Power Walk: Revisiting the Random Surfer

Laurence A. F. Park and Simeon Simoff
School of Computing, Engineering and Mathematics
University of Western Sydney, Australia
{l.park,s.simoff}@uws.edu.au

ABSTRACT

Measurement of graph centrality provides us with an indication of the importance or popularity of each vertex in a graph. When dealing with graphs that are not centrally controlled (such as the Web, social networks and academic citation graphs), centrality measure must 1) correlate with vertex importance/popularity, 2) scale well in terms of computation, and 3) be difficult to manipulate by individuals. The Random Surfer probability transition model, combined with Eigenvalue Centrality produced PageRank, which has shown to satisfy the required properties. Existing centrality measures (including PageRank) make the assumption that all directed edges are positive, implying an endorsement. Recent work on sentiment analysis has shown that this assumption is not valid. In this article, we introduce a new method of transitioning a graph, called Power Walk, that can successfully compute centrality scores for graphs with real weighted edges. We show that it satisfies the desired properties, and that its computation time and centrality ranking is similar to when using the Random Surfer model for non-negative matrices. Finally, stability and convergence analysis shows us that both stability and convergence when using the power method, are dependent on the Power Walk parameter β.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]; E.1 [DATA STRUCTURES]:Graphs and networks

General Terms: Algorithms, Design, Performance.

Keywords: PageRank, Random Surfer, Sentiment, Graph Centrality.

1. INTRODUCTION

Measures of graph centrality provide us with an identification of where the most important or popular vertices in a graph lie. For example, centrality over the Web graph tells us which Web pages are the most popular, centrality over a social network shows us which users are the most popular, and centrality over a citation network shows us which articles are the most popular.

When computing centrality over a graph that is not centrally controlled, we must make sure that the centrality scores cannot be easily manipulated for personal gain. Three methods of centrality have shown to achieve this (PageRank (Page et al., 1999), HITS (Kleinberg, 1999) and SALSA (Lempel and Moran, 2001)); of these three HITS and SALSA require an irreducible graph, and so cannot be computed for the whole graph. They are instead used on a subgraph dependent on the users query.

The assumption that a directed edge between two vertices implies an endorsement, is not always correct. Recent work on sentiment analysis for the Web (Miller et al., 2011), social networks (Tan et al., 2011), and academic citations (Teufel et al., 2006) has shown that edges in these graphs can have positive or negative weight, based on the text associated to the related vertices. Unfortunately, existing centrality methods are unable to use negative edge weights.

There have been attempts at computing centrality on sentiment graphs using PageRank. Wijaya and Bressan (2008) and Wu et al. (2009) computed the centrality on a real weighted sentiment graph by splitting the graph into the magnitude of its positive and negative graphs. The centrality was computed on both non-negative weighted graphs and analysed separately, losing any correlation between the polarised sentiment. Li et al. (2009) kept the sentiment scores between 0 and 1, which gives the non-ideal environment that no opinion (a missing edge) was equivalent to negative sentiment.

In this article, we introduce Power Walk, a new method of transitioning a random walk on a graph that allows not only positive and neutral edges, but also negative edges. We show that using Power Walk provides the desired centrality measurement properties for non-negatively weighted graphs, while also behaving well for graphs with real valued weights. Thus making Power Walk a powerful tool for the analysis of popularity on the Web, in social networks and in the academic community. Our contributions are:

- definition of the Power Rank transition probabilities and its solution using the Power method (Section 3.1 and 3.2),
- a stability and convergence analysis of the Power method using Power Walk transitions (Section 3.4), and
- an analysis of the centrality and timing similarity of Eigenvalue centrality using the Random Surfer and Power Walk transition probabilities (Section 4).

The article will proceed as follows: Section 2 gives an overview of current centrality methods, Section 3 defines the Power Walk probability transitions and its features, and Section 4 examines the similarity between centrality computed using Power Walk and other centrality methods.

2. GRAPH CENTRALITY

The centrality score of a vertex within a graph provides us with an indicator of the importance or popularity of the vertex in the graph. When measuring the centrality of the set of vertices in scale

ADCS '13, December 05 - 06 2013, Brisbane, QLD, Australia
Copyright 2013 ACM 978-1-4503-2524-0/13/12 ...$15.00.

free graphs, such as the World Wide Web, social networks and the academic citation graph, we require the following properties:

1. the centrality score of a vertex correlates with the importance or popularity of the vertex,
2. it is computable for large graphs in reasonable time, and
3. it is difficult for individuals to manipulate for personal gain.

Of these three points, the first is ensuring that we are measuring centrality. The second is required to allow the centrality score to be computed for large graphs, such as those found in the Web, social networks and academic citations. The third point ensures that individuals cannot bias the centrality score, thus keeping it fair for all. Traditional methods of computing graph centrality, such as Degree Centrality, Closeness Centrality, Betweenness Centrality and Eigenvector Centrality, are not suitable for this task.

Degree Centrality is a simple measure of graph centrality, where the centrality score of a vertex is equal to the degree (in-degree or out-degree) of the vertex. The Degree centrality score can be easily computed for any graph size, but unfortunately, it is also very easy to manipulate. The centrality score can be increased by adding more links, to or from a vertex, allowing individuals full control over their centrality score.

The centre of a circle is the point that has the shortest distance to all other points in the circle; *Closeness Centrality* is defined similarly. The Closeness centrality of a point is the inverse of the sum of shortest paths to all other points. If there is no path to a given vertex, then the total number of vertices is used in its place. To compute the Closeness centrality, we must compute the shortest paths between all vertices, which is computationally expensive.

Centrality can also be defined based on the utility of the vertex. The *Betweenness Centrality* score of a vertex is the sum of the fraction of all shortest paths in which the given vertex lies. If a vertex is useful, it should lie on the shortest path between two other vertices. To compute Betweenness centrality, we not only need to compute the shortest paths between all pairs of vertices, but also identify which vertices lie on these paths. Therefore, it is more computationally expensive than computing Closeness centrality.

Eigenvector Centrality poses centrality as a recursive problem. It states that the centrality score of a vertex is a function of the centrality scores of the vertices connected with an incoming edge to the vertex of interest. The centrality scores are found in the dominant eigenvector of the graph transition matrix. Computing the eigenvalue decomposition of the transition matrix is computationally expensive, but it has been shown that the dominant eigenvector can be computed using an iterative method (the Power method) as long as the all the elements of the transition matrix are positive or all elements are non-negative and the graph is irreducible. (Farahat et al., 2006)

The Eigenvector centrality for a given vertex can be manipulated by adding incoming edges to the vertex. The centrality score for a given vertex is added to any other vertices connected with out going edges. Therefore, if vertex v has an Eigenvector centrality of c, we can increase any other vertex's Eigenvector centrality by c by linking to it. *Stochastic Eigenvector Centrality* removes this manipulation problem by normalising the outdegree weight of each vertex so that the outdegree weight sums to 1. When linking to other vertices, Stochastic Eigenvector Centrality allows each vertex to distribute its centrality score to other vertices. If vertex v_i links to vertex v_j only, vertex v_i provides its whole centrality score to vertex v_j. If vertex v_i links to many other vertices, they all obtain a fraction of vertex v_i's centrality score.

Stochastic Eigenvector Centrality satisfies all three requirements that we posed. Unfortunately, scale free networks do not have the required properties of either 1) the adjacency matrix is positive, or 2) the adjacency matrix is non-negative and irreducible. Scale-free graphs (Barabási, 2009; Barabási et al., 2000) found in evolving networks such as the World Wide Web, social networks and citation graphs, are sparse graphs, meaning that the proportion of existing edges is low. Therefore, there are many zeros in their adjacency matrices. Also, these networks are not guaranteed to be irreducible. Irreducibility implies that there is a path from each vertex to each other vertex, which is unlikely in scale-free networks.

The Random Surfer model (Page et al., 1999; Brin and Page, 1998), was developed to allow the use of Stochastic Eigenvalue Centrality for scale-free networks. The Random Surfer model adjusts the values in the graph's adjacency matrix to ensure all values are positive and hence the graph is irreducible. The name *Random Surfer* implies a Web user navigating through the Web by randomly selecting and following hyperlinks within the currently displayed Web page. The random surfer model also provides a small probability that the Web user will jump to a randomly chosen Web page. The eigenvector centrality score using the Random Surfer transition matrix is more commonly known as *PageRank*. PageRank is computed as:

$$\mathrm{PR}(v_i) = \alpha \sum_{v_j \in \mathrm{In}(v_i)} \frac{\mathrm{PR}(v_j)}{|\operatorname{Out}(v_j)|} + \frac{1-\alpha}{n} \qquad (1)$$

where $\mathrm{In}(v)$ and $\mathrm{Out}(v)$ is the set of vertices with edges directed into and out of v respectively, $|x|$ is the cardinality of set x (meaning $|\operatorname{Out}(v)|$ is the out degree of vertex v and $|\operatorname{In}(v)|$ is the in-degree), $\mathrm{PR}(v)$ is the PageRank of vertex v, n is the number of vertices in the graph, and $\alpha \in [0, 1]$. We can write equation 1 in matrix notation as:

$$\vec{p} = \alpha A D_A^{-1} \vec{p} + \frac{1-\alpha}{n} \vec{1}$$

where A is the graph adjacency matrix containing the elements $a_{i,j}$ ($a_{i,j} = 1$ if there is an edge from vertex v_j to v_i and $a_{i,j} = 0$ otherwise), D_A is a diagonal matrix containing $|\operatorname{Out}(v_j)|$, $\vec{p}$ contains the elements $\mathrm{PR}(v_i)$, and $\vec{1}$ is the vector with each element 1. We can add the constraint that the sum of $\mathrm{PR}(v_i)$ is one without affecting the result, since PageRank scores are relative, allowing us to write $\vec{1} = O\vec{p}$, where O is an $n \times n$ matrix with every element 1. Using this knowledge, we can write the PageRank equation as:

$$\begin{aligned} \vec{p} &= \alpha A D_A^{-1} \vec{p} + \frac{1-\alpha}{n} O \vec{p} \\ &= (\alpha A D_A^{-1} + \frac{1-\alpha}{n} O) \vec{p} = T \vec{p} \end{aligned}$$

where T is the Random Surfer probability transition matrix (Park and Ramamohanarao, 2011, 2007).

A problem occurs when computing PageRank if there are vertices with zero out degree. If we examine the probability transition matrix, the columns will sum to one for all vertices that have an out degree of greater than 0, but the column sums to $1 - \alpha$ when the associated vertex out degree is zero. A fix for this problem is to renormalise the column (divide by $1 - \alpha$), which is equivalent to adding outgoing edges from the associated vertex, to every vertex in the graph). This gives us the probability transition matrix T, containing the elements:

$$t_{i,j} = \begin{cases} \frac{\alpha}{|\operatorname{Out}(v_j)|} + \frac{1-\alpha}{n} & \text{if } v_j \in \mathrm{In}(v_i) \\ \frac{1-\alpha}{n} & \text{if } v_j \notin \mathrm{In}(v_i) \text{ and } |\operatorname{Out}(v_j)| > 0 \\ \frac{1}{n} & \text{if } |\operatorname{Out}(v_j)| = 0 \end{cases}$$

To compute the PageRank of a given graph, we construct the transition probability matrix T and compute the eigenvector associated to the eigenvalue 1.

3. RANDOM WALKS WITH REAL VALUED WEIGHTS

We saw in the previous section that Stochastic Eigenvector centrality is a useful measure of graph centrality since it correlates with importance/popularity of vertices, is computable for large graphs using the Power method, and it is difficult to manipulate. For it to be computable for scale free graphs, we use the Random Surfer probability transitions, which 1) adds a random jump and 2) renormalises the probabilities for vertices with no out-degree.

So far we have assumed that an edge from vertex v_i to vertex v_j implies that vertex v_i endorses v_j. When examining the Web, social networks and academic citations, we find that this is not always the case. Social networks allow users to provide positive and negative votes towards items in the network, and sentiment analysis now allows us to examine the polarity of Web links and citations, allowing us to see if the directed edge implies a users approval or disapproval.

In this section, we will examine how we can include negatively weighted edges in our centrality computation. When considering a random walk over a graph with positive and negative edge weights, a positive edge weight implies that there is an attraction to walk to the connecting vertex, while a negative weighted edge implies repulsion. We want to ensure that, given a vertex and a set of outward directed edges:

1. The probability of transitioning to a vertex using a positively weighted edge is greater than the probability of transitioning to a disconnected vertex (no direct edge to the vertex).

2. The probability of transitioning to a vertex using a positively weighted edge is greater than the probability of transitioning to a vertex using a negatively weighted edge.

3. The probability of transitioning to a disconnected vertex is greater than the probability of transitioning to a vertex using a negatively weighted edge.

If we treat an absent directed edge as a directed edge with zero weight, and consider only graphs with edge weights 1, 0 and -1, we can adjust the Random Surfer model to have an additional term, allowing for the -1 edges:

$$T = \alpha_+ A_{+1} D_{A_{+1}}^{-1} + \alpha_- A_{-1} D_{A_{-1}}^{-1} + (1 - \alpha_+ - \alpha_-)E \quad (2)$$

where A_{+1} is the adjaceny matrix containing only the positive edges, A_{-1} is the adjaceny matrix containing only the negative edges, $D_{A_{+1}}$ and $D_{A_{-1}}$ are the diagonal out degree matrices of A_{+1} and A_{-1} respectively, α_+ and α_- are greater than zero and $\alpha_+ + \alpha_- < 1$.

The probability transition matrix in equation 2 satisfies requirements 1 and 2, but not 3. We can see that even though we assign a small probability of traversing to a vertex using a negatively weighted edge, there is also the chance of randomly jumping to the vertex. This makes the probability of arriving at a vertex, at the end of the negatively weighted edge, greater than the probability of traversing to an unattached vertex (using an edge of weight 0) .

We can further adjust the Random Surfer probability transition matrix to be:

$$T = \alpha_+ A_{+1} D_{A_{+1}}^{-1} - \alpha_- A_{-1} D_{A_{-1}}^{-1} + (1 - \alpha_+ + \alpha_-)E \quad (3)$$

This form satisfies all three requirements, as long as $\alpha_- < 1 - \alpha_+ + \alpha_- < \alpha_+$, or $0 < \alpha_- < 2\alpha_+ - 1$. We must also make sure that the final probability transition matrix T is positive, meaning that $(1 - \alpha_+ + \alpha_-)/n - \alpha_-/\text{negative\_outdegree}(v) > 0$ or:

$$0 < \alpha_- < (1 - \alpha_+)/(n - 1)$$

Unfortunately, this model has the strange effect that the more negatively weighted out going edges from a vertex, the more likely that they will be followed. Note that if there are no positive or negative edges from a vertex, we must either remove them, or add outward directed edges to every vertex.

We can see that the simple Random Surfer model has become complicated, has an unwanted behaviour, and the parameter α_- is restricted, due to the addition of the random jump to every edge.

3.1 Power Walk

Rather than adapting the Random Surfer model, we will take a step back and examine how we want our model to behave. We have three cases for each edge ($+1$, 0, -1) and there are a set of n edges for a given vertex. If we define the probability of following an edge of weight 0 as x, we can define that we want the probability of following an edge of weight 1 to be βx, meaning we are β times more likely to follow an edge to a given vertex, than to go to a given vertex with no connecting edge. We can also define the probability of following an edge of weight -1 as x/β, meaning we are β times more likely to jump to a page with no connecting edge, than to follow an edge with a negative weight. Using these probabilities, we can write them as $P(v_i \to v_j) = x\beta^{a_{i,j}}$, where $a_{i,j}$ is the weight of the directed edge from vertex v_j to vertex v_i (if there is no edge from vertex v_j to vertex v_i, the edge weight is zero). The probability distribution must sum to 1, therefore:

$$\sum_{j=1}^{n} P(v_i \to v_j) = \sum_{j=1}^{n} x\beta^{a_{i,j}} = 1$$

$$\Rightarrow x = 1/\sum_{j=1}^{n} \beta^{a_{i,j}}$$

This gives us the probability transition matrix values:

$$t_{i,j} = \frac{\beta^{a_{i,j}}}{\sum_{j=1}^{n} \beta^{a_{i,j}}}$$

where β is a positive real value. We call this the *Power Walk* probability transition matrix, given as:

$$T = BD_B^{-1}$$

where $B = \beta^A$, and D_B is a diagonal matrix containing the column sums of B, and hence when used with eigenvalue centrality, we have the *Power Walk* model.

Using the Power Walk transition probabilities, we satisfy all three properties for positive and negative weights, and we also ensure that that probability transition matrix is positive and stochastic for any graph containing real value weighted edges, meaning that an eigenvalue centrality solution exists, is associated to the largest eigenvalue $\lambda_1 = 1$, and we can use the Power method iterations to compute the eigenvector.

Note also that using the Power Walk probability transition matrix, we do not have to define a special case for vertices with an out degree of zero (which is required when using the Random Surfer model). In this case, the probability of transitioning to any other vertex is $\beta^0/\sum_{j=1}^{n} \beta^0 = 1/n$.

3.2 Power method solution

A useful property of the Random Surfer model is that it can be kept in a sparse form when using the Power method to obtain the stationary distribution. During each iteration of the power method, we compute:

$$\vec{p}_{i+1} = (\alpha A D_A^{-1} + (1 - \alpha)E)\vec{p}_i$$

where $\vec{p}_i$ is the state probability distribution after the ith iteration. We know that the sum of $\vec{p}_i$ is 1, therefore the product $E\vec{p}_i$ gives a vector containing $1/n$. This simplifies the power method iterations to:

$$\vec{p}_{i+1} = \alpha A D_A^{-1} \vec{p}_i + \frac{1-\alpha}{n}$$

For Web graphs, or graphs found in social networks or academic citations, the matrix AD_A^{-1} is sparse and can be stored in a compact form. Each iteration of the power method requires a matrix multiplication with a sparse matrix and the addition of a constant. Both of these operations can be performed efficiently on sparse graphs.

It is common to find vertices with zero out degree; in this case, we must adjust the transition matrix so that we have equal probability of transitioning to any other vertex. To do this, we introduce the matrix Z that contains all zeros, except for the columns that associate to vertices with zero out degree, these columns contain $1/n$.

$$\vec{p}_{i+1} = \alpha (A + Z) D_{A+Z}^{-1} \vec{p}_i + \frac{1-\alpha}{n}$$

If we add the matrices A and Z, we lose the sparsity of A, so we will keep them separate to obtain:

$$\vec{p}_{i+1} = \alpha A D_A^{-1} \vec{p}_i + \alpha \vec{z}' \vec{p}_i + \frac{1-\alpha}{n}$$

where $\vec{z}$ is a row from Z (all rows in Z are equal), and any zeros in the diagonal of D_A are replaced with 1.

Using the Power Walk method, we find that the matrix B is not sparse, since all of the zeros from A have been converted to ones. To keep B sparse, we must split it into $B - O$ and O, where O is a matrix of ones.

$$\begin{aligned}
\vec{p}_{i+1} &= B D_B^{-1} \vec{p}_i \\
&= (B - O + O) D_B^{-1} \vec{p}_i \\
&= (B - O) D_B^{-1} \vec{p}_i + O D_B^{-1} \vec{p}_i
\end{aligned}$$

where $B = \beta^A$, and $(B - O)D_B^{-1}$ is sparse. Again we find that the second term is an inner product:

$$\vec{p}_{i+1} = (B - O) D_B^{-1} \vec{p}_i + \vec{\delta}_B' \vec{p}_i \tag{4}$$

where $\vec{\delta_B}$ is the diagonal of D_B^{-1}. We have presented the algorithm required to compute the power method using a Power Walk in Algorithm 1.

Algorithm 1 Power method using the Power Walk transition matrix. The algorithm is designed to ensure that all matrices remain sparse throughout the computation. A is the adjacency matrix containing real values, β is positive, and $\vec{p}$ is the stationary distribution.

```
procedure POWER WALK($A, \beta$)
    $B \leftarrow \beta^A - 1$                          ▷ If $A$ is sparse, so is $B$.
    $\vec{\delta}_B \leftarrow (\vec{1}'B)^{-1}$        ▷ $\vec{\delta}_B$ is the inverse column sums of $B$.
    $D_B^{-1} \leftarrow \text{diag}(\vec{\delta}_B)$
    $T \leftarrow B D_B^{-1}$                           ▷ If $B$ is sparse, so is $T$.
    $\vec{p}_{\text{new}} \leftarrow \vec{1}/n$         ▷ Uniform initialisation.
    $\vec{p} \leftarrow \vec{0}$                        ▷ Initialise with the zero vector.
    while $\|\vec{p}_{\text{new}} - \vec{p}\| > \epsilon$ do
        $\vec{p} \leftarrow \vec{p}_{\text{new}}$
        $\vec{p}_{\text{new}} \leftarrow T\vec{p} + \vec{\delta}_B'\vec{p}$
    end while
    return $\vec{p}$
end procedure
```

3.3 Equivalence of the Power Walk and Random Surfer model

For any unweighted adjacency matrix, we can express the Power walk probability transition matrix in the form of a random surfer transition matrix. An unweighted adjacency matrix contains only values of 1 and 0, therefore the unnormalised Power Walk transition matrix will contain values of β and 1. The random surfer model has the form:

$$T = \alpha A D_A^{-1} + (1 - \alpha) E$$

where E is an $n \times n$ matrix containing $1/n$.

The Power Walk transition matrix can be rearranged to have the same form as the Random Surfer model, when using an unweighted adjacency matrix:

$$\begin{aligned}
T &= B D_B^{-1} \\
&= (B + O - O) D_B^{-1} \\
&= (B - O) D_B^{-1} + O D_B^{-1} \\
&= (B - O) D_B^{-1} + E(n D_B^{-1}) \\
&= (\beta - 1) A D_B^{-1} + E(n D_B^{-1}) \\
&= A D_A^{-1} D_A (\beta - 1) D_B^{-1} + E(n D_B^{-1})
\end{aligned}$$

where O is an $n \times n$ matrix of ones and I is the identity matrix. Since AD_A^{-1}, E and T are all stochastic matrices, we find that:

$$(\beta - 1) D_A D_B^{-1} = I - n D_B^{-1}$$

giving us the Random Surfer form:

$$T = A D_A^{-1} \Gamma + E(I - \Gamma) \tag{5}$$

where the diagonal matrix $\Gamma = I - nD_B^{-1}$ is equivalent to the Random Surfer model parameter α.

3.4 Convergence and Stability

If we order the eigenvalues in terms of their magnitude, we know, by the Perron-Frobenius theorem that the largest eigenvalue λ_1 of a positive stochastic matrix is 1 (Farahat et al., 2006). The second eigenvalue λ_2, by our definition, is greater than or equal to all remaining eigenvalues.

The rate of convergence of the power method is dependent on the magnitude of the second largest eigenvector $|\lambda_2|$ of the probability transition matrix. The smaller $|\lambda_2|$, the faster the power method converges to a solution (Bryan and Leise, 2006; Haveliwala and Kamvar, 2003). This can be identified by writing the probability transition matrix T in terms of its eigenvalue decomposition.

$$T = V \Lambda V^{-1}$$

where V is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues. The Power method after k iterations gives:

$$\begin{aligned}
\vec{p}_k &= T^k \vec{p}_0 \\
&= (V \Lambda V^{-1})^k \vec{p}_0 \\
&= V \Lambda^k V^{-1} \vec{p}_0
\end{aligned}$$

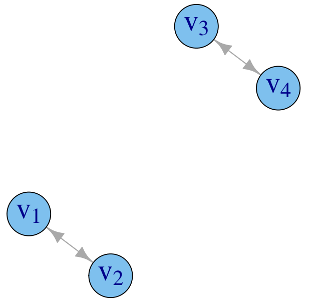

Figure 1: A small graph containing two irreducible closed subgraphs.

weighting the ith eigenvector by λ_i^k. All eigenvalues are less than or equal to zero, so the smaller the value of $|\lambda_i|$, the faster it approaches zero as k approaches infinity. When $|\lambda_2|^k$ is zero, T^k is a rank one matrix determined by the eigenvector associated to λ_1 (the stationary distribution). Therefore we can use $|\lambda_2|$ to judge the rate of convergence of the Power method.

Haveliwala and Kamvar (2003) showed that the second eigenvalue has magnitude $|\lambda_2| \leq \alpha$, for any matrix of the form $\alpha A + (1 - \alpha)E$ where A a stochastic matrix, E is a rank one stochastic matrix, and $0 \leq \alpha \leq 1$. They also showed that if A has at least two irreducible closed subsets, then $|\lambda_2| = \alpha$. A graph containing an irreducible closed subgraph implies that there is a subgraph which is irreducible, and there are no edges out of the subgraph to the remainder of the graph. Graphs such as the Web satisfy this requirement (Broder et al., 2000) and it is highly likely that social network graphs and academic citation graphs also satisfy this requirement, since they all evolve similarly (An et al., 2004; Kumar et al., 2010).

Ng et al. (2001a,b) examined the stability of the stationary distribution with respect to the probability transition matrix eigenvalues. They showed that a given stationary distribution, and hence set of centrality scores, is unstable if $|\lambda_2|$ is close to 1. The reason being, if there are any changes in the probability transition matrix (e.g. an edge added or removed), this may increase the second eigenvalue, swapping it with the first eigenvalue. This swap also results in a swap of the associated eigenvectors, which may drastically change the centrality scores.

This analysis of convergence rate and stability shows that we should reduce the magnitude of the second eigenvalue $|\lambda_2|$, meaning we should set α to be close to 0 when using the Random Surfer model. We should also remember that the choice of α reflects the probability of the random surfer choosing to follow an edge, over randomly jumping. Therefore a low value of α would cause all of the centrality scores to become uniform, making them meaningless. We find that most common setting for α is 0.85 (Haveliwala and Kamvar, 2003).

To examine the convergence rate and stability of the power method using a Power Walk transition matrix, we must examine the magnitude of the Power Walk transition matrix's second eigenvalue $|\lambda_2|$. We showed in equation 5 that the Power Walk transition matrix has the same form as the Random Surfer model when A contains binary values, where instead of a constant α, we have a diagonal matrix Γ.

To examine the relationship between Γ and λ_2, we generated a graph containing 4 vertices and two irreducible subsets (shown in Figure 1). If we use the Random Surfer probability transition matrix with $\alpha = 0.85$, we obtain the eigenvalues $\lambda_1 = 1$, $\lambda_2 = -0.85$, $\lambda_3 = 0.85$ and $\lambda_4 = -0.85$. We can see the relationship $|\lambda_2| = \alpha$. Using the Power Walk probability transition matrix, with $\beta = 10$, we obtain the eigenvalues $\lambda_1 = 1$, $\lambda_2 = 0.6923$, $\lambda_3 = -0.6923$ and $\lambda_4 = 0.6923$ and the diagonal of the matrix Γ containing $(0.6923, 0.6923, 0.6923, 0.6923)$. This result is encouraging since the matrix Γ did contain the value of $|\lambda_2|$. For this case, the diagonal matrix Γ contained the same repeated value, making it equivalent to a constant times an identity matrix, this was due to the adjacency matrix containing only one nonzero element per column.

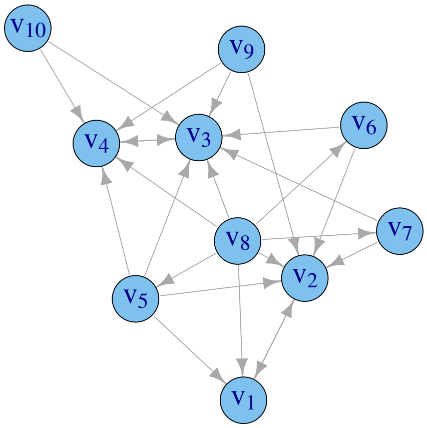

Figure 2: A slightly larger graph containing two irreducible closed subgraphs.

Out degree	0	1	2	3	4
γ_i	0.0000	0.6666	0.7999	0.8571	0.8888
Out degree	5	6	7	8	9
γ_i	0.9090	0.9230	0.9333	0.9411	0.9473

Table 1: The elements of the diagonal matrix Γ associated to vertices with the given outdegree in a graph containing 1000 vertices.

If we examine the graph in Figure 2, which has vertices with different out degree (meaning the elements of Γ will vary), we find the first four eigenvalues of the Random Surfer probability transition matrix are $\lambda_1 = 1$, $\lambda_2 = 0.85$, $\lambda_3 = -0.85$ and $\lambda_4 = -0.85$, using $\alpha = 0.85$. The first four eigenvalues of the Power Walk probability transition matrix are $\lambda_1 = 1$, $\lambda_2 = -0.4737$, $\lambda_3 = -0.4737$ and $\lambda_4 = 0.4737$, using $\beta = 10$. We compute the diagonal matrix Γ to contain the values $(0.4737, 0.4737, 0.4737, 0.4737, 0.6429, 0.7297, 0.6429, 0.6429, 0.64291, 0.7297)$, showing again the $|\lambda_2|$ is an element of Γ.

The values of the diagonal matrix Γ are dependent on the number of vertices in the graph n, and the out-degree of each vertex. We found that altering the out-degree distribution of the graph does not affect λ_2, but changing the size of the irreducible subgraphs does.

Table 1 contains the possible values of the diagonal of matrix Γ associated to vertices with out degree from 0 to 9, where $n = 1000$. When generating scale free graphs with 1000 vertices, containing irreducible subgraphs with two vertices, the magnitude of the second eigenvalue of the Power Walk transition matrix $|\lambda_2| = 0.6666$. When containing irreducible subgraphs with three vertices, $|\lambda_2| = 0.7999$. When containing irreducible subgraphs with four vertices,

$|\lambda_2| = 0.8571$. When containing irreducible subgraphs with five vertices, $|\lambda_2| = 0.8888$. When comparing these results with Table 1, it seems that we have the relationship $|\lambda_2|$ is the element of Γ associated to out degree k, where the graph has irreducible subgraphs containing $k + 1$ elements. Therefore, by rearranging Γ, we obtain the relationship:

$$|\lambda_2| = 1 - n/(n + k(\beta - 1))$$

where n is the number of vertices in the graph and k is the size of the irreducible subgraphs. As we increase β, we increase λ_2, therefore to increase stability and decrease the number of iterations for convergence of the power method, we should decrease β. We know that the Random Surfer parameter $\alpha = |\lambda_2|$, therefore, we can compute β as:

$$\beta = \frac{n\alpha}{k(1 - \alpha)} + 1 \qquad (6)$$

We need to examine this relationship further, since it is not obvious how the second eigenvalue behaves when the irreducible subgraphs are of different sizes.

4. POWER WALK PROPERTIES

We have shown that the Power Walk transition probabilities provide us with a method of computing the centrality of positive and negatively weighted graphs. In this section, we will examine how computing eigenvector centrality with Power Walk transition probabilities compares with existing centrality methods, and how altering the graph affects the measure of centrality.

Computing the centrality for a real data set would provide us with a list of vertices, but not show us how the algorithm performs. In this section, we will examine the behaviour of the Power Walk on randomly generated graphs having controlled properties. The random graphs are generated using the Barabási-Albert model to simulate a Web graph with in-degree power of 2.1 and out-degree power of 2.7. The number of vertices generated will be stated with the experiment. We will use the Random Surfer parameter $\alpha = 0.85$ and compute the Power Walk parameter β using equation 6 to obtain comparable results. If we can show that the Power Walk transitions are equivalent to Random Surfer transitions for non-negatively weighted graphs, and we can show that the Power Walk transition behave well for real weighed graphs, then we have shown there is a benefit in using the Power Walk transitions.

4.1 Rank Similarity

We will first examine how the stationary distribution using Power Walk transition compares to the mentioned centrality methods. For this experiment the edges are unweighted (an edge implies a weight of 1, no edge a weight of 0), since we cannot compare these methods using graphs with negative weights.

To perform the experiment, we generated 30 random graphs with 1000 vertices and computed the centrality using each of the five methods. The ranking of the set of vertices using each method were compared using Kendall's distance, and the mean and standard deviation results are shown in Table 2. Of the five methods, Eigenvector (RS) and Eigenvector (PW) stand for Eigenvector centrality using the Random Surfer and Power Walk transition respectively.

Kendall's distance is the number of discordances between two ranked lists. A Kendall's distance of 5 implies that it takes 5 adjacent transpositions to change one of the rankings into the other. The maximum distance that can be obtained is $n(n - 1)/2$, where n is the number of items in the list.

We can see that the eigenvector centrality computed using Power Walk transition is very similar to the eigenvalue centrality using Random Surfer transisition (PageRank), having an average Kendall's distance of just under 5, with similar standard deviation. The next closest pair is Closeness and Indegree having a mean Kendall's distance just under 50. All other pairs have a mean Kendall's distance greater than 100. Comparing the similarity of all of these graph centrality methods gives us in indication of the similarity between the Random Surfer and Power Walk models. We find that they are relatively close, meaning that if we used Power Walk transition in the place of Random Surfer transition, we would notice little change. We also examined the rank similarity using other graph sizes and found similar results.

4.2 Computation Time

We compared the time to compute the Power Walk centrality score and Random Surfer centrality scores using the power method, by generating 30 random graphs of containing 10^1 to 10^7 vertices. The mean and standard deviation computation time is shown in Table 3.

We can see that the mean and standard deviation computation times are very similar and that they increase linearly with the number of vertices after the 10^4 mark. The Power Walk stationary distribution does require more time, this is likely to caused by the additional inner product in equation 4.

4.3 Power Iterations

We showed earlier that the number of iterations of the Power method is dependent on the Random Surfer paramter α. In this section, we will examine the effect of the parameters α and β on the number of iterations required for convergence of the Power method using the Random Surfer and Power Walk transition.

To examine the number of iterations, we generate 30 random graphs containing 10^4 to 10^7 vertices and measured the number of iterations of the Power method required for convergence when using Random Surfer and Power Walk transitions. For each graph size, we varied the parameter α and computed β using equation 6. The mean and standard deviation results are shown in Table 4.

It is interesting to see that the number of iterations of the Power method is independent on the number of vertices in the graph. As we increase the vertex count, there is little change in the mean iterations and no correlation with standard deviation iteration change.

The mean and standard deviation iteration count seem to have an exponential relationship with α, showing the time benefit associated to having a smaller α. We also see that the mean number of iterations required when using Power Walk transition is one more than when using Random Surfer transition, but the standard deviation of the iteration count have little difference. These results show us that our equation for computing β allows us to obtain comparable timing results for positively weighted graphs. We are unsure why there is a mean increase of one iteration when using Power Walk iterations, and will examine this difference in future work.

4.4 Negative edge effect

In this section, we will examine the effect of negative edges on the number of iterations required for convergence of the Power method. When examining graphs with negative edges, we can examine only the Power Walk method (since the other four centrality methods are suited only for non-negative edges).

We generated 30 random graphs containing 1000 vertices with a proportion x of randomly chosen edges assigned the weight of -1. This was performed for proportions from 0 (no negative edges) to 1 (no positive edges). The mean and standard deviation iterations are provided in Table 5.

The results show that the mean iterations reduced as the proportion of negative edges increased, and the standard deviation rises

	Indegree		Closeness		Betweenness		Eigenvector (RS)		Eigenvector (PW)	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Indegree	0.00	0.00	49.57	34.92	195.90	75.08	280.90	123.01	276.73	121.54
Closeness	49.57	34.92	0.00	0.00	263.17	104.47	415.83	252.23	410.40	250.14
Betweenness	195.90	75.08	263.17	104.47	0.00	0.00	353.57	139.65	354.20	140.08
Eigenvector (RS)	280.90	123.01	415.83	252.23	353.57	139.65	0.00	0.00	4.93	4.98
Eigenvector (PW)	276.73	121.54	410.40	250.14	354.20	140.08	4.93	4.98	0.00	0.00

Table 2: The mean and standard deviation Kendall's distance between five graph centrality methods over 30 randomly generated scale free graphs containing 1000 vertices.

Method	Number of Vertices						
	10^1	10^2	10^3	10^4	10^5	10^6	10^7
Mean Computation Time (seconds)							
Random Surfer	0.128	0.127	0.137	0.179	0.741	6.265	61.262
Power Walk	0.133	0.132	0.144	0.194	0.925	7.565	77.447
Standard Deviation Computation Time (seconds)							
Random Surfer	0.016	0.007	0.006	0.009	0.058	0.266	3.317
Power Walk	0.024	0.007	0.015	0.009	0.080	0.296	10.389

Table 3: Mean and standard deviation computation times from 30 random graphs of seven different sizes (10^1 to 10^7), when computing eigenvector centrality using the power method.

and then falls. An interesting feature of the Power Walk method is if a constant is added to a column of the adjacency matrix, it does not affect the stationary distribution. We can see this using the following:

$$t_{i,j} = \frac{\beta^{e_{i,j}+x}}{\sum_{j=1}^{n} \beta^{e_{i,j}+x}} = \frac{\beta^{e_{i,j}} \beta^{x}}{\sum_{j=1}^{n} \beta^{e_{i,j}} \beta^{x}} = \frac{\beta^{e_{i,j}}}{\sum_{j=1}^{n} \beta^{e_{i,j}}}$$

Therefore if we have an adjacency matrix A containing 0s and 1s, and change all of the 1s to -1s, it is equivalent to changing A to $1 - A$, which also has the effect of increasing k in equation 6. Increasing k provides us with a smaller β and hence lowers the number of iterations requires. Having a fraction of the edges as negative would have a partial effect on β and so would explain the gradual reduction in iterations as the proportion of negative edges increases. The exact relationship between the magnitude of the second eigenvalue and β needs to be investigated in later work.

5. CONCLUSION

Measures of graph centrality provide us with an indication of the importance/popularity of each vertex. To be an effective measure of centrality for uncontrolled graphs such as the World Wide Web, social networks and academic citation graphs, three criteria must be satisfied 1) the score correlates with vertex importance/popularity, 2) computation of the score scales well to large graphs, and 3) the scores are difficult to manipulate by individuals. Eigenvalue Centrality was modified to produce the Random Surfer model to satisfy these three criteria.

When analysing the Web, social networks and academic citations, we assume that each directed edge from a vertex is an endorsement to the vertex linked to. Sentiment analysis of these graphs has shown that this assumption is not always true and that edges can have negative weight. Unfortunately, the existing measures of centrality are limited to graphs with non-negative edge weights.

In this article we introduce Power Walk, a novel method of computing graph transition probabilities from graphs with real weighted edges. We show that it satisfies the desired properties, and that its computation time and centrality ranking is similar to when using the Random Surfer model to obtain PageRank for non-negative matrices.

We also examined the convergence and stability of the Power Walk centrality scores and found that they are a function of the Power Walk parameter β, and can be estimated using properties of the probability transition matrix.

References

Y. An, J. Janssen, and E. E. Milios. Characterizing and mining the citation graph of the computer science literature. *Knowledge and Information Systems*, 6(6):664–678, 2004.

A.-L. Barabási. Scale-free networks: a decade and beyond. *Science*, 325 (5939):412–413, 2009.

A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69–77, 2000.

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer networks*, 33(1):309–320, 2000.

K. Bryan and T. Leise. The $25,000,000,000 eigenvector: The linear algebra behind google. *Siam Review*, 48(3):569–581, 2006.

A. Farahat, T. LoFaro, J. C. Miller, G. Rae, and L. A. Ward. Authority rankings from hits, pagerank, and salsa: Existence, uniqueness, and effect of initialization. *SIAM Journal on Scientific Computing*, 27(4):1181–1201, 2006.

T. Haveliwala and S. Kamvar. The second eigenvalue of the google matrix. *Stanford University Technical Report*, 2003.

J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online

Method	Vertex Count	Random Surfer parameter (α)								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Mean iterations before convergence										
Random Surfer	10^4	6.000	8.267	10.833	14.233	18.533	25.100	36.133	56.933	119.767
Power Walk		7.000	9.567	11.967	15.400	19.567	26.200	37.167	57.933	120.900
Random Surfer	10^5	6.000	8.333	10.867	14.233	18.867	25.467	36.400	57.233	120.233
Power Walk		7.000	9.633	12.000	15.300	19.933	26.500	37.433	58.200	121.333
Random Surfer	10^6	5.967	8.300	10.967	14.400	18.633	25.133	36.300	56.767	121.367
Power Walk		7.000	9.733	12.000	15.533	19.633	26.233	37.233	57.800	122.333
Random Surfer	10^7	6.000	8.133	10.967	14.100	19.000	25.067	36.200	57.500	117.733
Power Walk		6.933	9.767	12.000	15.467	19.967	26.300	37.233	58.533	118.733
Standard deviation iterations before convergence										
Random Surfer	10^4	0.000	0.640	0.531	0.774	1.008	0.845	1.106	2.406	6.383
Power Walk		0.000	0.626	0.556	0.770	1.040	0.887	1.206	2.406	6.093
Random Surfer	10^5	0.000	0.479	0.571	1.305	0.900	0.937	0.675	2.800	4.133
Power Walk		0.000	0.615	0.643	1.317	0.740	0.861	0.679	2.929	4.155
Random Surfer	10^6	0.183	0.535	0.320	0.675	0.964	1.592	0.596	2.825	3.728
Power Walk		0.000	0.521	0.263	0.681	0.964	1.633	0.568	2.747	3.781
Random Surfer	10^7	0.000	0.434	0.320	0.607	0.263	1.780	1.518	1.852	9.822
Power Walk		0.254	0.504	0.643	0.730	0.320	1.725	1.357	1.889	9.826

Table 4: Mean and standard deviation required iterations when using the Random Surfer and Power Walk transition probabilities for seven different graph sizes (10^1 to 10^7) and varying α. The statistics are computed over 30 random trials.

Method	Proportion of random negations										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Mean iterations before convergence											
Power Walk	78.900	78.833	77.133	73.367	53.767	57.867	54.667	41.867	29.933	27.233	23.700
Standard deviation iterations before convergence											
Power Walk	8.142	10.299	12.320	16.548	26.822	22.973	23.908	20.893	8.614	4.953	0.596

Table 5: Mean and standard deviation of the number of iterations required before convergence of the power method, with respect to the proportion of negative edges in the graph. The statistics are computed over 30 random trials.

social networks. In *Link Mining: Models, Algorithms, and Applications*, pages 337–357. Springer, 2010.

R. Lempel and S. Moran. Salsa: the stochastic approach for link-structure analysis. *ACM Transactions on Information Systems (TOIS)*, 19(2):131–160, 2001.

F. Li, Y. Tang, M. Huang, and X. Zhu. Answering opinion questions with random walks on graphs. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 737–745. Association for Computational Linguistics, 2009.

M. Miller, C. Sathi, D. Wiesenthal, J. Leskovec, and C. Potts. Sentiment flow through hyperlink networks. In *ICWSM*, 2011.

A. Y. Ng, A. X. Zheng, and M. I. Jordan. Link analysis, eigenvectors and stability. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 903–910. Citeseer, 2001a.

A. Y. Ng, A. X. Zheng, and M. I. Jordan. Stable algorithms for link analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 258–266. ACM, 2001b.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

L. A. Park and K. Ramamohanarao. Mining web multi-resolution community-based popularity for information retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 545–554. ACM, 2007.

L. A. Park and K. Ramamohanarao. Multiresolution web link analysis using generalized link relations. *Knowledge and Data Engineering, IEEE Transactions on*, 23(11):1691–1703, 2011.

C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li. User-level sentiment analysis incorporating social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1397–1405. ACM, 2011.

S. Teufel, A. Siddharthan, and D. Tidhar. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110. Association for Computational Linguistics, 2006.

D. T. Wijaya and S. Bressan. A random walk on the red carpet: rating movies with user reviews and pagerank. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 951–960. ACM, 2008.

Q. Wu, S. Tan, H. Zhai, G. Zhang, M. Duan, and X. Cheng. Sentirank: Cross-domain graph ranking for sentiment classification. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 01*, pages 309–314. IEEE Computer Society, 2009.