argmin_{i>R} |b_R^{(k)} - b_i^{(k)}| > 2\epsilon_k$
9         $k$ += 1
```

2.3 Bounds on $\|A\|_2$

We obtain a tight bound on ϵ_k which allows us to certify that the ranking of vertex i above j is correct if the gap between two elements in the ranking vector is greater than our error bound, $|b_i^{(k)} - b_j^{(k)}| > 2\epsilon_k$. Conjugate gradient readily provides the residual norm r_k at each iteration, and $\lambda_{min}(M)$ can be computed provided α is chosen in the given range. To certify portions of the ranking vector, we desire ϵ_k to be as small as possible to find places in the vector where the necessary gap $|b_i^{(k)} - b_j^{(k)}|$ exists. Obtaining a tight bound on $\|A\|_2$ is key to bounding ϵ_k; we present and compare two methods of bounding $\|A\|_2$.

The Gershgorin Circle Theorem [15] bounds the eigenvalues of the symmetric matrix A. Let $T_i = \sum_{j \neq i} |a_{ij}|$, the sum of the nondiagonal entries in row i. Then $D(a_{ii}, T_i)$ is the closed interval centered at a_{ii} with radius T_i and every eigenvalue $\lambda \in \sigma(A)$ must lie within at least one interval $D(a_{ii}, T_i)$, where $\sigma(A)$ is the spectrum of A. Since the diagonal entries a_{ii} of A are 0, the discs are all centered around the origin and $\forall i, T_i = d_i =$ the degree of vertex i. We then have $\|A\|_2 = \max \lambda_i < \max T_i = d_{max}$, where d_{max} is the largest degree in the graph. While this provides a basis for an upper bound of the matrix 2-norm of A, many real-world graphs such as social networks have a scale-free distribution and thus contain vertices with a very large degree. Therefore, this is often a non-optimal bound. By using just a few matrix-vector multiplications applied to random vectors, one can compute tighter bounds with high certainty.

We next examine probabilistic matrix norm bounds [8] and consider replacing the true bound σ_{up} with an estimate of a bound with some probability. These bounds are developed using the polynomials p, q implicitly formed as a part of the Lanczos bidiagonalization process with starting vector $\mathbf{v_1}$, which is chosen randomly with unit norm. The defining relations of Lanczos bidiagonalization are stated as $\alpha_m \mathbf{u}^{(m)} = A\mathbf{v}^{(m)} - \beta_{m-1}\mathbf{u}^{(m-1)}$ and $\beta_m \mathbf{v}^{(m+1)} = A^T\mathbf{u}^{(m)} - \alpha_m \mathbf{v}^{(m)}$ for $\beta_0 = 0$, $\mathbf{u}_0 = \mathbf{0}$. The recurrent polynomials are: $\gamma_{j+1} p_j(t) = q_j(t) - \beta_j p_{j-1}(t)$ and $\beta_{j+1} q_{j+1}(t) = t p_j(t) - \gamma_{j+1} q_j(t)$ where $\gamma_j = {\mathbf{u}^{(j)}}^T A \mathbf{v}^{(j)}$ and $\beta_j = {\mathbf{u}^{(j)}}^T A \mathbf{v}^{(j+1)}$ for $p_{-1}(t) = 0$ and $q_0(t) = 1$. The bound, stated in Theorem 2, is due to [8]. The result is an upper bound $\sigma_{up}(\theta)$ for $\|A\|_2$ with probability 1-θ, where θ is the user-chosen probability of bound failure. Define $\delta = \theta \cdot \frac{1}{2} B(\frac{n-1}{2}, \frac{1}{2})$ where B is Euler's Beta function, $B(x, y) = \int_0^1 t^{x-1}(1-t)^{y-1} dt$.

Theorem 2. *[8] Suppose we have carried out k steps of the Lanczos bidiagonalization process with starting vector $\mathbf{v_1}$, and let $\theta \in (0, 1)$. Then the largest zero of the polynomials,*

$$f_1(t) = q_k(t^2) - 1/\delta, f_2(t) = t p_k(t^2) - 1/\delta$$

with δ given above, is an upper bound $\sigma_{up}(\theta)$ for $\|A\|_2$ with probability at least 1-θ.

As a result of thorough experimentation, for all bounds used in this paper, we select values of θ=0.01 and k=10. For k=10, in order to calculate $\sigma_{up}(0.01)$ we are required to calculate the largest root of a tenth order polynomial. Since this does not change regardless of problem size n, this calculation is asymptotically a fixed cost. We use Python's `Sympy` package to calculate the roots of these polynomials.

The deterministic Gershgorin bounds yield large values of $\|A\|_2$, rendering these bounds useless. On average, these bounds return estimates of $\|A\|_2$ that are 30.9× greater than the true 2-norm. In contrast, the probabilistic bounds presented in Theorem 2 return estimates of $\|A\|_2$ that are only on average 1.07× greater than the true 2-norm, meaning that these are able to be used for practical purposes.

3 Results

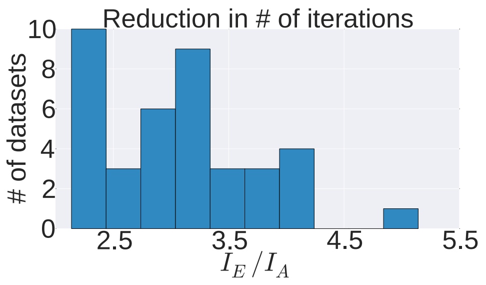
In this section we present comparisons to existing methods for identifying the top ranked vertices with respect to performance and experiments validating our bound with respect to precision. We are interested in determining if our method correctly identifies the set of top vertices and if so, how much faster we are able to certify this set. The common method of iterating to machine precision does not theoretically certify this set but our theory can be used on the machine precision solution as well. We conduct experiments on 39 graphs from the KONECT [10] and SNAP [11] datasets, including social networks, autonomous systems, citation, co-authorship, web, co-purchasing, and road graphs.

Table 1: Graphs used in experiments. Columns are numbers of vertices ($|V|$), number of edges ($|E|$), number of iterations using new stopping criterion (I_A), and number of iterations using old stopping criterion (I_E).

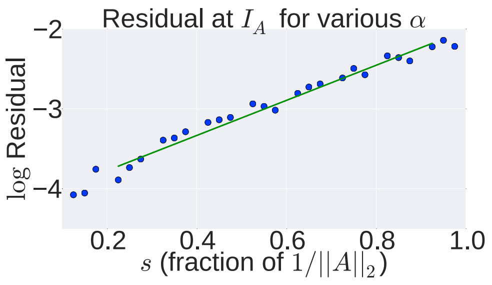
Graph	$\|V\|$	$\|E\|$	I_A	I_E
blogs	1,224	19,025	14	41
USAirports	1,574	28,236	22	62
UCIrvineMessages	1,899	59,835	338	784
figeys-protein	2,239	6,452	8	32
OpenFlights	2,939	30,501	30	81
reactome	6,327	147,547	70	144
as20000102	6,474	13,233	6	23
p2p-Gnutella06	8,717	31,525	22	47
p2p-Gnutella05	8,846	31,839	15	46
oregon1_010331	10,670	22,002	6	24
p2p-Gnutella04	10,876	39,994	12	40
oregon2_010331	10,900	31,180	8	27
ca-AstroPh	18,771	198,050	159	348
cit-cora	21,201	91,500	13	43
p2p-Gnutella25	22,687	54,705	7	36
ca-CondMat	23,133	186,936	57	120
as-caida20071105	26,475	106,762	13	41
cit-HepPh	34,546	421,578	56	141
p2p-Gnutella30	36,682	88,328	13	52
email-Enron	36,692	367,662	42	102
slashdot-threads	50,835	140,778	9	32
p2p-Gnutella31	62,586	147,892	10	37
soc-Slashdot0902	82,168	948,464	10	28
recordlabel	168,268	233,286	7	24
libimseti	220,970	17,359,346	67	189
web-Stanford	281,903	2,312,497	22	44
dblp	317,080	1,049,866	108	257
web-NotreDame	325,729	1,497,134	16	56
com-amazon	334,863	925,872	10	31
cit-citseer	384,413	1,751,463	10	38
soc-twitter	465,017	834,797	37	97
stack-overflow	545,196	1,301,942	11	33
country	590,112	637,134	8	25
web-Google	875,713	5,105,039	18	41
youtube	1,134,890	2,987,624	9	28
as-skitter	1,696,415	11,095,298	27	63
flickr-links	1,715,255	15,550,782	48	106
roadNet-CA	1,965,206	2,766,607	1,332	2,069
livejournal	7,489,073	112,307,385	28	65

3.1 New Stopping Criterion

We first analyze the effect of our stopping criterion on reducing the number of iterations taken by an iterative solver to identify the top R vertices in a network. Denote the number of iterations taken by conjugate gradient to machine precision as I_E and the number of iterations used with our new stopping criterion as I_A. Table 1 shows basic information about each graph used in the experiments as well as raw iteration counts using both the existing and our new stopping criterion. Figure 1a plots a histogram of the reduction in iterations $\frac{I_E}{I_A}$ for $R = 100$ and $\phi_0 = 0.95$. In all cases we obtain a reduction of at least $2\times$, an average of $3.07\times$ reduction, and up to a maximum reduction of $5.14\times$ the number of iterations using our method. This shows that we are able to identify the top $R = 100$ in a fraction of the time using our stopping criterion compared to running until machine precision, while providing a theoretical guarantee that these vertices are in the top of the ranking vector. This is especially significant because



(a) Histogram of the reduction in iterations from computing ranking vector to machine precision versus using new stopping criterion.



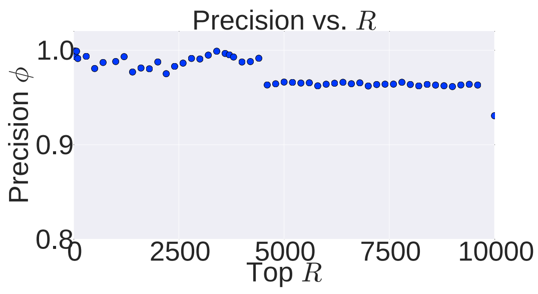
(b) Correlation between α and residual obtained after terminating at stopping criterion. Larger values of s yield larger values of α and indicate harder problems.

Figure 1: Performance results using new stopping criterion for $R = 100$.

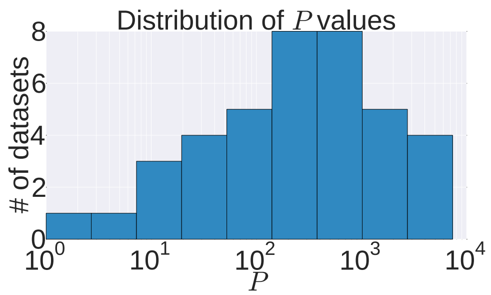
running to machine precision can sometimes take hundreds or thousands of iterations as seen in Table 1. Next we investigate on what problems our method proves to be the most useful. We know as $\alpha \to \frac{1}{\|A\|_2}$, the problem becomes more ill-conditioned. Since $\alpha \in (0, 1/\|A\|_2)$, we apply our stopping criterion to the different graphs for various α in this range. Figure 1b plots the relationship between α and the residual norm obtained when the solver terminates using our criterion. We use $\alpha = s\frac{1}{\sigma_{up}(0.01)}$, substituting the bound $\sigma_{up}(0.01)$ obtained in Section 2 for $\|A\|_2$, for $s \in (0, 1)$. For each value of s, the averaged residual norm is plotted across graphs. When running to machine precision, the residual norm upon termination is typically $r_k \approx 10^{-15}$. The average residual norm across all trials using our stopping criterion is 2.89×10^{-4}. We see that we never have to iterate until machine precision using our new stopping criterion if we are interested in only the top vertices in a graph. Regression analysis of these results (plotted as the green line in Figure 1b) shows a strong linear correlation with a slope of 3.21 and mean sum of squares of 0.83. The linear relationship suggests that we need less accurate approximate solutions for harder problems as $\alpha \to \frac{1}{\|A\|_2}$ to obtain the top vertices in the graph. Typically the harder problems tend to take thousands of iterations to converge with the standard stopping criterion of iterating until a residual norm of 10^{-15}, but with our stopping criterion we can converge faster at a lower tolerance to solve the desired data mining task. The low residual norm suggests we are able to certify the top R correctly with low fidelity solutions and we are able to use this technique to turn harder linear algebra problems into easier data mining problems.

3.2 Accuracy of Approximation

Using the theory presented in Section 2, we show that we are able to accurately identify sets of highly ranked vertices. Recall the *precision* ϕ is given by $\phi = \frac{R}{j-1}$. Intuitively, we calculate the ratio of the number of returned vertices that are relevant and in the desired top to the total number returned. A value close to 1 indicates we have obtained the top R vertices with very few false positives. We evaluate the final precision ϕ for various values of $R \in [10, 10000]$. For example, ϕ for $R = 100$ represents the precision we obtain using our theory to return the smallest set containing the top 100 ranked vertices in the graph. We find the first index $j > 100$ where $|b_R^{(k)} - b_j^{(k)}| > 2\epsilon_k$. Figure 2a plots a scatterplot of ϕ values versus various values of R. For clarity, the y-axis starts at 0.8. We evaluate the precision on the 39 real-world networks and on each network test on values of R ranging from 10 to 10000, dependent on network size. Each data point is averaged over all the datasets for that particular value of R. In the majority of cases we obtain a precision very close to 1, with an average of 0.98. This means we are able to accurately certify large portions of the ranking vector and with our theory, do so with few



(a) Precision values on all graphs for various R.



(b) Histogram of P values for different networks.

Figure 2: Qualitative results using new stopping criterion.

false positives. Note that we examine the number of top vertices in graphs instead of a top percentage. From a data analysis standpoint, an analyst is more likely to be concerned with the top 100 vertices, for example, across multiple graphs rather than the top 5% of vertices (which will result in vastly different numbers of vertices dependent on graph size and in some cases may be impossible to parse given available human resources).

3.3 Perfect Ordering of Top

We have shown that we are successfully able to efficiently identify sets of top ranked vertices in networks for various set sizes. Experimentation shows that the theory is sound across several real-world networks. While the previous experiments are only concerned with returning the top set of vertices, here we impose the additional constraint of perfect ordering of this set. We not only want the most highly ranked vertices, but we also want them in the correct ordering as given by the exact solution. Recall the example of a web-Google graph given in Section 1. In this use case, it is important to ensure the ordering of these results is correct. We are able to apply the theory from Theorem 1 in this application and provide a guarantee on how many vertices we can accurately certify are in the correct ordering in the top of the ranking vector compared to the exact solution. In this case, we look at the gaps between successive vertices i and $i+1$ to ensure each pairwise comparison of vertices has the necessary gap to prove the correctness of the relative ordering. Running a solver to machine precision to identify top sets in networks cannot in fact provide any theoretical guarantee of how many vertices in the approximation are in the correct ordering compared to the exact solution. In this experiment, we are interested in finding P such that $P = argmax_i|b_i^{(k)} - b_{i+1}^{(k)}| > 2\epsilon_k$, where P is the number of vertices in the top of the vector in the correct order compared to the exact solution. We traverse the sorted ranking vector $\mathbf{b}^{(k)}$ after 10 iterations of conjugate gradient to find the first place where the gap of $2\epsilon_k$ is not satisfied. When this occurs, we know that the previous vertices are in the correct ordering since each pair-wise comparison of previous vertices satisfied the gap.

Figure 2b plots the distribution of P values for the different networks, with values of 0 omitted. Note the x-axis is on a log-scale. In most cases, we are able to accurately certify at least hundreds of vertices, with an average across all datasets of $P = 903$. For cases where we are only able to guarantee 1 or 0 vertices, we offer a possible explanation. If there are vertices with the same ranking at the top of the exact solution, our theory will not be able to go beyond this point because the necessary gap does not exist. Regardless, from a data analysis standpoint, the numbers of vertices able to be accurately certified in the exact order in the top validate our theory being used in this use case. For example, in a web-Google graph, a user will only be concerned with the top 75-100 results, meaning that relative ordering of these results is very important. Therefore our ability to accurately certify hundreds of vertices in the correct order is very applicable.

4 Conclusions

This work relates the two research areas of numerical accuracy of solvers and network analysis by understanding how the error in a solver affects the data analysis problem of ranking. By treating the problem of ranking vertices in a graph as understanding numerical accuracy in a linear solver, we present how the error in the numerical problem affects the solution to the original data analysis problem of ranking. Our aim in this work was to provide theoretical guarantees to bound the error in an approximate solution from an iterative method to the exact Katz Centrality scores of vertices in a network. Using this theory, we are able to identify the most central vertices with high confidence without accurately computing the centrality scores for every vertex and therefore reduce computation time.

The result of our analysis is a reduction in the number of iterations taken to solve the data analysis problem of ranking in graphs while maintaining a high precision rate in identifying top vertices. We demonstrate this on several real-world networks using conjugate gradient as the iterative method, giving high confidence that the important portion of the ranking is correct. We present experiments validating the theory as a stopping criterion that can be used in conjunction with any iterative solver, leading to significant algorithmic improvements. When using the theory to identify top ranked vertices we are able to do so with very few false positives. Finally, we also show perfect recall of the top vertices with respect to the exact solution is possible with our theory. The results from this paper can also be applied to any linear solver based ranking. Identifying top ranked vertices by Katz is one example in practice presented in this work, but the theory is generalizable to other linear algebra based ranking metrics. This paper draws the following quantitative conclusions:

- Our stopping criteria leads to an average of 3.07× reduction in iterations with a maximum of 5.14× reduction across several graphs when identifying the top 100 vertices.
- The average residual norm when we terminate at our stopping criterion is 10^{-4}, which provides a practical tolerance for iterative solvers compared to that of machine precision.
- Regression analysis of the performance of our method as a function of α shows that our method tends to reduce cost more on harder problems.
- We obtain a 0.98 precision when identifying the top R vertices (for $R \in [10, 10000]$) in a graph which shows that our method returns high quality results.
- Applying the theory to several real datasets shows we are able to guarantee that large portions of the ranking vector are in the correct ordering compared to the exact solution.

Experiments show that we are able to answer the data analysis question of identifying the top ranked vertices in the graph before needing to converge to machine precision to answer the numerical problem. Future work will study the impact of these guarantees in a personalized setting, specifically studying Katz scores from a specific seed set of vertices, and will extend our theory for directed graphs.

5 Acknowledgments

Eisha Nathan is in part supported by the National Physical Science Consortium Graduate Fellowship. This work is in part performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 with release number LLNL-CONF-728271.

References

[1] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Internet: Diameter of the world-wide web. *Nature*, 401(6749):130–131, 1999.

[2] Michele Benzi, Ernesto Estrada, and Christine Klymko. Ranking hubs and authorities using matrix functions. *Linear Algebra and its Applications*, 438(5):2447–2474, 2013.

[3] Francesco Bonchi, Pooya Esfandiar, David F Gleich, Chen Greif, and Laks VS Lakshmanan. Fast matrix computations for pairwise and columnwise commute times and katz scores. *Internet Mathematics*, 8(1-2):73–112, 2012.

[4] Ulrik Brandes and Christian Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(07):2303–2318, 2007.

[5] Kurt C Foster, Stephen Q Muth, John J Potterat, and Richard B Rothenberg. A faster katz status score algorithm. *Computational & Mathematical Organization Theory*, 7(4):275–285, 2001.

[6] KA Hawick and HA James. Node importance ranking and scaling properties of some complex road networks. 2007.

[7] Nicholas J Higham. *Functions of matrices: theory and computation.* Siam, 2008.

[8] Michiel E Hochstenbach. Probabilistic upper bounds for the matrix two-norm. *Journal of Scientific Computing*, 57(3):464–476, 2013.

[9] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[10] Jérôme Kunegis. Konect: the koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1343–1350. ACM, 2013.

[11] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection, June 2014.

[12] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[13] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[14] Yousef Saad. *Iterative methods for sparse linear systems.* Siam, 2003.

[15] RS Varga. Gershgorin and his circles in springer series in computational mathematics, 36, 2004.

[16] Dirk Werner. *Funktionalanalysis.* Springer, 2006.