The PageRank Computation in Google, Randomized Algorithms and Consensus of Multi-Agent Systems

Roberto Tempo

IEIIT-CNR
Politecnico di Torino
tempo@polito.it
The objective of this talk is to discuss three apparently unrelated topics:

1. Search engines (PageRank computation in Google)
2. Randomized algorithms (Las Vegas type)
3. Consensus of multi-agent systems
This talk

- The objective of this talk is to discuss three apparently unrelated topics

- Search engines (PageRank computation in Google)
- Randomized algorithms (Las Vegas type)
- Consensus of multi-agent systems

- The math behind: theory of positive matrices
The objective of this talk is to discuss three apparently unrelated topics:

- Search engines (PageRank computation in Google)
- Randomized algorithms (Las Vegas type)
- Consensus of multi-agent systems

- Multiple web page updates, web page aggregation, robustness for fragile links, web semantics
The PageRank Problem in Google
PageRank is Google’s view of the importance of this page (7/10)
PageRank for UAE University

“PageRank is Google’s view of the importance of this page (7/10)”

PageRank is a numerical value in the interval [0,1] which indicates the importance of the page you are visiting.
Random Surfer Model

- Web surfer moves along randomly following the hyperlink structure.
- When arriving at a page with several outgoing links, one is chosen at random, then the random surfer moves to a new page, and so on...
Random Surfer Model

- Web representation with incoming and outgoing links
Random Surfer Model
Random Surfer Model

- Pick an outgoing link at random
Random Surfer Model

- Arriving at a new web page
Random Surfer Model

- Pick another outgoing link at random
Random Surfer Model
Random Surfer Model
Random Surfer Model

- If a page is “important” then it is visited more often...
- The time the random surfer spends on a page is a measure of the importance of the page
- If important pages point to your page, then your page becomes important (because it is often visited)
- For facilitating the web search we need to rank the pages
- We assign a numerical value to each page
Graph Representation

- Directed graph with nodes (pages) and links representing the web
- Graph is not necessarily strongly connected
- Graph is constructed using crawlers and spiders which move continuously along the web
For each node we count the number of outgoing links and normalize them to 1

Hyperlink matrix is a nonnegative (column) substochastic matrix
Hyperlink Matrix

\[ A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/2 & 0 \\
1/3 & 0 & 0 & 1/2 & 0 \\
1/3 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
\end{bmatrix} \]
PageRank: Bringing Order to the Web\textsuperscript{[1,2]}

- Need to rank pages in order of importance
- The PageRank $x^*$ is defined as
  \[ x^* = A x^* \quad \text{where} \quad x^* \in [0,1]^n \quad \text{and} \quad \sum_i x_i^* = 1 \]
- $x^*$ is a nonnegative unit eigenvector corresponding to the eigenvalue 1 for the hyperlink matrix $A$
- The question is when $x^*$ exists and it is unique

\textsuperscript{[1]} S. Brin, L. Page (1998)
Issue of Dangling Nodes

- **First issue:** We have *dangling nodes*
- Random surfer gets “stuck” when visiting a pdf file
- In this case the “back button” of the browser is used
- Mathematically, the hyperlink matrix is nonnegative and (column) substochastic
- **Easy fix:** Add artificial links to make the matrix stochastic
Page 5 is a Dangling Node

\[ A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/2 & 0 \\
1/3 & 0 & 0 & 1/2 & 0 \\
1/3 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
\end{bmatrix} \]
We add an outgoing link to page 5

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/2 & 0 \\
1/3 & 0 & 0 & 1/2 & 1 \\
1/3 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0
\end{bmatrix}
\]
But in General the Fix is not so Easy…

- Page 5 has two incoming links
But in General the Fix is not so Easy…

- We add an outgoing link from 5 to 3…

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/3 & 0 \\
1/3 & 0 & 0 & 1/3 & 1 \\
1/3 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1/3 & 0
\end{bmatrix}
\]
But in General the Fix is not so Easy…

… or we add an outgoing link from 5 to 4?

\[ A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/3 & 0 \\
1/3 & 0 & 0 & 1/3 & 0 \\
1/3 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1/3 & 0 
\end{bmatrix} \]
A solution may be to break page 5 into two pages 5a and 5b. This artificially changes the number of pages (not only the number of links) and the topology of the network.
Modified Hyperlink Matrix

\[ A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 1/3 & 0 & 0 & 0 \\
1/3 & 0 & 0 & 0 & 1/3 & 1 & 0 \\
1/3 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/3 & 0 & 0 & 0
\end{bmatrix} \]
Assumption: No Dangling Nodes

- We assume that there are no dangling nodes.
- This implies that $A$ is a nonnegative stochastic matrix (instead of substochastic) having at least one eigenvalue equal to one.
- **Second issue:** This eigenvalue is not necessarily unique.
Teleportation Matrix

- The random surfer may get bored after a while, and decides to “jump” to another page not directly connected to that currently visited.
Recall the Random Surfer Model

- Web representation with incoming and outgoing links
Recall the Random Surfer Model
Recall the Random Surfer Model
Recall the Random Surfer Model
We are “teleported” to a web page located far away.
Random Surfer Model Again

- Pick another outgoing link at random
Random Surfer Model Again

- Pick another outgoing link at random
Teleportation Model Again

- We are teleported to another web page located far away
Convex Combination of Matrices

- Teleported model is represented as a convex combination of matrices
- Instead of $A$ we consider a matrix $M$ defined as
  \[ M = (1 - m)A + \frac{m}{n}S \quad m \in (0,1) \]
  where $S$ is a matrix with all entries equal to 1 and $n$ is the number of pages
- The value $m = 0.15$ is proposed and used at Google\(^\text{[1]}\)

\(^\text{[1]}\) S. Brin, L. Page (1998)
Matrix $M$

- $M$ is positive stochastic (convex combination of two stochastic matrices and $m \in (0,1)$)
Matrix $M$ and Perron Theorem

- The matrix $M$ is primitive ($M^k$ is positive for some $k$)
- $M$ is irreducible and the corresponding graph is strongly connected (every page is connected to every page)
- The eigenvalue 1 is simple and it is the unique eigenvalue of maximum modulus
- The corresponding eigenvector is positive
PageRank Computation
PageRank Computation

- PageRank is computed with the power method
  \[ x(k+1) = M x(k) \]

- Convergence of this recursion is guaranteed by Perron Theorem because \( M \) is a primitive matrix
  \[ x(k) \to x^* \quad \text{for} \quad k \to \infty \]
  provided that \( \sum x_i(0) = 1 \)

- **Remark:** PageRank computation can be interpreted as finding the stationary distribution of a Markov Chain
PageRank Computation with Power Method

\[ A = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix} \]

\[ m = 0.15 \]

\[ M = \begin{bmatrix} 0.038 & 0.037 & 0.037 & 0.321 \\ 0.887 & 0.037 & 0.462 & 0.321 \\ 0.037 & 0.462 & 0.037 & 0.321 \\ 0.037 & 0.462 & 0.462 & 0.037 \end{bmatrix} \]

\[ x^* = \begin{bmatrix} 0.12 & 0.33 & 0.26 & 0.29 \end{bmatrix}^T \]
Convergence Properties

- Asymptotic rate of convergence of power method is exponential and depends on the ratio
  \[ \left| \frac{\lambda_2(M)}{\lambda_1(M)} \right| \]

- We have
  \[ \lambda_1(M) = 1 \quad \lambda_2(M) \leq 1 - m = 0.85 \]
Size of the Web

- The size of $M$ is 8 billion!
- The PageRank computation requires 50-100 iterations
- This takes about a week and it is performed centrally at Google once a month
- More and more computing power is needed…
Columbia River, The Dalles, Oregon
Randomized Decentralized Algorithms
Randomized Algorithm (RA): An algorithm that makes random choices during its execution to produce a result.
Randomized Algorithm (RA): An algorithm that makes random choices during its execution to produce a result.

Example of a “random choice” is a coin toss.

- heads
- or
- tails
Monte Carlo and Las Vegas Randomized Algorithms

- Monte Carlo was invented by Metropolis, Ulam, von Neumann, Fermi, … (Manhattan project)
- Las Vegas first appeared in computer science in the late seventies
- Successful applications of randomized algorithms in several areas, e.g. robotics, computer science, finance, bioinformatics, communication and networking, systems and control, …
Randomized Decentralized Approach\cite{1}

- **Main idea:** Develop a randomized decentralized approach for computing PageRank
- Key difference with a centralized approach (i.e. power method) which involves the entire web
- Randomized algorithm is of Las Vegas type

\cite{1} H. Ishii, R. Tempo (2008)
Basic Communication Protocol

Basic communication protocol: at time $k$ the randomly selected page $i$ initiates the PageRank update as follows:
Basic communication protocol: at time $k$ the randomly selected page $i$ initiates the PageRank update as follows:
Basic communication protocol: at time $k$ the randomly selected page $i$ initiates the PageRank update as follows:

1. by sending the value of page $i$ to the outgoing pages that are linked to $i$
Basic communication protocol: at time $k$ the randomly selected page $i$ initiates the PageRank update as follows:

1. by sending the value of page $i$ to the outgoing pages that are linked to $i$
2. by requesting their values from the incoming pages that are linked to page $i$
Las Vegas Randomized Approach

- The pages taking action are determined via a random process $\theta(k) \in \{1, \ldots, n\}$
- If at time $k$ $\theta(k) = i$ then page $i$ initiates PageRank update
- $\theta(k)$ is assumed to be i.i.d. with uniform probability
  \[ \text{Prob}\{\theta(k)=i\} = \frac{1}{n} \]
We consider the randomized update scheme

\[ x(k+1) = A_{\theta(k)} x(k) \]

where \( A_{\theta(k)} \) are the distributed link matrices (example next)

Consider the time average

\[ y(k) = \frac{1}{(k+1)} \sum_i x(i) \]
Distributed Link Matrices - 1

\[
A = \begin{bmatrix}
0 & 0 & 0 & 1/3 \\
1 & 0 & 1/2 & 1/3 \\
0 & 1/2 & 0 & 1/3 \\
0 & 1/2 & 1/2 & 0
\end{bmatrix}
\]

\[
A_4 = \begin{bmatrix}
1/3 \\
1/3 \\
1/3 \\
0 & 1/2 & 1/2 & 0
\end{bmatrix}
\]
Distributed Link Matrices - 2

\[ A = \begin{bmatrix}
  0 & 0 & 0 & 1/3 \\
  1 & 0 & 1/2 & 1/3 \\
  0 & 1/2 & 0 & 1/3 \\
  0 & 1/2 & 1/2 & 0
\end{bmatrix} \]

\[ A_4 = \begin{bmatrix}
  0 & 0 & 0 & 1/3 \\
  0 & 0 & 1/3 \\
  0 & 0 & 1/3 \\
  0 & 1/2 & 1/2 & 0
\end{bmatrix} \]
Distributed Link Matrices - 3

\[ A = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix} \]

\[ A_4 = \begin{bmatrix} 1 & 0 & 0 & 1/3 \\ 0 & 1/2 & 0 & 1/3 \\ 0 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix} \]
Distributed Link Matrices - 4

\[
A = \begin{bmatrix}
0 & 0 & 0 & 1/3 \\
1 & 0 & 1/2 & 1/3 \\
0 & 1/2 & 0 & 1/3 \\
0 & 1/2 & 1/2 & 0
\end{bmatrix} \quad A_3 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1/2 & 1/2 & 0 \\
0 & 1/2 & 0 & 1/3 \\
0 & 0 & 1/2 & 2/3
\end{bmatrix}
\]

\[
A_4 = \begin{bmatrix}
1 & 0 & 0 & 1/3 \\
0 & 1/2 & 0 & 1/3 \\
0 & 0 & 1/2 & 1/3 \\
0 & 1/2 & 1/2 & 0
\end{bmatrix}
\]
\[ A = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{bmatrix} \]

\[ A_1 = \begin{bmatrix} 0 & 0 & 0 & 1/3 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2/3 \end{bmatrix} \]

\[ A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 1/2 & 0 & 2/3 \end{bmatrix} \]
Average Matrix $\overline{A}$

- The average matrix $\overline{A} = E[A_{\theta(k)}]$
- Since we take uniform distribution in the random process $\theta(k)$ we have
  \[ \overline{A} = \frac{1}{n} \sum_i A_i \]

- Lemma:
  (i) $\overline{A} = \frac{2}{n} A + (1-\frac{2}{n}) I$
  (ii) The matrices $A$ and $\overline{A}$ have the same eigenvector corresponding to the eigenvalue 1
Revised Distributed Update Scheme

- Recall that we need to work with positive stochastic matrices.
- We consider the modified distributed update scheme:

\[ x(k+1) = M_{\theta(k)} x(k) \]

where \( M_{\theta(k)} \) are the modified distributed link matrices computed as:

\[ M_i = (1 - r) A_i + r/n S \quad i = 1, 2, \ldots, n \]

and \( r \in (0,1) \) is a design parameter (defined next).
The average matrix $M = \mathbb{E}[M_{\theta(k)}]$.

Define $r = 2m/(n - mn + 2m)$.

**Lemma:**

(i) $r \in (0,1)$ and $r < m = 0.15$

(ii) $M = r/m M + (1-r/m) I$

(iii) For $M$ the eigenvalue 1 is simple and it is the unique eigenvalue of maximum modulus. The PageRank is the corresponding eigenvector.
Theorem: 

Using the modified distributed update scheme the PageRank is obtained through the time average $y$

$$E[||y(k) - x^*||^2] \to 0 \text{ for } k \to \infty$$

provided that $\sum_i x_i(0) = 1$

Proof: Based on the theory of ergodic matrices

Remark: The algorithm is a LVRA
- The average $y(k)$ can be computed recursively in terms of $y(k-1)$
- Sparsity of the matrix $A_i$ can be preserved because
  $$x(k+1) = M_i x(k) = (1 - r) A_i x(k) + r/n \mathbf{1}$$
  where $\mathbf{1}$ is a vector with all entries equal to one
- Convergence rate is $1/k$
- Stopping criteria to compute approximately PageRank
- Different update schemes based only on outgoing links (not incoming): similar convergence results
Extensions: Simultaneous Updates
Recall the Random Surfer Model

- Web representation with incoming and outgoing links
Random Update of One Page

- Random update of one page
Simultaneous Random Updates of Multiple Pages

- Simultaneous random update of multiple webpages
Convergence Results

- Simultaneous random updates of multiple pages requires defining Bernoulli processes instead of i.i.d. processes
- The math is more involved
- Similar convergence results for multiple updates can be obtained...

Consensus and PageRank Problems
Unmanned Aerial Vehicle (UAV) for fire detection in Sicily

- On-board equipment: various sensors, two cameras (color and infrared), GPS, ...
- DC motor
- Remote piloting and autonomous flight
- Flight endurance of about 40 min
Several UAVs (Agents) Reaching a Target

UAV_1

UAV_2

UAV_3

UAV_4

target
Several UAVs (Agents) Reaching a Target

waypoints obstacles

UAV₁  UAV₂

UAV₃  target

UAV₄
Graph of Agents

- We consider a graph of agents which communicate using a random protocol.
- The value (e.g. position) of agent $i$ at time $k$ is $x_i$.
- Values of agents are updated using a random scheme:
  $$x(k+1) = A_{\theta(k)} \cdot x(k)$$
- Communication pattern (i.e. $A_{\theta(k)}$) is similar to that used for PageRank (with some technical differences).
We say that consensus (i.e., reaching the target) is achieved if for any initial condition $x(0)$ we have

$$|x_i(k) - x_j(k)| \rightarrow 0 \text{ for } k \rightarrow \infty$$

with probability one for all $i, j$.
Lemma: Assume that the graph is strongly connected. Then, the scheme

\[ x(k+1) = A_{\theta(k)} x(k) \]

achieves consensus

\[ |x_i(k) - x_j(k)| \to 0 \quad \text{for} \quad k \to \infty \]

with probability one for all \( i, j \) and for any initial condition \( x(0) \)
## PageRank and Consensus

<table>
<thead>
<tr>
<th>Consensus</th>
<th>PageRank</th>
</tr>
</thead>
<tbody>
<tr>
<td>All agent values become equal</td>
<td>Page values converge to constant</td>
</tr>
<tr>
<td>Graph is strongly connected</td>
<td>Web is not strongly connected</td>
</tr>
<tr>
<td>Convergence w.p.1 for all ( x_i, x_j )</td>
<td>MSE convergence for ( y )</td>
</tr>
<tr>
<td>(</td>
<td>x_i(k) - x_j(k)</td>
</tr>
<tr>
<td>Average is not necessary</td>
<td>Average crucial for convergence</td>
</tr>
<tr>
<td>Matrices ( A_i ) are row stochastic</td>
<td>Matrices ( M_i ) are column stochastic</td>
</tr>
</tbody>
</table>
Various research directions can be explored:

- robustness for fragile, time-varying and broken links
- aggregation and clustering of webpages
- web semantics
Fragile, Time-Varying and Broken Links[1]

Webpage Aggregation and Clustering\cite{1}

original web

aggregated web

\cite{1} H. Ishii, R. Tempo (2009)
Why the result of the web search is different if you type “roberto tempo” or “tempo roberto”?

If you type “Boston Hotel” are you looking for an hotel in the city of Boston or are you looking for an hotel with the name “Boston”? 
References and Software

- “RACT: Randomized Algorithms Control Toolbox”

http://staff.polito.it/roberto.tempo/
Acknowledgment: Research on PageRank is joint work with Hideaki Ishii