Suppose our input data to a map-reduce operation consists of integer values (the keys are not important). The map function takes an integer $i$ and produces the list of pairs $(p, i)$ such that $p$ is a prime divisor of $i$. For example, $\text{map}(12) = [(2, 12), (3, 12)]$. The reduce function is addition. That is, $\text{reduce}(p, [i, i, ..., i]) = (p, i + i + ... + i)$. Compute the output, if the input is the set of integers 15, 21, 24, 30, 49. Then, identify, in the list below, one of the pairs in the output.

- a. $(7, 70)$
- b. $(5, 49)$
- c. $(2, 47)$
- d. $(6, 54)$
Connections between political blogs
Polarization of the network [Adamic-Glance, 2005]

Citation networks and Maps of science
[Börner et al., 2012]

Seven Bridges of Königsberg
Euler, 1735
Return to the starting point by traveling each link of the graph once and only once.
Web as a Graph

- Web as a directed graph:
  - Nodes: Webpages
  - Edges: Hyperlinks


Web as a Directed Graph

- How to organize the Web?
  - First try: Human curated Web directories
    - Yahoo, DMOZ, LookSmart
  - Second try: Web Search
    - Information Retrieval investigates:
      - Find relevant docs in a small and trusted set
        - Newspaper articles, Patents, etc.
    - But: Web is huge, full of untrusted documents, random things, web spam, etc.
Web Search: 2 Challenges

2 challenges of web search:
- (1) Web contains many sources of information
  Who to “trust”?
  - Trick: Trustworthy pages may point to each other!
- (2) What is the “best” answer to query “newspaper”?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Ranking Nodes on the Graph

- All web pages are not equally “important”
  www.joe-schmoe.com vs. www.stanford.edu
- There is large diversity in the web-graph node connectivity.
  Let’s rank the pages by the link structure!

Link Analysis Algorithms

- We will cover the following Link Analysis approaches for computing importances of nodes in a graph:
  - Page Rank
  - Topic-Specific (Personalized) Page Rank
  - Web Spam Detection Algorithms

PageRank:
The “Flow” Formulation
**Links as Votes**

- **Idea: Links as votes**
  - Page is more important if it has more links
    - In-coming links? Out-going links?
- **Think of in-links as votes:**
  - [www.stanford.edu](http://www.stanford.edu) has 23,400 in-links
  - [www.joe-schmoe.com](http://www.joe-schmoe.com) has 1 in-link
- **Are all in-links equal?**
  - Links from important pages count more
    - Recursive question!

**Simple Recursive Formulation**

- Each link’s vote is proportional to the **importance** of its source page
- If page $j$ with importance $r_j$ has $n$ out-links, each link gets $r_j/n$ votes
- Page $j$’s own importance is the sum of the votes on its in-links

$$r_j = \frac{r_i}{3} + \frac{r_k}{4}$$

**Example: PageRank Scores**

- **A “vote” from an important page is worth more**
- **A page is important if it is pointed to by other important pages**
- **Define a “rank” $r_j$ for page $j$**

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

$d_i$ ... out-degree of node $i$
Solving the Flow Equations

- 3 equations, 3 unknowns, no constants
- No unique solution
- All solutions equivalent modulo the scale factor
- Additional constraint forces uniqueness:
  \[ r_y + r_a + r_m = 1 \]
  Solution: \[ r_y = \frac{2}{5}, \ r_a = \frac{2}{5}, \ r_m = \frac{1}{5} \]
- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs
- We need a new formulation!

PageRank: Matrix Formulation

- Stochastic adjacency matrix \( M \)
  - Let page \( i \) has \( d_i \) out-links
  - If \( i \rightarrow j \), then \( M_{ji} = \frac{1}{d_i} \) else \( M_{ji} = 0 \)
  - \( M \) is a column stochastic matrix
    * Columns sum to 1
- Rank vector \( r \): vector with an entry per page
  - \( r_i \) is the importance score of page \( i \)
  - \( \sum_i r_i = 1 \)
- The flow equations can be written
  \[ r = M \cdot r \]

Example

- Remember the flow equation: \[ r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i} \]
- Flow equation in the matrix form
  \[ M \cdot r = r \]
  Suppose page \( i \) links to 3 pages, including \( j \)

\[
\begin{array}{c c c}
\hline
i & j & r_j \\
\hline
1/3 & 1 & r_i \\
\hline
\end{array}
\]

\[ M \cdot r = r \]

Eigenvector Formulation

- The flow equations can be written
  \[ r = M \cdot r \]
  So the rank vector \( r \) is an eigenvector of the stochastic web matrix \( M \)
  - In fact, its first or principal eigenvector, with corresponding eigenvalue \( 1 \)
    * Largest eigenvalue of \( M \) is \( 1 \) since \( M \) is column stochastic (with non-negative entries)
      * We know \( r \) is unit length and each column of \( M \) sums to one, so \( Mr \leq 1 \)
- We can now efficiently solve for \( r \)!
  - The method is called Power iteration

\[ Ax = \lambda x \]
**Example: Flow Equations & M**

\[
\begin{array}{c|c|c|c}
\text{y} & \frac{1}{2} & \frac{1}{2} & 0 \\
\text{a} & \frac{1}{2} & 0 & 1 \\
\text{m} & 0 & \frac{1}{2} & 0 \\
\end{array}
\]

**r = M \cdot r**

\[
\begin{align*}
\text{ry} &= \text{ry}/2 + \text{ra}/2 \\
\text{ra} &= \text{ry}/2 + \text{rm} \\
\text{rm} &= \text{ra}/2
\end{align*}
\]

**Power Iteration Method**

- **Given a web graph with** \( n \) **nodes, where the nodes are pages and edges are hyperlinks**
- **Power iteration**: a simple iterative scheme
  - **Suppose there are** \( N \) **web pages**
  - **Initialize**: \( \text{r}^{(0)} = [1/N, \ldots, 1/N]^T \)
  - **Iterate**: \( \text{r}^{(t+1)} = \text{M} \cdot \text{r}^{(t)} \)
  - **Stop when** \( |\text{r}^{(t+1)} - \text{r}^{(t)}|_1 < \varepsilon \)
  
\[|\text{x}|_1 = \sum_{i=1}^N |\text{x}_i|\]  

**Example:**

\[
\begin{align*}
\text{ry} &= 1/3 \\
\text{ra} &= 3/6 \\
\text{rm} &= 1/3
\end{align*}
\]

**PageRank: How to solve?**

**Power Iteration:**

- **Set** \( \text{r}_j = 1/N \)
- **1**: \( r'_j = \sum_{i \rightarrow j} \text{r}_i / d_i \)
- **2**: \( r' = r' \)
- **Goto 1**

**Example:**

\[
\begin{align*}
\text{ry} &= 1/3 \\
\text{ra} &= 3/6 \\
\text{rm} &= 3/12
\end{align*}
\]

**PageRank: How to solve?**

**Power Iteration:**

- **Set** \( \text{r}_j = 1/N \)
- **1**: \( r'_j = \sum_{i \rightarrow j} \text{r}_i / d_i \)
- **2**: \( r' = r' \)
- **Goto 1**

**Example:**

\[
\begin{align*}
\text{ry} &= 1/3 \\
\text{ra} &= 3/6 \\
\text{rm} &= 1/3
\end{align*}
\]
Random Walk Interpretation

- Imagine a random web surfer:
  - At any time $t$, surfer is on some page $i$
  - At time $t+1$, the surfer follows an out-link from $i$ uniformly at random
  - Ends up on some page $j$ linked from $i$
  - Process repeats indefinitely

- Let:
  - $p(t)$ ... vector whose $i$th coordinate is the prob. that the surfer is at page $i$ at time $t$
  - So, $p(t)$ is a probability distribution over pages

The Stationary Distribution

- Where is the surfer at time $t+1$?
  - Follows a link uniformly at random
  - $p(t+1) = M \cdot p(t)$

- Suppose the random walk reaches a state
  - $p(t+1) = M \cdot p(t) = p(t)$
  - then $p(t)$ is **stationary distribution** of a random walk

- Our original rank vector $r$ satisfies $r = M \cdot r$
  - So, $r$ is a stationary distribution for the random walk

Existence and Uniqueness

- A central result from the theory of random walks (a.k.a. Markov processes):
  - For graphs that satisfy certain conditions, the stationary distribution is unique and eventually will be reached no matter what the initial probability distribution at time $t = 0$