

# RANKING NATIONAL FOOTBALL LEAGUE TEAMS USING GOOGLE'S PAGERANK

ANJELA Y. GOVAN\* AND CARL D. MEYER†

**Abstract.** The search engine Google uses the PageRanks of webpages to determine the order in which they are displayed as the result of a web search. In this work we expand Google's idea of webpage ranking to ranking National Football League teams. We think of the teams as webpages and use the statistics of a football season to create the connections (links) between the teams. The objective is to be able to rank NFL teams by their relative strengths.

**Key Words:** PageRank, stochastic, irreducible, left eigenvector, dominant eigenvalue.

**AMS Subject Classification:** 15A03, 15A51, 62F07.

**1. Preface.** The search engine Google uses an innovative technique of preranking as the main feature for webpage searching. Google's webpage ranking (determining the PageRank of each webpage) is query-independent [7]. This means computation of each webpage's PageRank is done off-line and not during the search query initiated by the user. Google simply pulls the query relevant web sites and displays them according to their predetermined rank.

In order to calculate the PageRanks Google uses a matrix derived from the graph representation of the internet. The goal of this article is to use Google's webpage ranking method to create an algorithm for ranking National Football League (NFL) teams. First we create the NFL graph. Then we construct and modify the matrix corresponding to the NFL graph. Finally, we compute the vector containing the ranks of the teams based on their season's performance. From this ranking we predict the following week's winners.

**2. Overview of PageRank.** Let us start with a basic overview of the PageRank algorithm. We introduce and develop the definition of PageRank via a simple example, a small web with five webpages.

**2.1. Directed internet graph.** First, we construct a directed graph to represent the structure of the internet. The webpages are nodes of the graph and the links between the webpages are the directed edges. In the graph the number of links from a node is called the *out-degree* of that node. Consider the Little web with only five webpages in Figure 2.1.1. Webpage 1 has a link to webpage 3, a link to 4, and a link to 5, so the out-degree of webpage 1 is three.

**2.2. Hyperlink matrix associated with the internet graph.** Now we construct a matrix to represent the hyperlink structure of the network. Let us call this matrix  $\mathbf{H}$ . In the case of the Little web, since there are five webpages, matrix  $\mathbf{H}$  has five rows and five columns. The entry in the  $i$ th row and  $j$ th column represents the link from  $i$ th webpage to the  $j$ th webpage. If there is a link from webpage  $i$  to  $j$  then the corresponding entry is positive and if there is no link the entry is zero. The value of the positive entry depends on the out-degree of the  $i$ th webpage, more precisely it is 1 divided by the out-degree of the webpage  $i$ .

---

\*Center for Research in Scientific Computing, North Carolina State University, Raleigh, NC 27695-8205 (aygovan@math.ncsu.edu)

†Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205 (meyer@math.ncsu.edu)

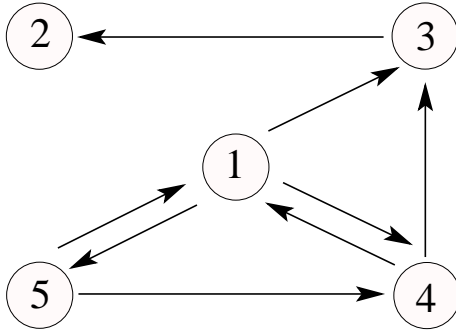


FIG. 2.1.1. *Little web.*

For the Little web the corresponding matrix  $\mathbf{H}$  is

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{pmatrix}$$

**2.3. The rank of a webpage.** Our goal now is to describe how Google determines the importance of a webpage. Brin and Page, the inventors of PageRank, originally defined the rank of a webpage  $P$  as [3]

$$r(P) = \sum_{Q \in B_P} \frac{r(Q)}{|Q|} \quad (2.3.1)$$

where  $r(P)$  represents the rank of a webpage  $P$ ,  $B_P$  is the set of all the webpages pointing to  $P$  and  $|Q|$  is the out-degree of a webpage  $Q$ . The definition is recursive. If webpages  $P$  and  $Q$  have links to each other then the rank of  $P$  depends on the rank of  $Q$  and vice versa. How high the rank of  $P$  is depends on the rank values of the webpages pointing to  $P$  and their out-degree. Simply put “PageRank’s thesis is that a webpage is important if it is pointed to by other important webpages” [7]. To resolve the recursiveness of 2.3.1 we define an iterative procedure for the rank of  $P$  as [7]

$$r_{k+1}(P) = \sum_{Q \in B_P} \frac{r_k(Q)}{|Q|} \quad (2.3.2)$$

where  $r_{k+1}(P)$  is the rank of the webpage  $P$  after  $k + 1$  iterations. We complete the iterative procedure by setting the initial rank of  $P$ ,  $r_0(P)$ , to  $1/n$ . This simple initial ranking is equivalent to giving all webpages the same amount of importance. Since there are five webpages on the Little web each receives an initial rank of  $1/5$ . Let us do a small demonstration of the procedure at work. For the ease of reference call the webpages on the Little web  $P_1$ ,  $P_2$ ,  $P_3$ ,  $P_4$ , and  $P_5$ . The ranks of the webpages can then be arranged in a vector

$$z_0 = ( r_0(P_1) \quad r_0(P_2) \quad r_0(P_3) \quad r_0(P_4) \quad r_0(P_5) ) = ( 1/5 \quad 1/5 \quad 1/5 \quad 1/5 \quad 1/5 )$$

where the first entry in the vector is the initial rank of  $P_1$ , the second is the initial rank of  $P_2$ , etc. Using the initial ranking we perform one iteration of (2.3.2) to recalculate the ranks of the webpages.

$$\begin{aligned} r_1(P_1) &= \frac{r_0(P_4)}{|P_4|} + \frac{r_0(P_5)}{|P_5|} = \frac{1/5}{2} + \frac{1/5}{2} = \frac{1}{5} \\ r_1(P_2) &= \frac{r_0(P_3)}{|P_3|} = \frac{1}{5} \\ r_1(P_3) &= \frac{r_0(P_1)}{|P_1|} + \frac{r_0(P_4)}{|P_4|} = \frac{1/5}{3} + \frac{1/5}{2} = \frac{1}{6} \\ r_1(P_4) &= \frac{r_0(P_1)}{|P_1|} + \frac{r_0(P_5)}{|P_5|} = \frac{1/5}{3} + \frac{1/5}{2} = \frac{1}{6} \\ r_1(P_5) &= \frac{r_0(P_1)}{|P_1|} = \frac{1/5}{3} = \frac{1}{15} \end{aligned}$$

After a single iteration new approximate rank vector is

$$z_1 = ( r_1(P_1) \quad r_1(P_2) \quad r_1(P_3) \quad r_1(P_4) \quad r_1(P_5) ) = ( 1/5 \quad 1/5 \quad 1/6 \quad 1/6 \quad 1/15 )$$

This is not the final ranking of the webpages in our example. The final rank vector (if it exists) is obtained by repeated iterations of (2.3.2). Let us develop more tools for finding the final rank vector and clear the question on whether this final rank vector exists and what it should look like.

**2.4. Determine the rank via matrix multiplication.** Note that the above computation of the rank vector  $z_1$  can be expressed as a simple vector-matrix multiplication

$$z_1 = z_0 \mathbf{H}$$

where  $\mathbf{H}$  is the hyperlink matrix associated with the Little web. Therefore we can rewrite the iterative procedure (2.3.2) as

$$z_{k+1} = z_k \mathbf{H} \quad k = 0, 1, 2, \dots$$

where the  $i$ th entry of vector  $z_{k+1}$  is the rank of the  $i$ th webpage after  $k + 1$  iterations. The consecutive iterations of 2.3.2 can be expressed as the multiplication of the initial ranking vector  $z_0$  by powers of the hyperlink matrix  $\mathbf{H}$ .

$$z_k = z_0 \mathbf{H}^k \quad k = 1, 2, 3, \dots$$

We continue multiplying the ranking vector by increasing powers of  $\mathbf{H}$  until (we hope) the entries in the ranking vector stop changing and we have the final ranking vector. Mathematically this is stated as

$$\lim_{k \rightarrow \infty} z_0 \mathbf{H}^k = \boldsymbol{\pi} \tag{2.4.1}$$

and is known as power method [8, pp. 532-534]. We need this limit (2.4.1) to exist and be unique and the resulting vector,  $\boldsymbol{\pi}$ , to be positive. All of these properties depend on matrix  $\mathbf{H}$ . As is,  $\mathbf{H}$  does not guarantee existence or uniqueness of  $\boldsymbol{\pi}$ .

Recall that we rewrote the iterative procedure for the rank of a webpage  $P$ ,  $r(P)$ , as the vector-matrix multiplication using the matrix  $\mathbf{H}$ . Since  $\mathbf{H}$  does not guarantee the existence or uniqueness of the limit (2.4.1) then the expression  $r(P)$  given in the initial definition 2.3.1 is ambiguous and we have to resolve this ambiguity. Fortunately a repair to the iterative procedure of the rank definition is equivalent to an adjustment of the matrix  $\mathbf{H}$ . First we make  $\mathbf{H}$  stochastic, we call the resulting matrix  $\mathbf{S}$ . Then we make alter  $\mathbf{S}$  so that the resulting Google matrix  $\mathbf{G}$  is irreducible. We do this in two steps. First we make  $\mathbf{H}$  stochastic, we call this new matrix  $\mathbf{S}$ . Then we alter  $\mathbf{S}$  such that the resulting Google matrix  $\mathbf{G}$  is irreducible. An added bonus of the adjustments we introduce in the following sections is that  $\boldsymbol{\pi}$  turns out to be a left eigenvector ( $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T \mathbf{G}$ ) corresponding to the dominant eigenvalue ( $\lambda = 1$ ) of matrix  $\mathbf{G}$ . Therefore the final ranking vector, PageRank vector, will not depend on the initial ranking vector  $z_0$ . The approach is to change the matrix  $\mathbf{H}$  so to make it *stochastic, irreducible* and ensure that  $\lambda = 1$  is only eigenvalue with the absolute value of one. We do this in two steps.

**2.5. Adjusting  $\mathbf{H}$  to make it stochastic.** A matrix is *stochastic* if it has nonnegative entries and the entries in each row sum to 1. The entries in  $\mathbf{H}$  are nonnegative and the number of nonnegative entries in each row is the out-degree of the corresponding webpage. This means that if you sum up the nonnegative entries they add to 1. But there is a catch. There are webpages on the internet that do not have links to any other webpage, they are called *dangling nodes*. Webpage 2 is a dangling node in the Little web and it is responsible for the row of all zeros in  $\mathbf{H}$ . The presence of the zero row violates the stochastic definition. The first step is to fix the zero row. A simple adjustment is to replace the zeros with  $1/5$  which makes the new matrix  $\mathbf{S}$  stochastic.

$$\mathbf{S} = \begin{pmatrix} 0 & 0 & 1/3 & 1/3 & 1/3 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 0 & 1 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{pmatrix}$$

This adjustment implies that we artificially created links from the dangling node to every other webpage on the internet, see Figure 2.5.1.

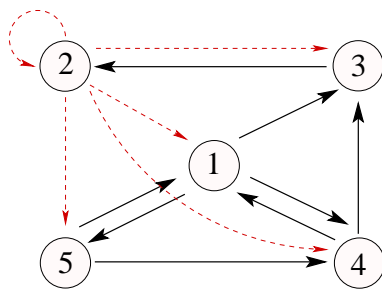


FIG. 2.5.1. *Little web with added links to make sure the corresponding matrix  $\mathbf{S}$  is stochastic.*

We can justify this with the following argument. Suppose you are surfing the internet by *randomly* clicking on the links in the webpages. Then each entry  $(i, j)$  in  $\mathbf{H}$  represents the

probability of you moving from webpage  $i$  to webpage  $j$ . If you get to the dangling node you have no option of clicking on a link to move on, however you can use the URL window to type in a random internet address (we ignore the use of the Back button) and “teleport” to another website [3].

**2.6. Adjusting  $\mathbf{S}$  to make it irreducible.** We need irreducibility of the matrix  $\mathbf{S}$  to insure the convergence of the iterative procedure, i.e. the existence of the final rank vector. A matrix is *irreducible* if the graph it represents is *strongly connected*, this means that for every pair of nodes in the graph there is a collection of edges which one can traverse from one node to the other. In reality the internet does not form a strongly connected graph. Our initial example Little web, Figure 2.1.1, is not strongly connected, there is no collection of edges which can take us from webpage 3 to webpage 5 for instance. Stochastic matrix  $\mathbf{S}$  corresponds to a new graph in which we added links from webpage 2 to every other webpage, this graph is strongly connected, Figure 2.5.1. Hence the corresponding matrix  $\mathbf{S}$  is not only stochastic but also irreducible. However, this is not always the case that the adjustment to  $\mathbf{H}$ , to make it stochastic, also produces the irreducibility of the matrix  $\mathbf{S}$ . To ensure the irreducibility we add artificial links from every webpage to every other webpage by adding a matrix  $\mathbf{E}$  of all  $1/n$  to  $\mathbf{S}$ . In the case of our example entries of matrix  $\mathbf{E}$  have value  $1/5$ .

$$\mathbf{S} + \mathbf{E} = \begin{pmatrix} 1/5 & 1/5 & 8/15 & 8/15 & 8/15 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 1/5 & 6/5 & 1/5 & 1/5 & 1/5 \\ 7/10 & 1/5 & 7/10 & 1/5 & 1/5 \\ 7/10 & 1/5 & 1/5 & 7/10 & 1/5 \end{pmatrix}$$

The implication again is that you have a choice of typing in a webpage address and “teleporting” from any webpage to any other webpage at random. Matrix  $\mathbf{E}$  can be written as  $\mathbf{E} = \mathbf{e}v^T$ , where  $\mathbf{e}$  is a column vector of all 1’s. In the above example we chose  $v = (1/5 \ 1/5 \ 1/5 \ 1/5 \ 1/5)$ . Choice of  $v$ , or “personalization” of  $v$ , can have a sizable effect on the final PageRank vector [3, 5, 7, 14]. Here we define the *personalization vector* to be a vector with all entries being positive and summing up to 1.

All entries of the matrix  $\mathbf{S} + \mathbf{E}$  are positive and this matrix is irreducible, but the row sums do not equal to 1 (i.e. the result of this summation is not a stochastic matrix). This can be easily fixed by using the following mathematical trick. Choose a number  $\alpha$  between zero and one and the new matrix

$$\mathbf{G} = \alpha\mathbf{S} + (1 - \alpha)\mathbf{E}.$$

is stochastic and irreducible. Note that  $\mathbf{G}$  is a combination of stochastic matrices ( $\mathbf{E}$  is stochastic). The positivity of  $\mathbf{G}$  guarantees us that  $\lambda = 1$  is the only eigenvalue of  $\mathbf{G}$  with absolute value of one [8].

**2.7. Google matrix  $\mathbf{G}$ .** With our new matrix  $\mathbf{G}$  the rewritten limit definition (2.4.1) makes sense (the limit exists and it is unique according to Perron-Frobenius Theorem [8, p. 667]):

$$\lim_{k \rightarrow \infty} z_0 \mathbf{G}^k = \boldsymbol{\pi}. \tag{2.7.1}$$

where the  $i$ th entry of vector  $\boldsymbol{\pi}$  is the PageRank of webpage  $i$ .

The number  $\alpha$  has a nice interpretation as the significance of the hyperlink structure of the internet (represented by the matrix  $\mathbf{S}$ ) in the ranking process. For example, it is reported that Google originally used  $\alpha = 0.85$  [2], and therefore the actual web structure imbedded in the matrix  $\mathbf{S}$  accounts for 85% when computing the PageRank of the webpages. For the Little web

$$\mathbf{G} = 0.85\mathbf{S} + 0.15\mathbf{E} = \begin{pmatrix} 3/100 & 3/100 & 47/150 & 47/150 & 47/150 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 3/100 & 22/25 & 3/100 & 3/100 & 3/100 \\ 91/200 & 3/100 & 91/200 & 3/100 & 3/100 \\ 91/200 & 3/100 & 3/100 & 91/200 & 3/100 \end{pmatrix}.$$

Consequently the PageRank vector, computed using Maple, of the Little web is

$$\begin{aligned} \boldsymbol{\pi} &= \left( \frac{3898800}{18552421} \quad \frac{4722161}{18552421} \quad \frac{3956260}{18552421} \quad \frac{3511200}{18552421} \quad \frac{2464000}{18552421} \right) \\ &\approx ( 0.210 \quad 0.255 \quad 0.213 \quad 0.189 \quad 0.133 ). \end{aligned}$$

where PageRank of webpage 1 is 0.210, PageRank of webpage 2 is 0.255, etc. The conclusion of our example is the sorted list of the webpages from most important to least important using the PageRank vector,  $\boldsymbol{\pi}$ :

$$P_2, P_3, P_1, P_4, P_5 .$$

It might be a bit surprising that the dangling node  $P_2$  ends up with the highest rank. However, in the example  $P_3$  has a high rank and since it directs 100% of its PageRank  $P_2$  we would expect that the PageRank of webpage  $P_2$  is also high. The number of webpages in our example might also be one of the possible reasons that  $P_2$  has accumulated such a high rank.

**2.8. Computing PageRank vector.** Due to the small size of our example the computation of the PageRank vector is easy. We use Maple eigenvector routine (Eigenvector(A) computes all the eigenvalues and eigenvectors of a matrix A) to compute  $\boldsymbol{\pi}$ . However, the vast size of the internet (more than 8 billion webpages [7]) precludes such an approach (we only need the left eigenvector associated with the largest eigenvalue,  $\lambda = 1$ ). The founder of Matlab, Clive Moler, referred to PageRank as “The world’s largest matrix computation” [9]. Google has been mentioned to use variants of the power method [8, pp. 533-534] to compute the PageRank vector  $\boldsymbol{\pi}$  [12]. There are other techniques developed for computing PageRank besides power method such as Gauss-Siedel, Jacobi, restarted GMRES, BICGSTAB, etc, see [1, 4, 13].

**3. Constructing the “NFL graph”.** The following discussion presupposes the reader’s basic knowledge of the National Football League rules [16]. We can now tackle the problem of ranking the NFL teams. We start with creating an internet equivalent of an “NFL graph” and proceed to obtain the corresponding Google matrix  $\mathbf{G}$ .

**3.1. NFL weighted graph.** In order to accomplish the team ranking we construct an “NFL graph” in the following fashion. Each team in the league is represented by a node in the graph. Each time two teams play they create a weighted directed edge between each other. The direction of the edge is from the loser to the winner and the weight of the link is equal to the positive difference in scores of the game played. In the PageRank model, a link from webpage  $i$  to webpage  $j$  causes

webpage  $i$  to give some of its own PageRank to webpage  $j$  (via 2.3.1). This is often interpreted as webpage  $i$  “voting for” webpage  $j$  in the big internet election [15]. By analogy, when Chicago Bears loose to Pittsburg Stealers by 12 points, we consider this to be Chicago casting 12 “votes” in favor of Pittsburg in a big NFL election. For example during 2005 season the Pittsburg Stealers played the Chicago Bears and won 21-9. This game results in a directed edge from the Bears(Chi) to the Stealers(Pit) with weight 12 as shown on Figure 3.1.1. In the case the team  $i$  loses more then once to the team  $j$  during one season the weight on the edge from node  $i$  to node  $j$  is the sum of all the positive score differences of the games team  $i$  lost to team  $j$ .

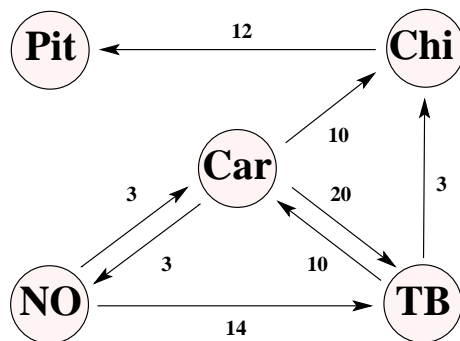


FIG. 3.1.1. Little NFL graph based on 2005 regular season. Car - Carolina Panthers; Chi - Chicago Bears; NO - New Orleans Saints; Pit - Pittsburg Stealers; TB - Tampa Bay Buccaneers.

**3.2. NFL graph matrix.** Since the weights on the edges of the NFL graph depend not only on the NFL graph structure (wins) but also on the score differences the resulting ranking vector will not be the PageRank vector, but will be similar to it. The weight on the edge  $(ij)$  corresponds to the number of “votes” team  $i$  casts in favor of team  $j$ . There are only sixteen games played by each team during the regular season. This makes for a very small set of statistics with which to determine the overall ranking. The more games teams play the more realistic the resulting ranking vector. The challenge is to modify Google’s algorithm to make up for the small size of the NFL graph.

We define the  $(i, j)$ th entry,  $h_{ij}$  in the matrix  $\mathbf{H}$  as

$$h_{ij} = \frac{w_{ij}}{\sum_{j=1}^n w_{ij}}$$

where  $w_{ij}$  is the weight on the edge from team  $i$  to team  $j$ .

For the Little NFL graph the matrix  $\mathbf{H}$  is

$$\mathbf{H} = \begin{matrix} & \begin{matrix} \text{Car} & \text{Pit} & \text{Chi} & \text{TB} & \text{NO} \end{matrix} \\ \begin{matrix} \text{Car} \\ \text{Pit} \\ \text{Chi} \\ \text{TB} \\ \text{NO} \end{matrix} & \begin{pmatrix} 0 & 0 & 10/33 & 20/33 & 3/33 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 10/13 & 0 & 3/13 & 0 & 0 \\ 3/17 & 0 & 0 & 14/17 & 0 \end{pmatrix} \end{matrix}$$

There is also a possibility of having a zero row in  $\mathbf{H}$ . This corresponds to an undefeated season, as evident in our small example. In case of an undefeated season we adjust the matrix  $\mathbf{H}$  by converting the zero row into a row with entries equal to  $1/n$ . This is the equivalent of saying that on any given Sunday the best team might lose to any other team in NFL with probability  $1/n$ . Setting zeroes in the zero row to  $1/n$  is not an ideal solution since  $1/n$  is a significant number (probability) given that  $n \leq 32$ . This adjustment could penalize the undefeated team. An alternative approach could be to adjust the zero row by putting a 1 on the diagonal. This is interpreted as saying that the undefeated team votes only for itself in the “election.” Improving the adjustment of the matrix  $\mathbf{H}$ , so to avoid penalizing the undefeated teams, is one of the topics of our future research. Back to our example, the stochastic matrix  $\mathbf{S}$  for the Little NFL graph:

$$\mathbf{S} = \begin{pmatrix} 0 & 0 & 10/33 & 20/33 & 3/33 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 0 & 1 & 0 & 0 & 0 \\ 10/13 & 0 & 3/13 & 0 & 0 \\ 3/17 & 0 & 0 & 14/17 & 0 \end{pmatrix}$$

Like the internet graph the NFL graph may not be strongly connected. The final adjustment is

$$\mathbf{G} = \alpha\mathbf{S} + (1 - \alpha)\mathbf{e}\mathbf{v}^T$$

where  $\mathbf{v}$  is a personalization vector and  $\mathbf{e}$  is a column vector of all 1’s. This time both  $\mathbf{v}$  and  $\alpha$  are computed experimentally based on the data from several NFL seasons. Future work will include a technique for optimizing the selection of  $\alpha$  as well as expanding the algorithm to add more statistical data to  $\mathbf{G}$ . To continue with our example of the Little NFL graph let  $\mathbf{v}^T = ( 1/5 \ 1/5 \ 1/5 \ 1/5 \ 1/5 )$  and  $\alpha = 0.85$  then the rank vector of the Little NFL graph is

$$\begin{aligned} \boldsymbol{\pi} &= \left( \frac{1056000}{4234481} \quad \frac{2835863}{12703443} \quad \frac{2320780}{12703443} \quad \frac{3270800}{12703443} \quad \frac{1108000}{12703443} \right) \\ &\approx ( 0.249 \quad 0.223 \quad 0.183 \quad 0.257 \quad 0.087 ) \end{aligned}$$

and the list of ranked of teams from number one to number five is

Tampa Bay, Carolina, Pittsburg, Chicago, New Orleans

If we change our vector  $\mathbf{v}$  to reflect a hypothetical statistic (e.g. defense, offense, injuries, etc.) of each team in the Little web  $\mathbf{v}^T = ( 8/30 \ 10/30 \ 6/30 \ 2/30 \ 4/30 )$ <sup>1</sup> then the rank vector is

$$\boldsymbol{\pi} = \left( \frac{37027881}{148206835} \quad \frac{22033561}{88924101} \quad \frac{81421474}{444620505} \quad \frac{3021226}{12703443} \quad \frac{36204673}{444620505} \right)$$

---

<sup>1</sup>This vector is not based on any actual data but on the whims of the authors. Its sole purpose is to illustrate the effect change in the vector  $\mathbf{v}$  can have on the final ranking of the football teams.

$$\approx ( 0.2498 \quad 0.2478 \quad 0.1831 \quad 0.2378 \quad 0.0814 )$$

and the list of ranked of teams from number one to number five is

Carolina, Pittsburg, Tampa Bay, Chicago, New Orleans

We update matrix  $\mathbf{G}$  every week and predict the winners of the games in the following week based on the computed team ranks. The accuracy of the predictions increases as the season progresses.

Amy Oliver’s Masters thesis [10] provides a summary of the experimental runs for the proposed algorithm against human predictions <sup>2</sup> and two other ranking methods one by Keener and one by Redmond [6, 11] for the 2005-2006 NFL season. Each “method” (Google, Keener, Redmond, and humans) made predictions for the following week’s outcomes. Our PageRank based method outperformed both humans as well as algorithms by Keener and by Redmond by predicting more winners correctly.

**4. Conclusion.** We have encountered a number of questions and issues while working on the ranking algorithm for the NFL teams. In our future work we want to include the following:

- Develop a mathematical model to predict expected point spreads in a given contest. This might be based on determining an aspatial metric using the PageRank model to measure how “far” team  $i$  is from team  $j$ .
- Try out and compare alternative methods for handling an undefeated season (adjusting the zero row in the NFL graph matrix  $\mathbf{H}$ ).
- Add a modification that prevents a team from artificially improving its ranking by running up the scores against weak teams.
- In National Football League each team plays less then sixteen opponents out of total thirty two. The respective ranks of the teams that never play each other during the season are primarily based on the teams statistical records. Therefore we wish to use most of the seasonal statistics, associated with each team, in our computation of each team’s rank. A solution to this problem will be to develop a methodology for the statistical data processing. The results of this process should be relevant to the application and easy to interpret. The other part of the solution is a technique for incorporating the processed statistics into Google matrix  $\mathbf{G}$ .

#### REFERENCES

- [1] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and Van der Vorst. *Templates for the solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 2nd edition, 1994.
- [2] Sergey Brin and Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine*. Computer Networks and ISDN Systems, 33:107-17, 1998.
- [3] Sergey Brin, Lawrence Page, P. Motwami, and Terry Winograd. *The PageRank Citation Ranking: Brining Order to the Web*. Technical Report 1999-0120, Computer Science Department, Stanford University, 1999
- [4] James Gleich, Leonis Zhukov, and Pavel Berkhin. *Fast parallel PageRank: A linear system approach*. In the *Fourteenth International World Wide Conference* ACM Press, New York, 2005.

---

<sup>2</sup>The group of humans making weekly predictions consisted of some NCSU professors, graduate students, their families and friends. Some individuals in the group used their own intuition of the NFL to make predictions. Others relied on the predictions made by sports experts and Las Vegas oddsmakers.

- [5] Taher H. Haveliwala, Sepandar D. Kamvar, and Glen Jeh. *An analytical comparison of approaches to personalizing PageRank*. Technical Report, Computer Science Department, Stanford University, 2003.
- [6] James P. Keener *A The Perron-Frobenius Theorem and the Ranking of Football Teams*. SIAM Review, vol.35, No.1, pp. 80-9, 1993.
- [7] Amy N. Langville, Carl D. Meyer *Google's PageRank and Beyond The Science of Search Engine Rankings*. 2006.
- [8] Carl D. Meyer *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2005.
- [9] Cleve Moler *The world's largest matrix computation*. Matlab News and Notes, pp.12-13, 2002.
- [10] Amy Oliver *Ranking NFL Teams*. Masters thesis, North Carolina State University, 2004.
- [11] C. Redmond *A Natural Generalization of the Win-Loss Rating System, 76*. Mathematical Magazine, 2003.
- [12] Sara Robinson *The Ongoing Search for Efficient Web Search Algorithms*. SIAM News, vol.37, No.9, 2004.
- [13] William J. Stewart *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, 2005.
- [14] Kristen Thorson *Modeling the Web and the computation of PageRank*. Undergraduate thesis, Hollins University, 2004.
- [15] <http://www.google.com/technology/>
- [16] <http://www.nfl.com>