Part 1: The Web as a Graph & Link Analysis in Information Retrieval

Introduction to Link Analysis

Link analysis is a critical component in web search algorithms, significantly impacting the ranking of web pages in response to a query. The underlying idea is derived from citation analysis and bibliometrics, where the influence of academic papers is gauged by analyzing how often a paper is cited by others. Similarly, in web search, hyperlinks are treated as endorsements of authority from one web page to another.

Concepts in Link Analysis

1. **Web Graph and Hyperlinks**: The web can be represented as a directed graph, where each webpage is a node, and hyperlinks between pages act as directed edges. For instance, if page A links to page B, the edge from A to B represents a "vote" or endorsement from A for B.

2. **Anchor Text**: Anchor text is the clickable text in a hyperlink, and it is often used to describe the target page. Search engines use anchor text as a key ranking signal. For example:

   ```
   <a href="http://www.example.com">Example Site</a>
   ```

   In this case, "Example Site" is the anchor text describing `http://www.example.com`. Even if the target page doesn't explicitly mention the words "Example Site," search engines use this anchor text to index the page.

3. **Authority Conferral**: Links between web pages act as endorsements. A page with many in-links (pages linking to it) is perceived as more authoritative. However, not all links confer authority—internal links within a single website often serve navigational purposes rather than endorsements. For example, most corporate websites have internal links to privacy policies or terms of service, which do not imply authority.

Graph Theory and the Web

The web graph consists of millions of nodes (web pages) and edges (hyperlinks). Key concepts in graph theory that apply to link analysis include:

- **Directed Edges**: In a web graph, hyperlinks are directed, meaning they point from one page to another.
- **Connected Components**: Large portions of the web form interconnected subgraphs. It's important to consider that not all pages are connected to all others; some nodes might be isolated or form distinct clusters.

Exercise Example:

Exercise 21.1: **Question**: Is it always possible to follow directed edges (hyperlinks) in the web graph from any node to any other? Why or why not? **Answer**: No, it is not always possible. The web graph is not a fully connected graph, meaning that some nodes (pages) might not be reachable from others due to a lack of hyperlinks between them.

Anchor Text Usage in Information Retrieval

The idea behind using anchor text is to supplement the content of a webpage with external descriptions. Often, a webpage may lack relevant text that matches search queries (e.g., corporate websites that avoid common industry terms for marketing reasons). Anchor text can fill this gap by offering alternative keywords from other web pages linking to it. For instance, if many pages link to a company website with the anchor text "tech company," the search engine will associate that company website with the term, even if it doesn't appear on the company's page.

This leads to some interesting phenomena, such as:

- **Popular Nicknames**: The homepage of IBM, at one point, did not contain the word "computer," yet it would still rank highly for searches including this term due to numerous links with anchor text like "computer company."
- **Derogatory Anchor Text**: Sometimes negative phrases can become linked to a page due to orchestrated campaigns, creating misleading or unintended results in search rankings.

Exercise Example:

Exercise 21.2: **Question**: Find an instance of misleading anchor-text on the Web. **Answer**: A classic example would be the search term "miserable failure," which in the past led to a high ranking for certain political figures' pages due to a coordinated effort by web users to link these pages with that derogatory term.

Numerical Illustrations

Basic Web Graph Example: Consider a simplified web graph with three pages A, B, and C. Let A link to both B and C, B links to C, and C links to A.

Page	Links To
A	B, C
B	C
C	A

- Page A votes for B and C by linking to them.

- Page B votes for C.
- Page C votes for A.

In this setup:

- A and B have one in-link each, while C has two in-links, suggesting C is more authoritative than A or B.

In this way, the number of in-links provides a basic measure of the relative importance of a page.

Upcoming Topics in Later Parts:

1. **Part 2**: Introduction to PageRank, Markov Chains, and how link analysis influences ranking algorithms.
2. **Part 3**: Topic-specific PageRank and the HITS algorithm for authority and hub score computations.
3. **Part 4**: Practical considerations, spam detection in link analysis, and advanced applications in information retrieval.

Part 2: PageRank and Markov Chains

PageRank Overview

PageRank is one of the most well-known link analysis algorithms, originally developed by Google's founders. It assigns a numerical score to each web page based on the structure of the web graph, giving higher scores to pages that are linked to more frequently by other important pages.

The core idea behind PageRank is that a link from one page to another is like a vote of confidence. However, not all votes are equal. Links from important pages carry more weight than links from less important pages.

Random Surfer Model

The PageRank algorithm is based on the idea of a "random surfer" who navigates the web by randomly clicking on links. The surfer begins at a random page and moves from one page to another by following links. If the surfer is on a page with no outgoing links, or if they get bored of clicking, they can "teleport" to any other page on the web.

Here's how the process works:

1. At each time step, the surfer moves to a random page by either:
   - Clicking a link on the current page.
   - Teleporting to a random page on the web with a certain probability (typically modeled as a small probability, such as 10%).

2. As the surfer continues this random walk over time, some pages will naturally be visited more often than others. Pages with more incoming links, especially from important pages, will tend to be visited more often.

3. The **PageRank** score of a page is proportional to the likelihood that the random surfer will land on that page after many steps.

Markov Chains and PageRank

The process of the random surfer navigating the web can be modeled as a **Markov chain**, which is a mathematical model of a system that transitions between states (in this case, web pages) with certain probabilities.

- **States**: Each web page is considered a state in the Markov chain.
- **Transition Matrix**: The probability of moving from one page to another is stored in a matrix, where each entry represents the probability of moving from one state (web page) to another. This matrix is called the **transition probability matrix**.

Markov Chains: Basic Concepts

1. **Transition Matrix (P)**: The matrix P stores transition probabilities between states (pages). The entry $P(i, j)$ represents the probability of transitioning from page i to page j.

   Example of a simple 3-page web graph:

$$P = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \end{bmatrix}$$

2. **Steady-State Distribution**: After many steps, the surfer reaches a steady state where the probability of being on any given page becomes constant. This steady-state probability distribution represents the **PageRank scores** of the pages.

3. **Teleportation**: To ensure that the Markov chain is **ergodic** (i.e., every page can be reached eventually, no matter where the surfer starts), the surfer is allowed to "teleport" to any random page with a small probability (typically 10-15%). This prevents the surfer from getting stuck on pages with no outgoing links.

PageRank Computation

The computation of PageRank involves solving for the **principal eigenvector** of the transition probability matrix, which corresponds to the steady-state probabilities of the Markov chain.

Let's break down the process:

1. **Transition Matrix**: Given a web graph, we create the adjacency matrix A, where each entry $A(i, j)$ is 1 if there is a link from page i to page j and 0 otherwise.

   Example:
   $$A = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

   Next, we normalize this matrix so that each row sums to 1 (representing the transition probabilities). For example, if a page has 2 outgoing links, the probability of following each link is 0.5.

2. **Adding Teleportation**: To account for teleportation, we adjust the transition matrix P by adding a small probability α that the surfer will randomly jump to any other page. This is done by modifying the transition matrix as follows:
   $$P = (1 - \alpha) \cdot A + \alpha \cdot \left(\frac{1}{N}\right)$$
   where N is the total number of pages, and α is the teleportation probability (e.g., $\alpha = 0.15$).

3. **Power Iteration**: To compute the PageRank scores, we repeatedly multiply an initial probability vector by the transition matrix until it converges to a steady-state vector (the PageRank vector).

   Example: If the initial vector is [1 0 0] (i.e., the surfer starts at page 1), we repeatedly compute the new vector as $v = v \cdot P$ until the vector changes by less than a small threshold.

Numerical Example: Simple PageRank Computation

Consider the following small web graph with three pages A, B, and C, where:

- A links to B and C.
- B links to C.
- C links to A.

Step 1: Adjacency Matrix (A)

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Step 2: Transition Matrix (P with $\alpha = 0.15$)

$$P = (1 - 0.15) \cdot A + 0.15 \cdot \left(\frac{1}{3}\right)$$

After performing the matrix operations, we get the following transition matrix:

$$P = \begin{bmatrix} 0.05 & 0.90 & 0.90 \\ 0.05 & 0.05 & 0.90 \\ 0.90 & 0.05 & 0.05 \end{bmatrix}$$

Step 3: Power Iteration Start with an initial probability vector:

$$v_0 = [1\ 0\ 0]$$

Multiply v_0 by the transition matrix P to get v_1:

$$v_1 = v_0 \cdot P = [0.05\ 0.90\ 0.90]$$

Continue multiplying by P until the vector converges to a steady state. After several iterations, we reach the PageRank vector:

$$PR = [0.33\ 0.33\ 0.33]$$

In this simple example, each page ends up with an equal PageRank score, reflecting the symmetry of the web graph.

Key Properties of PageRank

1. **Query-Independent**: PageRank is a **query-independent** ranking signal, meaning that it does not depend on the specific search query but is instead based on the structure of the web.

2. **Importance of In-Links**: Pages that are linked to by many other important pages will receive a high PageRank score. However, not all links are equal—links from high-ranking pages carry more weight than links from low-ranking pages.

3. **Damping Factor**: The teleportation probability $(1 - \alpha)$ is called the **damping factor** and typically set to 0.85. This reflects the probability that a user will continue clicking on links rather than teleporting to a random page.

Exercises for Further Understanding

Exercise 21.6:

- **Question**: Consider a web graph with three nodes 1, 2, and 3. The links are as follows: $1 \to 2$, $3 \to 2$, and $2 \to 1$, $2 \to 3$. Write down the transition probability matrix for the surfer's walk with teleporting, for the following three values of the teleport probability: (a) $\alpha = 0$; (b) $\alpha = 0.5$; (c) $\alpha = 1$.

- **Answer**:

- (a) For $\alpha = 0$, the transition matrix is just the normalized adjacency matrix without any teleportation.
- (b) For $\alpha = 0.5$, the transition matrix will be a weighted sum of the adjacency matrix and the teleportation matrix (each page has a 50% chance of teleporting).
- (c) For $\alpha = 1$, the transition matrix becomes a uniform matrix where every page has an equal chance of being visited (i.e., teleportation dominates).

This completes Part 2. In **Part 3**, we will cover **topic-specific PageRank** and the **HITS algorithm** for identifying both hubs and authorities in web graphs. Let me know if you are ready to proceed!

Part 3: Topic-Specific PageRank and HITS Algorithm

In this part, we will explore **topic-specific PageRank** and the **HITS algorithm** (Hyperlink-Induced Topic Search). Both are advanced techniques used in link analysis, designed to handle more specialized web search problems.

Topic-Specific PageRank

Basic Idea: Topic-specific PageRank modifies the standard PageRank algorithm to prioritize pages related to a particular topic or interest. Instead of treating all web pages equally during teleportation, we bias the random surfer toward pages within a specific topic.

For example, if a user is interested in sports, the random surfer would prefer to teleport to sports-related pages more frequently than other pages, thereby increasing the PageRank scores of pages within that topic.

How Topic-Specific PageRank Works:

1. **Collection of Topic-Specific Pages**: First, we collect a subset of web pages related to the specific topic of interest (e.g., sports, politics, science). This collection can be built manually, using directories like the Open Directory Project (DMOZ), or automatically by analyzing content.

2. **Biasing the Teleportation**: In standard PageRank, the teleportation step allows the random surfer to jump to any page with equal probability. In topic-specific PageRank, we modify the teleportation probability to favor pages from the topic-specific collection. For example, when teleporting, the surfer might jump to a sports-related page 60% of the time and to a non-sports page 40% of the time.

3. **Personalized PageRank**: The idea can be extended to **personalized PageRank**, where the teleportation bias is based on a user's individual

preferences. For instance, a user might have an interest mixture of 70% sports and 30% technology, and the PageRank scores would reflect this preference.

Numerical Illustration of Topic-Specific PageRank:

Consider a small web graph with pages A, B, C, and D, where:

- Pages A, B, and C are sports-related.
- Page D is not sports-related.

The teleportation matrix for standard PageRank would look like this (assuming $\alpha = 0.15$):

$$T_{\text{standard}} = [0.25\ 0.25\ 0.25\ 0.25]$$

For topic-specific PageRank (with a 60% bias toward sports pages):

$$T_{\text{topic}} = [0.20\ 0.20\ 0.20\ 0.40]$$

The biased teleportation gives more weight to the topic-specific pages (A, B, and C). Over time, this bias increases their PageRank scores relative to Page D.

Advantages of Topic-Specific PageRank:

- **More Relevant Results**: Users searching within a particular domain (e.g., sports) get higher-ranked results related to that domain.
- **Improved Personalization**: Search engines can tailor results based on a user's interests or browsing history.

The HITS Algorithm (Hyperlink-Induced Topic Search)

The **HITS algorithm** is an alternative to PageRank that assigns two scores to each page: a **hub score** and an **authority score**. It's particularly useful for identifying authoritative pages and the hubs that link to them.

Key Concepts in HITS:

1. **Authorities**: Pages that provide valuable information on a specific topic (e.g., the National Cancer Institute page on leukemia). These are the pages that many other pages link to when discussing a topic.
2. **Hubs**: Pages that serve as directories or collections of links to authority pages. Good hubs point to many authoritative pages on a topic.

The Relationship Between Hubs and Authorities:

- **Good Hubs Link to Good Authorities**: A hub page is useful if it links to authoritative pages.
- **Good Authorities are Linked by Good Hubs**: A page is authoritative if many good hubs link to it.

This creates a mutually reinforcing relationship between hubs and authorities, which is captured by the iterative computation of hub and authority scores.

HITS Algorithm Steps:

1. **Initial Scores**: Assign every page in the web graph an initial hub and authority score (usually 1 for all pages).
2. **Iterative Updates**:
   - Update the **hub score** of each page to be the sum of the authority scores of the pages it links to.
   - Update the **authority score** of each page to be the sum of the hub scores of the pages that link to it.

   These updates are iterated until the hub and authority scores converge.

Mathematical Formulation:

Let $h(v)$ denote the hub score of page v, and $a(v)$ denote the authority score of page v.

- **Hub Update**: The hub score of page v is the sum of the authority scores of the pages it links to:
$$h(v) = \sum_{v \to y} a(y)$$
- **Authority Update**: The authority score of page v is the sum of the hub scores of the pages that link to it:
$$a(v) = \sum_{y \to v} h(y)$$

Example of HITS Calculation:

Consider the following small web graph:

Page	**Links to**
A	B, C
B	C
C	A

Initial Scores: Set $h(A) = h(B) = h(C) = 1$ and $a(A) = a(B) = a(C) = 1$.
First Iteration:

- Update hub scores:

$$h(A) = a(B) + a(C) = 1 + 1 = 2, \quad h(B) = a(C) = 1, \quad h(C) = a(A) = 1$$

- Update authority scores:

$$a(A) = h(C) = 1, \quad a(B) = h(A) = 2, \quad a(C) = h(A) + h(B) = 2 + 1 = 3$$

Second Iteration: Repeat the process until the hub and authority scores stabilize.

HITS vs. PageRank:

- **HITS**:
  - Computes two scores (hub and authority).
  - Scores depend on the specific query.
  - Focuses on identifying both authoritative pages and the hubs that point to them.
- **PageRank**:
  - Computes a single, query-independent score.
  - Focuses on identifying globally important pages based on the web's link structure.

Exercise for HITS:

Exercise 21.19:

- **Question**: If all the hub and authority scores are initialized to 1, what is the hub/authority score of a node after one iteration?
- **Answer**: After one iteration, the hub score of a node will be the sum of the authority scores of the nodes it points to, and the authority score of a node will be the sum of the hub scores of the nodes pointing to it.

Choosing the Subset of the Web for HITS:

In practice, it's computationally infeasible to run HITS on the entire web. Instead, HITS is run on a **subset** of the web graph around a specific query topic. This subset includes:

1. **Root Set**: Pages that contain the query term (e.g., pages that contain the word "leukemia").

2. **Base Set**: The root set, plus pages that link to or are linked by the root set.

By expanding the root set to the base set, HITS captures both hubs and authorities related to the query, even if some important pages do not contain the query term directly.

Conclusion of Part 3:

- **Topic-Specific PageRank** helps to personalize search results by focusing on a user's topic of interest.
- **HITS** provides a dual scoring system (hubs and authorities) to identify both authoritative sources and useful directories of links.
- Both methods extend traditional link analysis to address more specialized web search needs, improving relevance and personalization.

—

In **Part 4**, we will delve into practical considerations of implementing PageRank and HITS, as well as methods for combating spam (e.g., link farms) in link analysis. Let me know when you're ready to continue!

Part 4: Practical Considerations, Spam Detection, and Advanced Applications in Link Analysis

In this final part, we will focus on the practical aspects of implementing PageRank and HITS, discuss methods to detect and combat spam in link analysis, and explore advanced applications in information retrieval.

1. Practical Considerations in Implementing PageRank and HITS

PageRank Computation

The computation of PageRank can be complex, particularly for large-scale web graphs with billions of pages. Here are some key considerations:

1. **Sparse Matrices**: The web graph is typically very sparse, meaning that most web pages have relatively few links compared to the total number of pages. This allows for efficient storage and computation using sparse matrix representations.
2. **Power Iteration Method**: A common technique for computing PageRank is **power iteration**, where we start with an initial guess for the

PageRank vector (usually an equal probability for all pages) and iteratively multiply this vector by the transition probability matrix until it converges.

3. **Convergence Criteria**: The algorithm stops when the PageRank vector changes by less than a small threshold (e.g., less than 0.0001) from one iteration to the next. In practice, this usually requires 50-100 iterations for a large web graph.

4. **Handling Dead Ends**: Dead ends are pages with no outgoing links, which can cause problems for the random surfer model. When the surfer lands on a dead-end page, we teleport to a random page. This is handled by modifying the transition matrix to include teleportation probabilities.

5. **Handling Spider Traps**: A spider trap is a group of pages that only link to each other and not to any outside pages. The random surfer could get trapped in this group, causing these pages to accumulate an artificially high PageRank. The teleportation step prevents spider traps from dominating the PageRank scores.

6. **Efficient Storage and Computation**: For very large web graphs, it may not be possible to store the entire matrix in memory. Instead, we use algorithms that can process the matrix in chunks, reading it from disk as needed. Distributed computing frameworks like MapReduce are often used to parallelize the computation across multiple machines.

HITS Algorithm Computation

1. **Subset of the Web**: The HITS algorithm is typically run on a subset of the web graph around a specific query (the **root set** and **base set**). This reduces the size of the graph, making it more computationally feasible.

2. **Iterative Updates**: The hub and authority scores are computed iteratively, similar to PageRank. However, since both hub and authority scores are computed at each step, the computation involves two passes through the graph per iteration (one to update hub scores and one to update authority scores).

3. **Convergence Criteria**: As with PageRank, the algorithm stops when the hub and authority scores change by less than a small threshold from one iteration to the next.

4. **Sparse Matrix Representation**: Like PageRank, the web graph in HITS is represented as a sparse matrix, allowing for efficient storage and computation.

Scaling and Parallelization

For large-scale web search engines, both PageRank and HITS need to be scalable and parallelizable. Distributed computing techniques like **Hadoop MapReduce** or **Apache Spark** are commonly used to distribute the computation across multiple servers.

2. Spam Detection in Link Analysis

One of the major challenges in link analysis is the presence of **link spam**, where webmasters manipulate the link structure of the web to artificially boost the rankings of their pages. Two common forms of link spam are **link farms** and **spammy anchor text**.

Link Farms

Link farms are groups of interconnected web pages created with the sole purpose of inflating the PageRank of certain pages. These pages link to each other in a way that artificially boosts their authority.

How Link Farms Work:

- A webmaster creates multiple web pages and links them to a target page.
- These pages are often hosted on different domains to appear more legitimate.
- The goal is to increase the number of in-links to the target page, boosting its PageRank.

Combating Link Farms

Search engines employ several techniques to detect and combat link farms:

1. **Link Structure Analysis**: Link farms typically have unnatural linking patterns, such as many low-quality pages linking to each other. Search engines can detect these patterns by analyzing the topology of the web graph.
2. **Link Quality Metrics**: Instead of counting all links equally, modern search engines use link quality metrics that consider the trustworthiness of the linking pages. Pages linked to by trusted, authoritative sites are given more weight than those linked by unknown or suspicious sites.
3. **Spam Detection Algorithms**: Several algorithms have been developed to detect and penalize link farms, including **TrustRank** and **SpamRank**. These algorithms propagate trust (or distrust) through the web graph, reducing the influence of spammy pages.

Anchor Text Spam

Anchor text can be manipulated to boost the ranking of a page for specific keywords. For example, spammers may create many links with the same anchor text (e.g., "best car deals") pointing to a target page, even if the page is not relevant to that keyword.

Combating Anchor Text Spam

1. **Anchor Text Analysis**: Search engines analyze the diversity of anchor text pointing to a page. If many links use identical or similar anchor text, it may be a sign of manipulation.
2. **Contextual Analysis**: Search engines consider the context around the anchor text, not just the anchor text itself. Pages with suspiciously repetitive or irrelevant surrounding text may be flagged as spam.
3. **Spam Filtering**: Search engines apply filters to detect and remove the influence of spammy anchor text. This may involve reducing the weight of anchor text in the ranking algorithm for certain pages or queries.

3. Advanced Applications in Information Retrieval

In addition to their use in web search, link analysis techniques like PageRank and HITS have been applied to various other areas of information retrieval.

Social Network Analysis

Link analysis techniques are widely used in **social network analysis** to identify influential individuals (nodes) in a network. For example:

- **PageRank** can be used to rank users based on their importance in the network.
- **HITS** can identify users who act as hubs (who point to other important users) and authorities (who are referenced by others).

Scientific Citation Networks

In **scientific citation networks**, papers can be treated as nodes, and citations as directed edges. PageRank-like algorithms are used to measure the importance of academic papers, similar to how they are used to rank web pages.

- A paper cited by many highly cited papers will have a high PageRank, reflecting its influence in the academic community.
- The HITS algorithm can be used to identify **hub papers** (which cite many authoritative papers) and **authority papers** (which are cited by many hub papers).

Recommender Systems

Link analysis algorithms can also be applied to **recommender systems**. For instance, PageRank can be used to rank items (e.g., movies, products) based on user interactions (e.g., clicks, likes). The underlying graph in such systems consists of users and items, with edges representing user interactions with items.

Information Propagation in Networks

In studies of **information propagation**, link analysis can help identify key nodes (users, pages, etc.) that are most likely to spread information. This is useful in fields like viral marketing, where companies aim to identify influencers who can help spread their messages to a large audience.

Exercises for Further Practice

- **Exercise 21.10**: **Question**: Show that the PageRank of every page is at least α/N. What does this imply about the difference in PageRank values (over the various pages) as α becomes close to 1? **Answer**: As α becomes closer to 1, the teleportation step dominates, and every page receives roughly the same PageRank. This reduces the difference in PageRank values between pages, making the algorithm less sensitive to the link structure.
- **Exercise 21.22**: **Question**: For a given web graph, compute the PageRank, hub, and authority scores for each of the pages. Also, give the relative ordering of the nodes for each of these scores, indicating any ties. **Hint**: Using symmetries to simplify and solving with linear equations might be easier than using iterative methods.

Conclusion of Part 4

- **Practical Considerations**: Both PageRank and HITS require efficient algorithms and data structures to handle the massive scale of the web. Sparse matrices, distributed computing, and iterative methods like power iteration are essential for large-scale implementations.
- **Spam Detection**: Link farms, anchor text spam, and other manipulative practices can distort rankings. Modern search engines use sophisticated algorithms to detect and combat spam, ensuring that the rankings remain reliable.
- **Advanced Applications**: Link analysis extends beyond web search to social networks, citation networks, recommender systems, and information propagation.

This concludes the in-depth explanation of the topics in **link analysis**. If you have any further questions or need additional elaboration on any part, feel free to ask!