TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

THUẬT TOÁN PAGERANK TRONG CÔNG CỤ TÌM KIẾM GOOGLE

ĐỒ ÁN II

Chuyên ngành: TOÁN TIN

Chuyên sâu: Tin học

Giảng viên hướng dẫn: **THS. NGUYỄN DANH TÚ**

Sinh viên thực hiện: **VŨ MINH PHONG**

Lớp: **KSTN Toán Tin - K61**

HÀ NỘI - 2020

Mục lục

Lời mở đầu

Ngày nay, trong thời đại thông tin, khối lượng thông tin và tri thức của nhân loại gần như là vô tận. Nếu như không có các công cụ tìm kiếm như Google thì việc tìm kiếm và thu thập thông tin trên Web là vô cùng khó khăn. Dựa trên những kiến thức cơ bản về ma trận, vector riêng và xích Markov, Sergey Brin và Larry Page đã phát minh ra một công cụ tìm kiếm làm thay đổi mãi mãi cách thức con người tiếp cận thông tin. Điều này không chỉ có ý nghĩa cho sự phát triển của tri thức nhân loại, mà còn tạo ra cơ hội phát triển kinh tế bằng việc rút ngắn khoảng cách giữa khách hàng và doanh nghiệp. Google đã giúp cho việc tiếp cận thông tin trên Web dễ dàng hơn và trái tim của cỗ máy tìm kiếm Google chính là thuật toán PageRank.

Mục đích của đồ án là giới thiệu và tìm hiểu cách thức hoạt động của PageRank cũng như đưa ra các ứng dụng trong kinh doanh và marketing.

Em xin chân thành cảm ơn thầy giáo ThS. Nguyễn Danh Tú đã tận tình hướng dẫn em hoàn thành đồ án này. Em xin cảm ơn các thầy cô trong Viện Toán ứng dụng và Tin học, Trường Đại học Bách Khoa Hà Nội, đã giúp đỡ em trong quá trình học tập. Báo cáo đồ án này hẳn vẫn còn các thiếu sót, em mong nhận được nhận xét của thầy cô và các bạn đọc. Em xin chân thành cảm ơn!

Chương 1

Tổng quan về các công cụ tìm kiếm

Trong chương này em sẽ trình bày về các công cụ tìm kiếm. Nội dung trình bày bao gồm:

- Lược sử về thu thập thông tin
- Tổng quan về các phương pháp thu thập thông tin cổ điển
- Tìm kiếm thông tin trên Web
- Xếp hạng trang web dựa trên độ phổ biến

Các nội dung trình bày được tham khảo từ nguồn [1], [2], [3]

1.1 Lược sử về thu thập thông tin

Thu thập thông tin là quá trình tập hợp thông tin theo những tiêu chí cụ thể nhằm làm rõ những vấn đề, nội dung liên quan đến lĩnh vực nhất định. Mặc dù chỉ phát triển mạnh theo sau những tiến bộ vượt bậc của khoa học có được từ sự ra đời của máy tính điện tử, ngành khoa học thu thập thông tin đã hình thành và phát triển từ trước đó rất lâu.

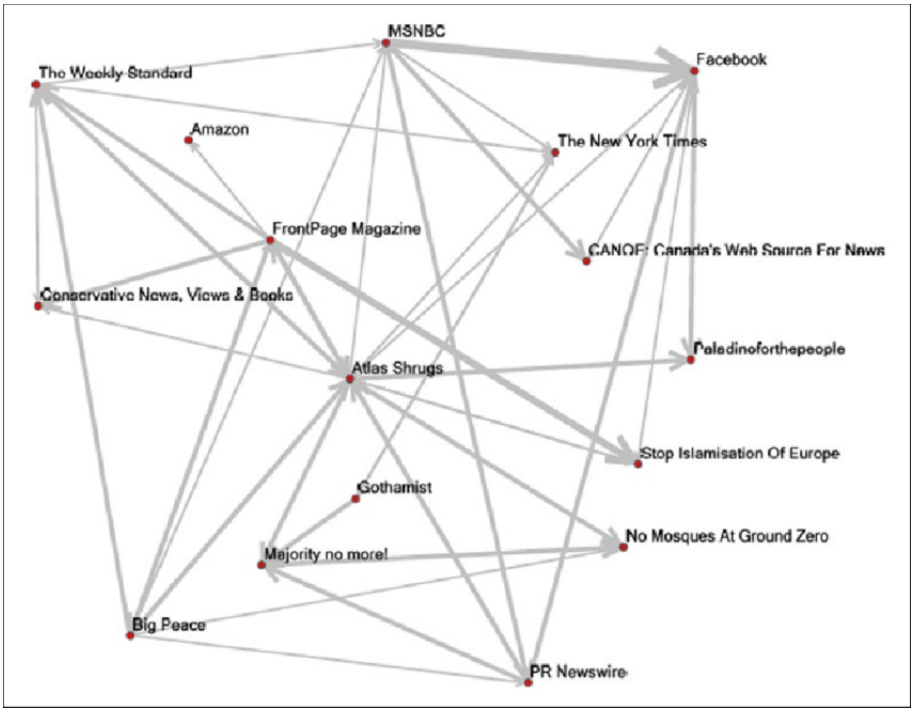
Hình thức ghi lại thông tin đầu tiên của con người là những bức tranh trên bức tường trong các hang động. Con người tiền sử muốn tìm kiếm một thông tin nào đó sẽ phải đi

đến từng bức tường, đứng trước chúng và nhìn vào đó để lấy thông tin. Sau này khi xã hội con người phát triển hơn, người Lã Mã và Hy Lạp cổ đại đã ghi lại thông tin ra các cuộn giấy cói. Chúng có thể đính kèm các thẻ tên. Mỗi thẻ tên đính kèm ghi tóm tắt nội dung của cuộn giấy để người đọc không phải mở những cuộn giấy chứa thông tin không liên quan. Ngoài ra, thông tin còn được lưu trữ qua đường truyền miệng thông qua các vở kịch và các bài hát kể về các sự kiện lịch sử. Sách truyền thống được phát minh sau đó vài thế kỉ. Những quyển sách này bền lâu hơn cuộn giấy cói và dễ sử dụng hơn sớm thay thế toàn bộ giấy cói sau này.

Khi mà các thư viện ngày càng to lớn và mở cửa đối với công chúng, nhu cầu tìm kiếm nhanh và chính xác cũng phát triển theo. Hệ thống phân loại cấp bậc được dùng để nhóm các văn bản có cùng chủ đề với nhau. Mặc dù được hệ thống như vậy, truyền miệng và lời hướng dẫn của người thủ thư vẫn cách tìm kiếm thông tin chính xác nhất và tốt nhất. Đương nhiên một người thủ thư không thể ghi nhớ toàn bộ vị trí và tiêu đề của cả một thư viện. Nhiều cách sắp xếp và tổ chức lưu trữ tài liệu tốt hơn được đề ra

Công việc tìm kiếm được cải thiện đáng kể, nhưng với khối lượng tri thức nhân loại ngày càng lớn thì vẫn không đáp ứng được nhu cầu. Phải đến khi máy tính điện tử ra đời và cùng với đó là hệ thống tìm kiếm trên máy tính ta mới thấy bước chuyển tiến rõ rệt. Hệ thống tìm kiếm trên máy tính đầu tiên sử dụng các cú pháp đặc biệt để tự động thu thập thông tin của sách và các bài báo liên quan đến câu hỏi của người dùng. Tuy nhiên, các cú pháp này còn cồng kềnh và chỉ được dùng bởi những người thủ thư đã qua đào tạo.

Vào năm 1989, mạng toàn cầu (World Wide Web) ra đời đã bắt đầu một cuộc cách mạng trong lưu trữ, truy cập và tìm kiếm thông tin. Mạng toàn cầu, gọi tắt là Web là một phát minh của nhà khoa học người Anh Tim Berners-Lee vào năm 1989. Nó là hệ thống thông tin mà trong đó các văn bản và các tài nguyên khác được nhận dạng bằng các đường dẫn (URL - Uniform Resource Locator) và có thể liên kết tới nhau bằng các hyperlink (siêu liên kết) và có thể truy cập qua Internet. Berners-Lee nảy ra ý tưởng về Web sau khi cảm thấy rắc rối mỗi lần ông muốn tìm kiếm thông tin trên nhiều máy tính. Trong ý tưởng ban đầu của mình, ông mô tả một hệ thống quản lý thông tin dựa trên các đường liên kết được gắn vào trong các văn bản, để khi người đọc gặp các tài liệu tham khảo trong một văn bản thì có thể chuyển tức thời đến tài liệu được đề cập chỉ với một cú kích chuột vào các đường dẫn Mặc dù mạng toàn cầu là một đột phá trong lưu trữ và truy cập thông tin, người dùng lại gặp khó khăn trong việc tìm kiếm. Khối lượng thông tin là khổng lồ và càng ngày càng mở rộng trong khi lượng thông tin người dùng muốn tìm kiếm và truy cập là rất nhỏ, công việc tìm kiếm được ví như mò kim đáy bể. Hầu hết thông tin trên Web lúc bấy giờ là gần như không thể tiếp cận



Hình 1.1: Một tập văn bản được liên kết với nhau bằng các hyperlink.

Mọi chuyện đều thay đổi khi phân tích liên kết ra đời vào năm 1998. Phân tích liên kết (Link Analysis) là một kỹ thuật phân tích dữ liệu dùng để đánh giá các qua hệ hoặc các liên kết giữa các nút trong một mạng lưới. Những công cụ tìm kiếm bắt đầu sử dụng phân tích liên kết để cải thiện kết quả tìm kiếm. Hiệu quả đem lại là rõ rệt và nhiều công cụ tìm kiếm dần trở thành thú vui ưa thích của nhiều người dùng mạng. Họ sử dụng công cụ tìm kiếm để tìm kiếm tin tức từ khắp nơi trên thế giới, để kiểm tra lỗi chính tả khi sử dụng tiếng nước ngoài, để tìm kiếm các hình ảnh. Google là một trong những công cụ tìm kiếm đầu tiên sử dụng phân tích liên kết và đến năm 2004, Google chiếm thị phần lớn nhất trên thị trường tìm kiếm với 37% số tìm kiếm được thực hiện bởi Google, tiếp sau là 27% thực hiện bởi Yahoo.

Hầu hết các công cụ tìm kiếm ngày nay đều áp dụng phân tích liên kết trong các thuật toán chấm điểm và xếp hạng các trang web. Và thuật toán nổi tiếng nhất cũng đã làm nên thương hiệu Google là thuật toán PageRank.

1.2 Tổng quan về phương pháp thu thập thông tin cổ điển

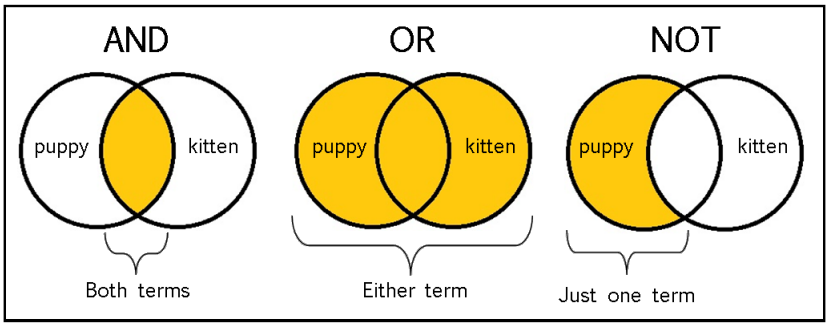
Trước khi có phân tích liên kết, người ta sử dụng các phương pháp thu thập thông tin cổ điển. Trong khi thu thập thông tin trên Web được thực hiện trên một khối tài liệu khổng lồ được liên kết với nhau thì thu thập thông tin cổ điển chỉ tìm kiếm trên một bộ tài liệu nhỏ hơn. Tìm kiếm thông tin trong thư viện của trường đại học Bách Khoa là một ví dụ của thu thập thông tin cổ điển.

Những bộ tài liệu không liên kết này thường là tĩnh và được phân loại bởi những người có chuyên môn như thủ thư và biên tập. Chúng được lưu trữ dưới dạng vật chất như sách, báo, tranh ảnh và cả dạng điện tử như đĩa CD, thẻ nhớ và các trang web. Tuy nhiên phương thức tìm kiếm trên các bộ tài liệu này hầu như được thực hiện trên máy tính. Các phương thức này được gọi là công cụ tìm kiếm (search engine), là các cỗ máy ảo tạo ra bởi phần mềm giúp chúng duyệt qua các thư mục trên máy tính để tìm các tài liệu có liên quan. Có 3 phương thức tìm kiếm cơ bản trong các bộ tài liệu không liên kết: Mô hình Boolean, mô hình không gian vector và mô hình xác suất.

1.2.1 Công cụ tìm kiếm sử dụng mô hình Boole

Mô hình tìm kiếm Boole là mô hình sớm nhất và đơn giản nhất trong thu thập thông tin và vẫn được sử dụng ở hầu hết các thư viện ngày nay. Mô hình Boole các toán tử Boole AND, OR, NOT kết hợp các từ khoá. Mô hình tìm kiếm Boole hoạt động bằng việc xét xem những từ khoá nào có mặt hay không có mặt trong văn bản và dựa vào đó kết luận xem văn bản là liên quan hay không liên quan. Tuy nhiên mô hình này có nhược điểm là không có trường hợp thứ 3 là liên quan một phần. Ví dụ như khi tìm kiếm những quyển sách có tiêu đề chứa hai từ là **car** và **maintenance** bằng mô hình Boole thì mô hình sẽ trả về tất cả những quyển sách có tiêu đề chứa cả hai từ đó. Thế nhưng quyển sách có tiêu đề "Automobile Mantenance" mặc dù có liên quan sẽ không được trả về. Khi sử dụng lý thuyết mờ thì quyển sách trên sẽ được đánh dấu là có liên quan và được trình tìm kiếm trả về cho người dùng.

Nhược điểm lớn nhất của mô hình Boole là từ đồng nghĩa và từ nhiều nghĩa. Mô hình Boole không thể nhận biết hai từ nào là giống nhau về mặt ngữ nghĩa, như trong ví dụ trên **car** và **automobile** là hai từ đồng nghĩa nhưng vẫn được coi là khác nhau. Hơn nữa kết quả tìm kiếm có thể không chính xác bởi kết quả trả về có thể chứa những từ cùng



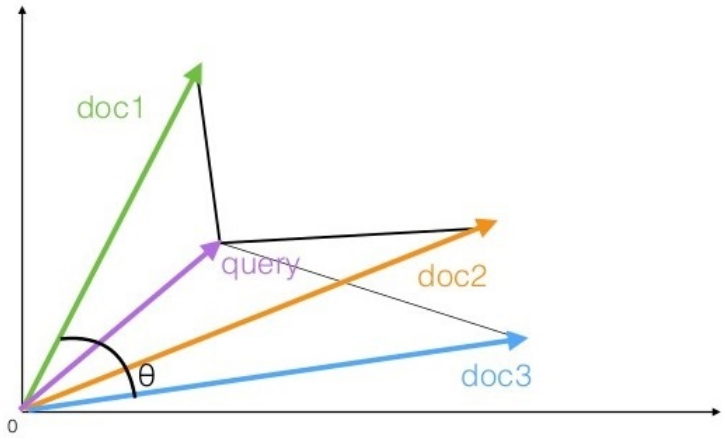
Hình 1.2: Các toán tử Boole

âm nhưng khác nghĩa.

Dù có nhiều hạn chế nhưng những biến thể của mô hình tìm kiếm Boole vẫn có ứng dụng rộng rãi. Đó là vì thứ nhất, việc cài đặt và lập trình một công cụ tìm kiếm Boole không khó khăn và rắc rối. Thứ hai, các yêu cầu tìm kiếm có thể xử lý một cách nhanh chóng, có thể xử lý song song dễ dàng. Thứ ba, mô hình Boole hoạt động tốt khi khối lượng tài liệu lớn.

1.2.2 Công cụ tìm kiếm sử dụng mô hình không gian vector

Mô hình không gian vector biến đổi dữ liệu kiểu chữ thành các vector số và các ma trận, rồi sau đó sử dụng các kỹ thuật xử lý ma trận để tìm ra những đặc điểm nổi bật. Một số mô hình không gian vector phức tạp có thể giải quyết các vấn đề về từ đồng nghĩa và nhiều nghĩa. Ví dụ như mô hình LSI (Latent Semantic Idexing), có thể hiểu được ngữ nghĩa truy vấn của người dùng. Trong ví dụ về **car** và **maintenance** thì LSI có thể trả về kết quả **Automobile maintenance**. Khả năng hiểu được ngữ nghĩa này khiến cho LSI được ứng dụng trong hầu hết công cụ tìm kiếm ngày nay. Mô hình không gian cho phép các tài liệu được giống một phần với câu truy vấn bằng việc gán cho các tài liệu một số từ 0 đến 1, đại diện cho múc độ liên quan đến câu truy vấn. Tập hợp các tài liệu sau đó có thể được sắp xếp theo mức độ liên quan, điều mà mô hình Boole không thể làm được. Vì vậy nên mô hình không gian vector trả về một danh sách đã được sắp xếp. Tài liệu đầu tiên được trả về được coi là có liên quan nhất đến câu tìm kiếm của người dùng. Phản hồi liên quan là một ưu điểm nữa của mô hình không gian vector, là một phương pháp tinh chỉnh hoạt động của mô hình. Phản hồi liên quan cho phép người dùng lựa chọn một tập con của tập các tài liệu được trả về mà theo người dùng là tốt nhất. Và trình tìm kiếm sẽ dựa vào đó, trả về kết quả tìm kiếm mới chứa nhiều tài liệu tốt hơn theo phản hồi của người dùng.



Hình 1.3: Mô hình không gian vector

Nhược điểm lớn nhất của mô hình không gian vector là chi phí tính toán. Sau khi nhận được yêu cầu truy vấn của người dùng, trình tìm kiếm phải tính toán mức độ liên quan (thường dựa trên khoảng cách trong không gian vector) cho từng tài liệu một. Còn các mô hình phức tạp như LSI, cần phải sử dụng thuật toán SVD (Singular Value Decomposition) cho một ma trận lớn đại diện cho toàn bộ tập hợp các tài liệu. Khi tập hợp này lớn dần thì mô hình không gian vector hoạt động kém khi thời gian tính toán tăng lên rất nhanh. Mô hình này chỉ phát huy hiệu quả ở quy mô nhỏ.

1.2.3 Công cụ tìm kiếm sử dụng mô hình xác suất

Các công cụ tìm kiếm sử dụng mô hình xác suất mong muốn ước lượng được xác suất mà người dùng sẽ đánh giá một tài liệu là có liên quan. Các tài liệu được thu thập và trả về được xếp hạng theo khả năng liên quan (là tỷ lệ giữa xác suất tài liệu có liên quan tới câu truy vấn chia cho xác suất mà tài liệu không liên quan tới câu truy vấn). Mô hình xác suất là mô hình đệ quy và yêu cầu ta phải có một ước lượng ban đầu, và sau đó lặp lại các tính toán để cải thiện ước lượng này cho đến khi đạt được ước lượng cuối cùng là khả năng liên quan của các tài liệu đối với câu truy vấn của người dùng.

Không may rằng, mô hình xác suất rất khó xây dựng và lập trình. Độ phức tạp tính toán của mô hình này tăng nhanh, giới hạn khả năng của chúng trong thực tiễn. Hơn nữa, chúng còn yêu cầu các điều kiện không phù hợp với thực tế như sự độc lập xác suất giữa các từ và các tài liệu. Ví dụ như trong một tập truyện **Harry Potter** thì từ **phù** thường theo sau nó sẽ là từ **thuỷ** thế nhưng theo giả thiết độc lập về xác suất thì các từ được

Tiêu chí	**Mô hình Boole**	**Mô hình không gian vector**	**Mô hình xác suất**
Độ chính xác	Tài liệu tìm được hoặc đều liên quan hoặc đều không liên quan	Tìm được các tài liệu dựa theo các trọng số của các từ khoá	Tìm được các tài liệu dựa theo xác suất liên quan
Độ nhạy cảm	Thấp vì yêu cầu phải giống y hệt	Cao hơn mô hình Boole vì có thể giống một phần	Cao hơn mô hình Boole và không gian vector
Ưu điểm	Đơn giản	Đơn giản	Không bị giới hạn bởi các từ khoá
Nhược điểm	Không xếp hạng được các kết quả tìm kiếm	Phức tạp hơn mô hình Boole và tốc độ tính toán chậm	Phức tạp nhất và yêu cầu giả thiết độc lập xác suất

Bảng 1.1: Caption

coi là như nhau nên sau từ **phù** thì có thể là bất kỳ từ gì. Mặt khác, mô hình xác suất có thể tích hợp thêm sở thích của người dùng. Ví dụ như lịch sử tìm kiếm của người dùng có thể được dùng làm ước lượng ban đầu của mô hình xác suất, dẫn đến kết quả tìm kiếm tốt hơn là ước lượng một cách ngẫu nhiên.

1.2.4 So sánh giữa các công cụ tìm kiếm

Khi so sánh và đánh giá các công cụ tìm kiếm, người ta thường đánh giá dựa trên hai tiêu chí là độ chính xác (precision) và độ nhạy cảm (sensitivity). Độ chính xác là tỷ lệ giữa số tài liệu liên quan được trả về và tổng số tài liệu được trình tìm kiếm trả về. Còn độ nhạy cảm là tỷ lệ giữa số tài liệu liên quan được trả về và tổng số tài liệu liên quan trong toàn bộ tài liệu có trong tập hợp. Độ chính xác cao tức là các tài liệu tìm được có mức độ liên quan cao. Độ nhạy cảm cao tức là khả năng bỏ sót các tài liệu liên quan thấp. Sau đây ta có bảng so sánh 3 mô hình tìm kiếm trên.

1.3 Tìm kiếm thông tin trên Web

1.3.1 Những vấn đề trong tìm kiếm thông tin trên mạng

Trong phần trước ta mới chỉ đề cập đến các tập hợp tài liệu cổ điển. Sau khi Tim Berners-Lee phát minh ra World Wide Web, một hướng nghiên cứu mới đã được mở ra tập trung vào nghiên cứu các phương pháp tìm kiếm trên tập hợp tài liệu mới này và được gọi là thu thập thông tin trên web. Nhiều công cụ tìm kiếm được xây dựng dựa trên các phương pháp của các công cụ cổ điển nhưng chúng vẫn khác biệt nhau theo nhiều khía cạnh. Những đặc điểm khiến cho mạng Web lại khác biệt so với các tập hợp tài liệu thông thường là:

- Kích thước vô cùng lớn.
- Không ngừng thay đổi.
- Tự quản lý.
- Được kết nối bằng các siêu liên kết.

Mạng toàn cầu quả thực rất lớn, thậm chí với các công cụ thống kê hiện đại cũng không thể có được một ước lượng chính xác kích thước của nó. Mức độ phát triển của Web đã chậm lại trong những năm gần đây, nhưng Web vẫn là tập hợp tài liệu lớn nhất đang tồn tại trên thế giới. Thậm chí đa phần dữ liệu trên Web không thể truy cập được theo cách thông thường mà cần phải có những trình duyệt đặc biệt. Những dữ liệu này nằm ở khu vực được gọi là Deep Web, chiếm đến 99,996% tổng số dữ liệu nằm trên Web. Người sử dụng mạng muốn truy cập vào các trang Deep Web phải gửi yêu cầu đến một hệ cơ sở dữ liệu nào đó, sau đó các trang web liên quan mới được tạo ra và gửi trả về cho người dùng. Vì lý do này mà các công cụ tìm kiếm không thể truy cập vào các trang deep web dễ dàng, vì chúng không tồn tại cho đến khi được yêu cầu truy cập và sau khi truy cập vào thì các trang web này lại bị xoá đi.

Mạng Web liên tục thay đổi từng phút và từng giây. Khác với điều này, các bộ tài liệu thông thường có thể coi là tĩnh tại, không thay đổi theo hai cách. Thứ nhất, một khi một tài liệu, văn bản được thêm vào tập hợp, nó không thay đổi. Những quyển sách nằm trên giá sách của một thư viện không thay đổi được nội dung nhưng một trang web thì hoàn toàn có thể. Thứ hai, kích cỡ của một bộ tài liệu truyền thống có kích cỡ tương đối ổn định và không thay đổi nhiều so với Web. Dù mỗi năm một thư viện có thể có thêm vài

trăm, hay vài nghìn quyển sách thì sự thay đổi này cũng chỉ như hạt muối bỏ bể so với kích thước của Web. Hàng tỷ trang web được tạo ra mỗi năm.

Mạng Web có thể tự quản lý. Các bộ tài liệu truyền thống thường được thu thập và phân loại bởi những người có chuyên môn. Tuy nhiên, bất kỳ ai cũng có thể đăng tải một trang web. Dữ liệu trên Web không ổn định mà thay đổi liên tục, đôi lúc xuất hiện những đường dẫn lỗi, tệp bị mất. Dữ liệu tồn tại ở nhiều định dạng, ngôn ngữ và bảng chữ cái. Và vì không qua kiểm duyệt nên các thông tin sai lệch, các phát ngôn vô căn cứ tồn tại khắp nơi. Hơn nữa, khả năng tự quản lý này còn giúp cho những kẻ **spammer** lợi dụng để kiếm lời. *Spammer* là thuật ngữ ban đầu dùng để chỉ những người gửi hàng loạt những thư quảng cáo. Chỉ với một cú kích chuột, một spammer có thể gửi những thư quảng cáo đến hàng ngàn người chỉ trong vài giây. Khi có các công cụ tìm kiếm trên web và bắt đầu có bán hàng online, khái niệm spammer được mở rộng ra bao gồm cả những người sử dụng những kỹ thuật đặc biệt để đánh lừa các công cụ tìm kiếm đưa trang web lên thứ hạng cao khi người dùng tìm kiếm một cụm từ nào đó.

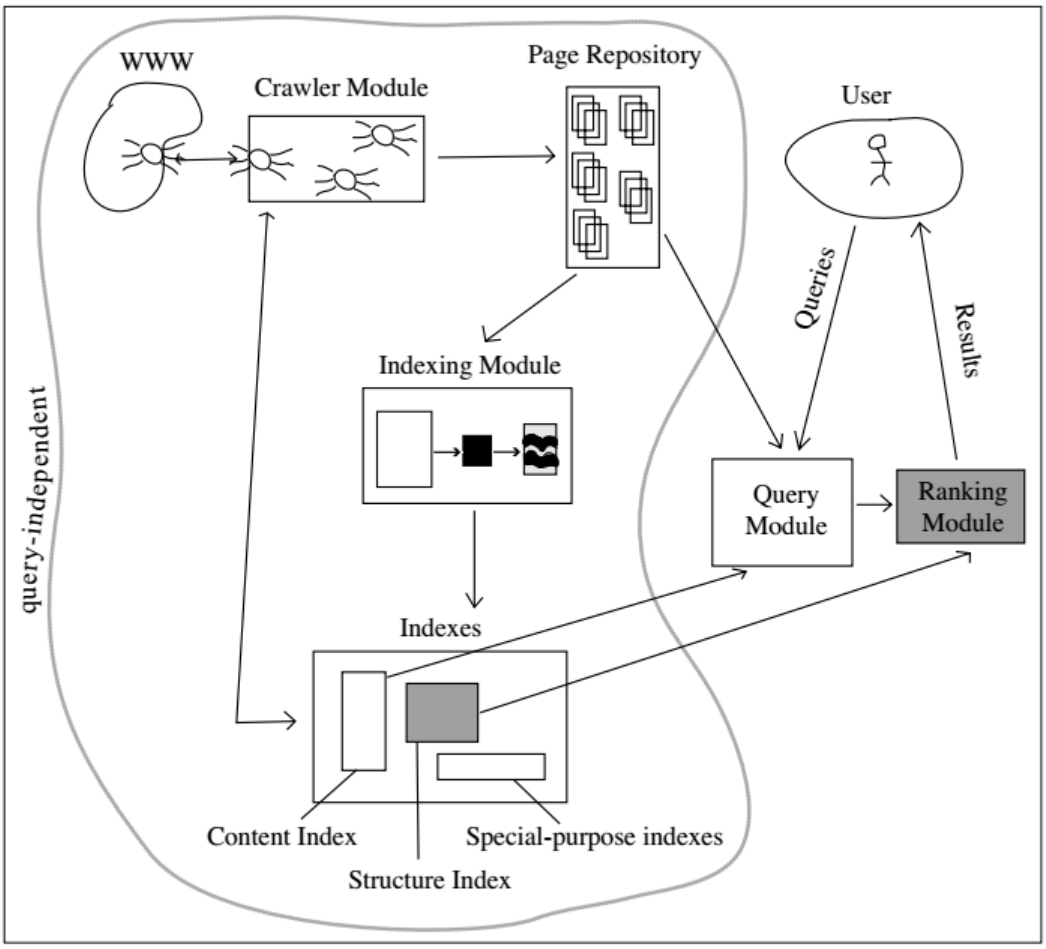
Nhưng điều khiến cho Web thực sự đặc biệt chính là các siêu liên kết giữa các trang web. Các công cụ tìm kiếm trên Web cũng dựa vào đặc điểm để xây dựng mô hình tìm kiếm. Nó cho phép quá trình tìm kiếm có thể tập trung hơn và hiệu quả hơn. Các công cụ tìm kiếm khai thác các thông tin tiềm ẩn nằm trong cấu trúc các đường dẫn trên Web để cải thiện kết quả tìm kiếm. Tuy nhiên, lợi thế có được từ cấu trúc các đường dẫn này cũng có các tác hại xấu. Các spammer sớm biết được ứng dụng của phân tích liên kết trong các công cụ tìm kiếm thịnh hành, và ngay lập tức họ bắt đầu thực hiện việc tạo các trang web giả chỉ chứa đường dẫn đến một trang web mong muốn để tăng thứ hạng của trang web đó trong kết quả tìm kiếm, hành vi này gọi là **link spamming**.

Mặc dù khối lượng thông tin sẵn sàng cho người dùng truy cập ngày càng tăng, khả năng người dùng tìm thấy những thông tin này thì lại không. Người dùng thường hiếm khi đọc quá 10 đến 20 kết quả trả về đầu tiên. Chính vì sự thiếu kiên nhẫn này mà các công cụ tìm kiếm phải ngày càng chính xác khi số lượng tài liệu tăng cao.

1.3.2 Các thành phần của một công cụ tìm kiếm trên Web

Các thành phần của một công cụ tìm kiếm web và mối quan hệ giữa chúng được minh hoạ trong hình 1.2. Trong đó trọng tâm của đồ án sẽ là mảng xếp hạng (ranking) của quá trình tìm kiếm.

- **Mô-đun Crawler (Crawler Module)**. Vì Web có khả năng tự quản lý nên khác



Hình 1.4: Các thành phần của một công cụ tìm kiếm trên Web

với các bộ tài liệu thông thường, Web không có một khu tập trung và không được tổ chức phân loại. Các kiểu tổ chức tài liệu truyền thống được đặt trong các nhà kho, các thư viện, hoặc bảo tàng, nơi chúng được phân loại và sắp xếp. Ngược lại, các tài liệu trên Web nằm ở trong các máy tính, một không gian ảo không bị giới hạn bởi không gian vật chất và có thể mở rộng vô cùng. Tuy nhiên để có được sự tự do về không gian địa lý phải đánh đổi bằng việc các công cụ tìm kiếm phải tự mình thu thập dữ liệu và phân loại chúng. Vì vậy mà tất cả các công cụ tìm kiếm đều có mô-đun crawler. Mô-đun này chứa các phần mềm để thu thập và phân loại các tài liệu trên web. Các phần mềm crawling tạo ra các con rô-bốt ảo, thường gọi là **spider**, liên tục khám phá Web để thu thập các thông tin mới các trang web để đem trở về và lưu trữ chúng ở trong một nơi chứa trung tâm.

- **Kho Chứa Trang (Page Repository)**. Các spider trở về cùng với các trang web

mới và tạm thời được lưu trữ đầy đủ thông tin trong kho chứa trang. Các trang web sẽ nằm trong đó cho đến khi chúng được gửi đến mô-đun chỉ mục, ở đó các thông tin quan trọng sẽ được trích xuất và tạo thành phiên bản thu gọn của trang. Các trang web phổ biến mà thường được truy cập và tìm kiếm sẽ nằm trong kho lâu hơn và có thể là vô hạn định.

- **Mô-đun Đánh Chỉ Mục(Indexing Module)**. Mô-đun đánh chỉ mục nhận được các trang web và trích xuất ra những thông tin quan trọng, tạo ra phiên bản thu gọn của trang web và lưu chúng ở trong các chỉ mục. Các trang web sau đó bị loại bỏ đi hoặc nếu nó là trang web phổ biến sẽ được đem trở lại vào trong kho chứa trang.

- **Chỉ Mục**. Các chỉ mục chứa các thông tin quan trọng của từng trang web. Có ba loại chỉ mục chính. Chỉ mục đầu tiên gọi là chỉ mục nội dung (content index). Trong này các nội dung như từ khoá, tiêu đề, và các anchor text của từng trang web được lưu trữ dưới dạng nén. Loại chỉ mục thứ hai là chỉ mục cấu trúc (structure index) chứa các thông tin về các đường hyperlink và kết cấu của chúng. Mô-đun crawler đôi khi truy cập chỉ mục cấu trúc để tìm những trang web chưa di qua. Cuối cùng ta có các chỉ mục đặc biệt (special purpose indexes), như chỉ mục ảnh (image index) và chỉ mục pdf.

Bốn mô-đun trên và tệp dữ liệu tương ứng của chúng tồn tại và hoạt động độc lập với người dùng và hoạt động tìm kiếm của họ. Các spider luôn luôn lướt qua các trang web, thu thập và mang về thông tin của những trang web mới hoặc những trang có cập nhật để có thể đánh chỉ mục và lưu trữ. Trong hình 1.2, các mô-đun này được khoang tròn và có nhãn là độc lập với truy vấn (**query-independent**). Khác với những mô-đun trên là mô-đun truy vấn (**query-module**), mô-đun này phục thuộc trực tiếp vào câu truy vấn và được sử dụng khi người dùng nhập câu tìm kiếm, và công cụ tìm kiếm phải trả lời trong thời gian thực.

- **Mô-đun Truy Vấn (Query module)**. Mô-đun truy vấn nhận câu hỏi của người dùng dưới dạng ngôn ngữ tự nhiên và chuyển chúng về ngôn ngữ mà máy tính có thể hiểu (thường là những con số), và tham khảo nhiều chỉ mục để trả kết quả về cho người dùng. Ví dụ như mô-đun truy vấn tham khảo chỉ mục nội dung để tìm xem trang nào có sử dụng cụm từ trong câu tìm kiếm. Những trang này là những trang liên quan. Sau đó mô-đun truy vấn chuyển những trang liên quan này đến mô-đun xếp hạng.

- **Mô-đun xếp hạng (Ranking Module)**. Mô-đun xếp hạng nhận những trang có liên quan và xếp hạng chúng dựa trên vài tiêu chí. Kết quả là một danh sách đã sắp xếp của các trang web sao cho những trang web ở trên đầu có khả năng là trang web người dùng mong muốn cao nhất. Danh sách mà mô-đun xếp hạng sắp xếp lại lọc ra những trang ít liên quan xuống dưới cùng, làm cho kết quả trả về thân thiện hơn với người dùng. Hình thức xếp hạng mới này có được sức phân hoá mạnh là nhờ kết hợp hai hệ thống đánh giá điểm số, content score (điểm số về nội dung) và popularity score (điểm số về độ phổ biến). Nhiều quy tắc được sử dụng để chấm điểm cho từng trang web. Popularity score được tính nhờ việc phân tích cấu trúc của hệ thống hyperlink trên Web và cũng là trọng tâm của đồ án này.

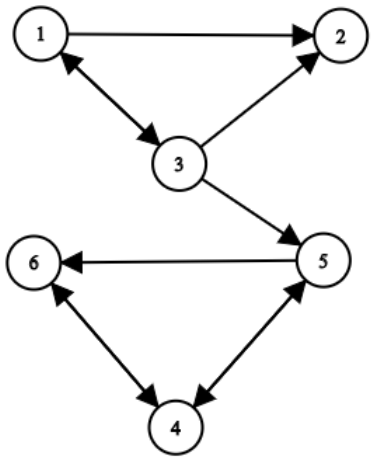
1.4 Xếp hạng trang web dựa trên độ phổ biến

Năm 1998, thuật toán tìm kiếm HITS, viết tắt của Hypertext Induced Topic Search, được nhà khoa học Jon Kleinberg công bố. Sau đó vài tháng PageRank được Sergey Brin và Larry Page giới thiệu trước công chúng. Hai mô hình có nhiều điểm tương đồng với nhau và PageRank đã được phát triển thành gã khổng lồ Google sau này.

Để hiểu được nguyên lý hoạt động của PageRank và HITS, trước tiên ta phải định nghĩa Web dưới dạng đồ thị. Hệ thống hyperlink của Web làm thành một đồ thị có hướng, gọi là đồ thị Web (Web graph). Những đỉnh trên đồ thị đại diện cho các trang web và các cạnh có hướng đại diện cho các đường dẫn. Theo vậy, khi dẫn đến một trang, gọi là inlink thì hướng của cạnh trỏ vào trang đó, còn outlink thì thì trỏ ra khỏi trang chứa đường dẫn. Đồ thị sau minh hoạ một tập tài liệu nhỏ bao bồm 6 trang web được liên kết với nhau bằng các hyperlink.

Các mô hình PageRank và HITS coi một hyperlink như một lá phiếu giữa hai trang web. Một đường hyperlink từ trang web A đến trang web B là một lá phiếu hoặc giới thiệu của A cho B. Vì vậy mà một trang web có nhiều lá phiếu (cùng nghĩa với có nhiều inlink) thì trang web đó phải là một trang web quan trọng hơn các trang web chỉ có vài inlink. Tuy nhiên, giống như các hình thức giới thiệu, đề cử khác như trích dẫn sách tham khảo hoặc thư giới thiệu, uy tín của bên giới thiệu, đề cử cũng rất quan trọng. Càng cho đi nhiều phiếu thì mức độ uy tín của từng lá phiếu lại giảm đi.

Dựa trên lập luận này, Brin và Page viết thuật toán PageRank của Google tính ra popularity score cho các trang web. Đây là thuật toán rất nổi tiếng và mặc dù công cụ tìm kiếm của Google còn có đóng góp của nhiều thuật toán khác nhưng PageRank luôn



Hình 1.5: Tập hợp 6 trang web và các đường dẫn giữa chúng

là trái tim của Google.

Phương pháp xếp hạng các trang của HITS tương đối giống với PageRank, nhưng nó sử dụng cả inlink và outlink để tạo ra 2 popularity score cho từng trang web. HITS định nghĩa các hub và authority. Một trang được coi là hub nếu nó chứa nhiều outlink, ngược lại authority là những trang chưa nhiều inlink. Dương nhiên một trang có thể vừa là hub, vừa là authority. Vì vậy mà HITS vừa chấm điểm hub vừa chấm điểm authority cho từng trang. Giống như ý tưởng của Google, HITS cũng hoạt động theo ý tưởng: *một trang được coi là hub tốt (tức có điểm hub cao) nếu nó dẫn đến các authority tốt, và một trang là authority tốt nếu nó được dẫn đến bởi các hub tốt.*

Trong chương này em đã trình bày về quá trình phát triển của các phương pháp thu thập thông tin, các phương pháp thu thập thông tin cổ điển và giới thiệu về thuật toán PageRank. Trong chương tiếp theo em sẽ trình bày trọng tâm của đồ án, đó là cơ sở lý thuyết, ý tưởng và cách thức cài đặt của thuật toán PageRank.

Chương 2

Thuật toán PageRank

Trong phân này em sẽ trình bày về các vấn đề liên quan đến thuật toán PageRank. Nội dung trình bày bao gồm:

- Cơ sở lý thuyết
- Ý tưởng thuật toán
- Cài đặt
- Ưu điểm thuật toán

Các nội dung trình bày được tham khảo từ nguồn [1], [4], [5], [6]

2.1 Cơ sở lý thuyết

2.1.1 Đại số tuyến tính

Định nghĩa giá trị riêng và vector riêng của ma trận

Cho ma trận vuông $A \in \mathbb{K}^{n\times n}$, với $\mathbb{K} = \mathbb{R}; \mathbb{C}$. Số $\lambda \in \mathbb{K}$ được gọi là giá trị riêng của A nếu tồn tại vector $x \neq 0; x \in \mathbb{K}^n$ sao cho $Ax = \lambda x$. Khi đó vector x được gọi là vector riêng của ma trận A ứng với giá trị riêng λ. Ký hiệu $\sigma(A)$ là tập tất cả các trị riêng của ma trận A

Tính chất của giá trị riêng và vector riêng

1. Các giá trị riêng của $A_{n\times n}$ là nghiệm của phương trình đa thức đặc trưng $p(\lambda) = det(A - \lambda I)$. Bậc của $p(\lambda)$ bằng n vậy nên A có n trị riêng, tuy nhiên một số có thể là số phức (ngay cả khi ma trận A là ma trận số thực), hoặc có thể là bội. Nếu ma trận A chỉ chứa số thực thì các trị riêng phức của nó (nếu có) phải theo cặp số phức liên hợp, tức là nếu $\lambda \in \sigma(A)$ thì $\bar{\lambda} \in \sigma(A)$.
2. Nếu như ma trận A có trị riêng $\lambda = 0$ thì ma trận A là không khả nghịch. Ngược lại, nếu mọi giá trị riêng của A đều khác không thì A khả nghịch.
3. Nếu λ là trị riêng của A thì λ^k là trị riêng của A^k
4. Nếu như λ là nghiệm bội k của phương trình đặc trưng thì λ là giá trị riêng bội k của ma trận A. Nếu λ là bội 1 thì λ là giá trị riêng đơn.
5. Nếu như λ là giá trị riêng bội k thì tồn tại k vector riêng độc lập tuyến tính tương ứng với λ.

Chứng minh

Định nghĩa giá trị riêng và vector riêng trội

Cho A là ma trận vuông cấp n. Giả sử A có đủ n trị riêng $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ thực hoặc phức sao cho

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq ... \geq |\lambda_n|$$

Các vector riêng tương ứng lần lượt là $X_1, X_2, X_3, ..., X_n$. Khi đó λ_1 được gọi là giá trị riêng trội của ma trận A và vector riêng X_1 ứng với λ_1 được gọi là vector riêng trội.

Chú ý: Theo định nghĩa trên thì trị riêng trội λ_1 phải là một số thực, vì nếu $\lambda_1 \in \mathbb{C}$ thì $\bar{\lambda}_1$ cũng là trị riêng có mô-đun bằng λ_1. Điều này trái với giả thiết $|\lambda_1| \geq |\lambda_2|$

Phương pháp luỹ thừa

Phương pháp luỹ thừa (power method) là một thuật toán lặp dùng để tìm ra trị riêng trội và vector riêng trội. Thuật toán bắt đầu bằng việc chọn một vector x_0 ban đầu nào đó, x_0 có thể là ngẫu nhiên hoặC là một xấp xỉ của vector riêng trội X_1. Giả sử các vector

riêng của A là một hệ độc lập tuyến tính. Khi đó các vector riêng của A tạo thành một cơ sở trong $\mathbb{R}^n$. Do đó x_0 có thể biểu diễn thành:

$$x_0 = \sum_{i=1}^{n} a_i X_i$$

Sau đó, ta tính dãy:

$$x_1 = Ax_0 = A\sum_{i=1}^{n} a_i X_i = \sum_{i=1}^{n} a_i A X_i = \sum_{i=1}^{n} a_i \lambda_i X_i$$

(Vì $AX_i = \lambda X_i$)

$$x_2 = Ax_1 = A^2 x_0 = A\sum_{i=1}^{n} a_i \lambda_i X_i = \sum_{i=1}^{n} a_i \lambda_i^2 X_i$$

$$\cdots\cdots$$

$$x_k = Ax_{k-1} = A^k x_0 = \sum_{i=1}^{n} a_i \lambda_i^k X_i = \lambda_1^k \left[a_1 X_1 + \sum_{i=2}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^k a_i X_i \right]$$

Theo giả thiết trị riêng trội $|\lambda_1| > |\lambda_i|, \forall i = 2, ..., n$. Suy ra khi $k \longrightarrow \infty$ thì $\left(\frac{\lambda_i}{\lambda_1}\right)^k \longrightarrow 0, \forall i = 2, ..., n$. Do đó ta có

$$\frac{A^k x_0}{\lambda_1^k} = a_1 X_1, \text{ khi } k \longrightarrow \infty$$

hay khi k đủ lớn thì

$$A^k x_0 \approx \lambda_1^k a_1 X_1$$

đồng thời

$$A^{k+1} x_0 \approx \lambda_1^{k+1} a_1 X_1$$
$$\Rightarrow A^{k+1} x_0 \approx \lambda_1 A^k x_0$$

hay

$$A(A^k x_0) \approx \lambda_1 (A^k x_0)$$

Vậy $A^k x_0 = x^k$ là vector riêng ứng với trị riêng λ_1 và λ_1 được tính theo tỷ số $\lambda_1 \approx \frac{(Ax_k).x_k}{x_k.x_k}$

Vậy, ta thấy các vector x_k có xu hướng hội tụ về một vector riêng nào đó trong số các vector riêng ứng với trị riêng trội. Để đảm bảo x_k hội tụ tới một vector duy nhất, ta thêm bước chuẩn hoá $x_k = \frac{x_k}{||x_k||}$ với $||.||$ là một chuẩn nào đó trong không gian vector. Ngoài để cho quá trình hội tụ đến một vector duy nhất, việc chuẩn hoá còn để tránh bị tràn số.

Tốc độ hội tụ: Ta xét phương pháp luỹ thừa cho hai ví dụ sau

Ví dụ 1: $A = \begin{bmatrix} 4 & 5 \\ 6 & 5 \end{bmatrix}$ có hai trị riêng là $\lambda_1 = 10$ và $\lambda_2 = -1$. Vì vậy nên tỷ số $\frac{|\lambda_2|}{|\lambda_1|} = 0.1$. Đối với ma trận này chỉ cần 4 vòng lặp đã thu được kết quả chính xác đến 3 chữ số sau dấu thập phân

$$x_0 = \begin{pmatrix} 1.000 \\ 1.000 \end{pmatrix}, x_1 = \begin{pmatrix} 0.818 \\ 1.000 \end{pmatrix}, x_2 = \begin{pmatrix} 0.835 \\ 1.000 \end{pmatrix}, x_3 = \begin{pmatrix} 0.833 \\ 1.000 \end{pmatrix}, x_4 = \begin{pmatrix} 0.833 \\ 1.000 \end{pmatrix}$$

Ví dụ 2: $B = \begin{bmatrix} -4 & 10 \\ 7 & 5 \end{bmatrix}$ có hai trị riêng là $\lambda_1 = 10$ và $\lambda_2 = -9$. Đối với ma trận này tỷ số $\frac{|\lambda_2|}{|\lambda_1|} = 0.9$ và phương pháp luỹ thừa phải mất đến 68 vòng lặp mới đưa ra được kết quả với độ chính xác đến 3 chữ số sau dấu thập phân

$$x_0 = \begin{pmatrix} 1.000 \\ 1.000 \end{pmatrix}, x_1 = \begin{pmatrix} 0.500 \\ 1.000 \end{pmatrix}, x_2 = \begin{pmatrix} 0.941 \\ 1.000 \end{pmatrix}, \ldots\ldots$$
$$\ldots\ldots, x_{66} = \begin{pmatrix} 0.715 \\ 1.000 \end{pmatrix}, x_{67} = \begin{pmatrix} 0.714 \\ 1.000 \end{pmatrix}, x_{68} = \begin{pmatrix} 0.714 \\ 1.000 \end{pmatrix}$$

Dựa vào hai ví dụ trên cũng như chứng minh sự hội tụ của phương pháp luỹ thừa, ta nhận thấy tốc độ hội tụ của phương pháp phụ thuộc vào tỷ số $\frac{|\lambda_2|}{|\lambda_1|}$. Cụ thể hơn, tốc độ hội tụ của thuật toán tương đương với tốc độ tỷ số $\left[\frac{|\lambda_2|}{|\lambda_1|}\right]^k$ hội tụ về 0 khi $k \longrightarrow \infty$

2.1.2 Xích Markov

Kết quả cuối cùng của thuật toán PageRank bản chất là phân phối dừng của một xích Markov. Vì vậy để hiểu rõ PageRank, ta cần nắm được một số kiến thức về xích Markov, bắt đầu bằng các khái niệm.

Định nghĩa: Ma trận xác suất là một ma trận vuông không âm $P_{n\times n}$ trong đó tổng các phần tử trên mỗi hàng đều bằng 1.

Định nghĩa: Quá trình ngẫu nhiên là một tập hợp các biến ngẫu nhiên $\{X_t\}_{t=0}^{\infty}$ có cùng một tập giá trị $\{S_1, S_2, ..., S_n\}$ được gọi là không gian trạng thái của quá trình. Biến

t thường được coi là biến thời gian, và X_t là trạng thái của quá trình tại thời điểm t. Ví dụ, ta xét một quá trình là sự thay đổi của thời tiết qua các ngày. Không gian trạng thái là các tình thái của thời tiết như nắng, mưa, lạnh, nhiều mây và biến X_t là trạng thái của thời tiết tại ngày thứ t.

- Theo như định nghĩa trên thì biến thời gian là biến rời rạc, đồng thời không gian trạng thái là hữu hạn.

Định nghĩa: Xích Markov là một quá trình ngẫu nhiên mà thoả mãn tính chất Markov

$$P(X_{t+1} = S_j | X_t = S_{i_t}, X_{t-1} = S_{i_{t-1}}, ..., X_0 = S_{i_0}) = P(X_{t+1} = S_j | X_t = S_{i_t})$$

Điều kiện này thể hiện tính chất *không nhớ* của xích Markov, có nghĩa là trạng thái của xích tại thời điểm tiếp theo chỉ phụ thuộc vào trạng thái ở thời điểm hiện tại mà không quan tâm đến các trạng thái trước. Ví dụ như, quá trình biến đổi của thời tiết là một xích Markov nếu như trạng thái thời tiết ngày mai không phụ thuộc vào trạng thái thời tiết của các ngày trước đây, mà chỉ phụ thuộc vào trạng thái của ngày hôm nay.

Định nghĩa: Một vector phân phối xác suất (gọi tắt là vector xác suất) là một vector không âm $\lambda^T = (\lambda_1, \lambda_2, ..., \lambda_n)$ sao cho $\sum_k \lambda_k = 1$. Nếu ta đặt

$$\lambda_i = P(X_t = i)$$

thì $\lambda = (\lambda_i)$ là một phân phối. Ta nhận thấy tại mỗi thời điểm t, xích Markov có một vector phân phối xác suất. Ta gọi $\lambda = P(X_0 = i)$ là vector phân phối ban đầu.

Định nghĩa: xác suất chuyển $p_{ij} = P(X_j = S_j | X_{t-1} = S_i)$ là xác suất ở trạng thái S_j tại thời điểm t với điều kiện xích ở trạng thái S_i tại thời điểm $t-1$, hay nói cách khác là xác suất chuyển từ trạng thái S_i sang trạng thái S_j tại thời điểm t bất kỳ.

Định nghĩa: Ma trận xác suất chuyển $P_{n\times n(t)} = [p_{ij}(t)]$ là ma trận không âm, mỗi hàng là xác suất từ trạng thái i đi đến tất cả các trạng thái còn lại, vì vậy mà tổng các phần tử mỗi hàng bằng 1. Hay nói cấch khác, $P(t)$ là ma trận xác suất cho mỗi t.

Với định nghĩa như trên, ta thấy với mỗi xích Markov xác định một ma trận xác suất chuyển duy nhất và điều ngược lại cũng đúng, mỗi một ma trận xác suất $P_{n\times n}$ xác định một xích Markov n trạng thái

Định lý: Giả sử X_t là một xích Markov với vector phân phối xác suất ban đầu λ, khi đó ta có

- $P(X_n = j) = (\lambda P^n)_j$
- $P_i(X_n = j) = P(X_{n+m} = j | X_m = j) = p_{ij}^{(n)}$

theo định lý này, ta thấy $p^(n)_{ij}$ là xác suất chuyển từ trạng thái i sang trạng thái j sau n bước.

Định nghĩa: Ta nói rằng trạng thái i đến được trạng thái j và ký hiệu là $i \longrightarrow j$ nếu tồn tại $n \geq 0$ sao cho $p_{ij}^{(n)} > 0$. Hai trạng thái i và j được gọi là liên thông và ký hiệu là $i \longleftrightarrow j$ nếu $i \longrightarrow j$ và $j \longrightarrow i$

Định nghĩa: Một xích Markov được gọi là tối giản nếu hai trạng thái bất kỳ đều liên thông với nhau. Ma trận chuyển P được gọi là tối giản nếu xích Markov tương ứng với nó là tối giản

Định nghĩa: vector phân phối xác suất dừng của một chuỗi Markov có ma trận xác suất chuyển P là một vector xác suất π^T sao cho $\pi^T P = \pi^T$. Nếu như P là ma trận tối giản thì P có phân phối dừng.

Định nghĩa: vector phân phối xác suất bậc k của một xích Markov là

$$\lambda^T(k) = (\lambda_1(k), \lambda_2(k), ..., \lambda_n(k)), \quad \text{trong đó} \quad \lambda_j(k) = P(X_k = S_j)$$

hay nói cách khác, $\lambda_j(k)$ là xác suất xích ở trạng thái thứ j sau k lần chuyển trạng thái.

Quá trình lướt web của người dùng bằng việc chọn các đường dẫn để di chuyển qua các trang web có thể được coi là một quá trình ngẫu nhiên. Không gian trạng thái sẽ là tập hợp toàn bộ các trang web. Thuật toán PageRank coi quá trình này là một xích Markov, Với điều kiện là trang web tiếp theo người dùng chọn không phụ thuộc vào các trang đã đi qua, mà chỉ phụ thuộc vào trang hiện tại. Tức là, người dùng chọn bất kì một đường dẫn tại trang web hiện tại để đến trang tiếp theo, quá trình này gọi là *random walk* trên đồ thị hyperlink.

Cho một phân phối ban đầu $\lambda^T(0) = (\lambda_1(0), \lambda_2(0), ..., \lambda_n(0))$, câu hỏi đặt ra là, xác suất để xích ở một trạng thái là bao nhiêu sau 1 lần chuyển trạng thái. Sử dụng các quy luật xác suất, ta có.

$$\lambda_j(1) = P(X_1 = S_j) = \sum_{i=1}^{n} P\left[X_0 = S_i\right].P\left[X_1 = S_j | X_0 = S_i\right] = \sum_{i=1}^{n} \lambda_i(0) p_{ij}$$

Tổng hợp lại cho toàn bộ các thành phần của vector phân phối xác suất, ta có $\lambda^T(1) = \lambda^T(0)P$. Phương trình nảy thể hiện sự biến đổi của phân phối xác suất ban đầu sau 1 bước chuyển trạng thái. Nhờ tính chất "không nhớ" của chuỗi Markov mà ở cuối bước chuyển thứ nhất, có thể coi xích bắt đầu lại từ đầu và lấy $\lambda^T(1)$ làm phân phối xác suất ban đầu mới. Vì vậy, ta có $\lambda^T(2) = \lambda^T(1)P$, và $\lambda^T(3) = \lambda^T(2)P$,...và cuối cùng ta có

$$\lambda^T(k) = \lambda^T(0)P^k$$

Tới đây, ta nhận thấy biểu thức trên chính là công thức của phương pháp luỹ thừa với vector ban đầu là $\lambda^T(0)$. Và nếu như $\lambda^T(k)$ hội tụ khi k tiến tới vô cùng thì nó sẽ hội tụ đến vector phân phối dừng của xích Markov X_t.

2.2 Ý tưởng của thuật toán PageRank

Brin và Page, những người sáng tạo ra PageRank, mô hình hoá bài toán với một phương trình đơn giản, bắt nguồn từ các công cụ đo lường khoa học (bibliometric), các phân tích về trích dẫn giữa các bài báo khoa học. PageRank của một trang P_i, ký hiệu $r(P_i)$, là tổng của PageRank của tất cả các trang trỏ đến P_i

$$r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|} \tag{2.2.1}$$

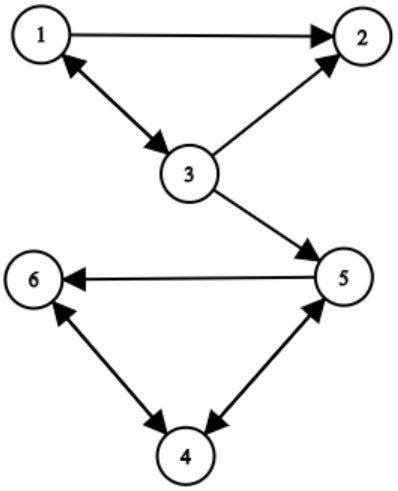
trong đó B_{P_i} là tập hợp các trang trỏ đến P_i và $|P_j|$ là số outlink đi từ P_j. Ta có thể thấy PageRank của P_i phụ thuộc vào số outlink của P_j, ký kiệu $|P_j|$ trong phương trình (2.1.1), điều này là phù hợp với lập luận của Brin và Page về popularity score. Vấn đề với phương trình (2.1.1) đó là giá trị $r(P_j)$, PageRank của các trang P_j là chưa biết. Brin và Page giải quyết vấn đề này bằng phương pháp lặp. Họ giả sử rằng ban đầu tất cả các trang có PageRank bằng nhau (cụ thể hơn là $1/n$, trong đó n là số trang trong danh mục Web của Google). Sau đó các giá trị đó được dùng trong phương trình (2.2.1) để tính $r(P_i)$ mới cho từng trang P_i trong danh mục. Sau khi tính được các $r(P_i)$ mới, chúng lại được thay vào (2.2.1) ở vị trí của $r(P_j)$ trong vòng lặp tiếp theo. Ký hiệu $r_{k+1}(P_i)$ là PageRank của trang P_i tại vòng lặp thứ $k+1$, khi đó,

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|} \tag{2.2.2}$$

Trong đó $r_0(P_i) = 1/n$ với mọi trang và quá trình này được lặp lại liên tục với mong muốn rằng từ một thời điểm nào đó trở đi, các giá trị PageRank sẽ hội tụ đến các giá trị cố định. Áp dụng phương trình (2.2.2) cho một mô hình Web nhỏ ta tính được các giá trị PageRank sau một vài vòng lặp.

Vòng lặp 0	Vòng lặp 1	Vòng lặp 2	Thứ hạng ở vòng lặp 2
$r_0(P_1) = 1/6$	$r_1(P_1) = 1/18$	$r_2(P_1) = 1/36$	5
$r_0(P_2) = 1/6$	$r_1(P_2) = 5/36$	$r_2(P_2) = 1/18$	4
$r_0(P_3) = 1/6$	$r_1(P_3) = 1/12$	$r_2(P_3) = 1/36$	5
$r_0(P_4) = 1/6$	$r_1(P_4) = 1/4$	$r_2(P_4) = 17/72$	1
$r_0(P_5) = 1/6$	$r_1(P_5) = 5/36$	$r_2(P_5) = 11/72$	3
$r_0(P_6) = 1/6$	$r_1(P_6) = 1/6$	$r_2(P_6) = 14/72$	2

Các phương trình (2.1.1) và (2.1.2) tính PageRank cho từng trang một, việc này có thể khá cồng kềnh về mặt hình thức. Thay vào đó, ta sử dụng ma trận để biểu diễn toàn bộ các $\frac{1}{|P_j|}$ và tại mỗi vòng lặp tính ra một vector chứa toàn bộ các PageRank của các trang có trong danh mục, gọi là vector PageRank. Ta gọi H là ma trận $n \times n$ và vector hàng π^T, trong đó H đã được chuẩn hoá theo hàng và $H_{ij} = 1/|P_i|$ nếu như có một đường dẫn từ trang i đến trang j, và bằng 0 nếu ngược lại. Mặc dù H có điểm giống với ma trận liền kề của đồ thị hyperlink, xong các hàng của nó lại có tính chất xác suất. Trong ví dụ trên, ma trận H sẽ là



Hình 2.1: Đồ thị có hướng biểu diễn một mô hình Web nhỏ

$$H = \begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_4 \\ P_6 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

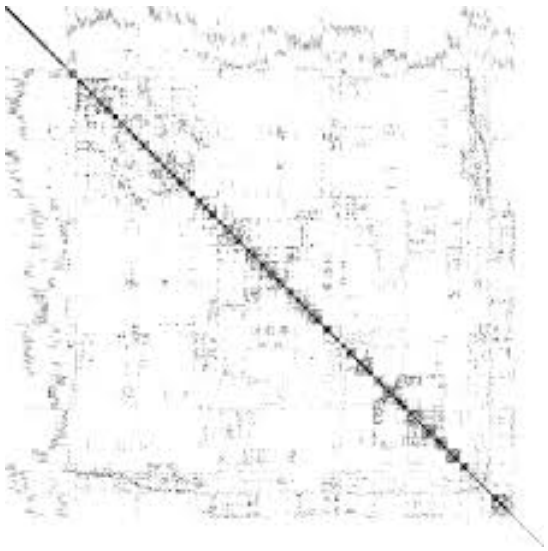
Các phần tử khác 0 ở dòng i tương ứng với các outlink của trang thứ i, trong khi đó, các phần tử khác không của cột i tương ứng với inlink của trang thứ i. Tiếp theo, ta định nghĩa vector $\pi^{(k)T}$ là vector PageRank ở vòng lặp thứ k. Khi đó, phương trình (4.1.2) có thể viết lại thành

$$\pi^{(k+1)T} = \pi^{(k)T} H \tag{2.2.3}$$

Khi đó, ta có các nhận xét sau.

1. Mỗi vòng lặp của phương trình (2.2.3) sử dụng một phép nhân ma trận với vector, có độ phức tạp tính toán vào khoảng $O(n^2)$, với n là kích thước ma trận H.

2. Ma trận H là ma trận thưa với đa số phần tử là bằng 0, vì mỗi trang web thì chỉ có đường dẫn đến một vài trang web khác. Ma trận thưa có nhiều lợi thế. Thứ nhất, chúng đòi hỏi ít không gian bộ nhớ, vì ta chỉ cần lưu các phần tử khác 0 và vị trí của chúng trong ma trận. Thứ hai, phép tích vector với ma trận đối với ma trận thưa có độ phức tạp ít hơn $O(n^2)$ đối với ma trận đầy. Chính xác độ phức tạp là $O(nnz(H))$ trong đó $nnz(H)$ là số



Hình 2.2: Ví dụ về ma trận thưa, trong đó các phần tử khác không là các chấm đen

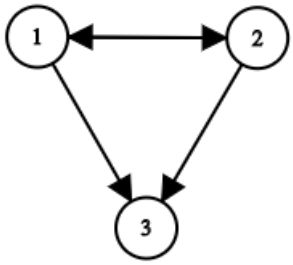
phần tử khác không trong H. Theo thống kê thì một trang web có khoảng 10 outlink, tức là H có khoảng $10n$ phần tử khác không, so với n^2 ở trong ma trận đầy. Điều này có nghĩa là phép nhân vector ma trận ở phương trình (2.2.3) có độ phức tạp chỉ còn $O(n)$.

3. Công thức truy hồi (2.2.3) chính là công thức của phương pháp luỹ thừa
4. Ma trận H trông giống ma trận xác suất chuyển của một xích Markov

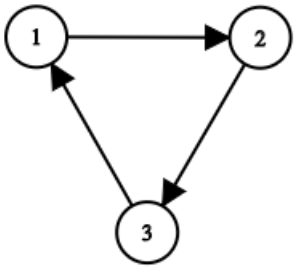
Như vậy, ta thấy nếu như theo cơ sở lý thuyết của phương pháp luỹ thừa thì công thức (2.2.3) sẽ hội tụ về một vector riêng của ma trận H. Tuy nhiên quá trình lặp này gặp phải một số trở ngại sau.

Brin và Page ban đầu đã sử dụng vector ban đầu $\pi^{(0)T} = 1/ne^T$, trong đó e^T là vector hàng chứa các số 1. Họ ngay lập tức gặp phải vài vấn đề khi sử dụng vector ban đầu này với công thức (2.2.3). Trong đó có vấn đề **rank sink**, là khi có những trang tích luỹ dần các PageRank sau mỗi vòng lặp, chiếm giữ và không chịu chia sẻ PageRank cho các trang khác. Trong đồ thị dưới đây, đỉnh treo 3 là một rank sink. Trong đồ thị 2.2.1, cụm các đỉnh 4, 5, 6 cùng nhau độc chiếm PageRank. Chỉ sau 13 vòng lặp, $\pi^{(13)T} = (0 \quad 0 \quad 0 \quad 2/3 \quad 1/3 \quad 1/5)$. Việc độc chiếm này có thể là do vô tình hoặc có ý đồ. Trong ví dụ của $\pi^{(13)T}$ còn bộc lộ ra một vấn đề nữa, đó khi các trang chiếm hết PageRank, sẽ có các trang có PageRank bằng 0. Khi đó việc xếp hạng các trang theo giá trị PageRank trở nên phức tạp vì phần lớn các trang có PageRank bằng 0. Ta mong muốn tất cả các PageRank là dương.

Ngoài ra còn có các vấn đề gây ra bởi các chu trình. Ta quan sát đồ thị (2.2.4) chỉ có 3 đỉnh, trong đó đỉnh 1 chỉ trỏ tới đỉnh 2 và đỉnh 2 chỉ trỏ đến đỉnh 3 và đỉnh 3 lại trỏ về đỉnh 1 tạo thành một đồ thị vòng. Nếu ta áp dụng công thức (2.2.3) với vector ban đầu $\pi^{(0)T} = (1 \quad 0)$ thì quá trình lặp sẽ diễn ra mãi mãi mà không bao giờ hội tụ.



Hình 2.3: Đồ thị với rank sink



Hình 2.4: Đồ thị với chu trình

2.3 Cài đặt thuật toán PageRank

Ở trên ta đã đưa ra nhận xét rằng ma trận H trông giống một ma trận xác suất của một xích Markov. Và trong phần cơ sở lý thuyết ta cũng đã chỉ ra rằng áp dụng phương pháp luỹ thừa cho ma trận xác suất chuyển của một xích Markov sẽ hội tụ đến một vector đặc biệt, được gọi là vector phân phối xác suất dừng. Vì vậy,Brin và Page đã chỉnh sửa H trở thành một ma trận xác suất của một chuỗi Markov để giải quyết các vấn đề gây ra bởi rank sink và chu trình. Tuy nhiên trong bài báo năm 1998, họ lại không đề cập gì đến các khái niệm về xích Markov.

Thay vì sử dụng xích Markov để lý giải cho cách làm của mình, Brin và Page sử dụng khái niệm **"random surfer"**. Random surfer là một người lướt web, truy cập ngẫu nhiên vào các trang trên hệ thống hyperlink của Web, tức là khi người đó đang ở một trang có vài outlink, họ sẽ chọn một đường outlink ngẫu nhiên để đi đến một trang mới. Quá trình này tiếp diễn liên tục và khi thời gian đủ lớn, khoảng thời gian mà random surfer ở một trang có thể coi làm đánh giá mức độ quan trọng của trang đó. Nếu người đó dành nhiều thời gian ở một trang web nào đó, thì tức là khi lướt web ngẫu nhiên người này liên tục quay trở lại trang web đó. Các trang mà người này thường xuyên truy cập phải là những trang web quan trọng vì chúng được trỏ tới bởi

các trang web quan trọng khác. Tuy nhiên, quá trình random surfer sẽ gặp phải nhiều trở ngại, như khi họ đi vào các đỉnh treo trên đồ thị hyperlink. Và trên Web tồn tại rất nhiều đỉnh treo như các tệp, các ảnh, v.v,... Để giải quyết vấn đề này, Brin và Page đã đưa ra chỉnh sửa đầu tiên cho ma trận H. Họ thay thế các hàng 0^T trong ma trận H bằng các hàng $1/ne^T$, biến H trở thành một ma trận xác suất. Khi đó, nếu random surfer có đi vào một đỉnh treo thì họ có thể nhảy tới ngẫu nhiên bất kỳ trang nào khác. Đối với đồ thị 2.1 thì ma trận xác suất ký hiệu S là

$$S = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Nếu như ta đặt $a = (a_i)$ trong đó $a_i = 1$ nếu như trang web i là một đỉnh treo và bằng 0 nếu như ngược lại thì khi đó, $S = H + a(1/ne^T)$. Vector a được gọi là **vector đỉnh treo** và ma trận $1/nae^T$ có hạng bằng 1.

Sự điều chỉnh này đảm bảo S là một ma trận xác suất, và cũng là ma trận xác suất chuyển cho một xích Markov. Tuy nhiên, điều này thôi chưa đảm bảo được quá trình lặp sẽ hội tụ đến kết quả mong muốn (đó là vector π^T tồn tại duy nhất và quá trình lặp sẽ hội tụ nhanh đến vector này). Theo như cơ sở lý thuyết về xích Markov, thì nếu ma trận xác suất P là tối giản thì tồn tại vector phân phối dừng, mà trong trường hợp này chính là vector PageRank. Brin và Page tiếp tục sử dụng khái niệm random surfer để lý giải cho cách làm của mình.

Bây giờ ngoài việc lướt ngẫu nhiên qua các trang web dọc theo các đường Hyperlink, đôi lúc, người lướt web sẽ cảm thấy chán và bỏ trang web hiện tại và nhảy cóc sang một trang web khác bất kỳ và bắt đầu lướt ngẫu nhiên từ trang web mới, cho tới khi lại cảm thấy chán và nhảy tiếp đến các trang web khác. Khi đưa ý tưởng này vào mô hình của mình, Brin và Page đã đưa ra một ma trận mới, đặt là G, sao cho

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

trong đó α là một số vô hướng trong khoảng 0 và 1. **G** lúc này được gọi mà **ma trận Google**. Trong mô hình này, α là tham số quyết định khoảng thời gian mà random surfer sẽ lướt web bằng các đường Hyperlink thay vì nhảy cóc đến các trang web khác. Giả sử $\alpha = 0.6$, khi đó 60% thời gian người lướt web sẽ đi theo các hyperlink và 40% thời gian họ sẽ thực hiện nhảy cóc. Việc nhảy cóc là hoàn toàn ngẫu nhiên vì ma trận xác suất $E = 1/nee^T$ có phân phối đều, tức là người lướt web có thể nhảy tới một trang web bất kỳ với xác suất như nhau.

Ma trận **G** có một vài tính chất sau.

- **G** là ma trận xác suất. Nó là tổ hợp lồi của hai ma trận xác suất S và $E = 1/nee^T$.
- **G** là ma trận là tối giản. Vì mọi trang lúc này đều có thể đi đến được nhau thông qua việc nhảy cóc nên điều này là hiển nhiên.

- **G** là ma trận đầy đủ, khiến cho việc sử dụng nó khá tốn kém về mặt tính toán. May thay ta có thể biểu diễn **G** theo H là một ma trận thưa khiến cho việc tính toán dễ dàng hơn rất nhiều

$$\begin{aligned} G &= \alpha S = (1-\alpha)1/nee^T \\ &= \alpha(H + 1/nae^T) + (1-\alpha)1/nee^T \\ &= \alpha H + (\alpha a + (1-\alpha)e)1/ne^T \end{aligned}$$

- **G** không có tính tự nhiên theo ý nghĩa là ma trận hyperlink H đã bị chỉnh sửa 2 lần để có thể đảm bảo quá trình lặp là hội tụ. Vector phân phối dừng (tức vector PageRank) không tồn tại với H nhưng có tồn tại với **G** và kết quả thực nghiệm cho thấy vector này rất tốt khi dùng để xếp hạng cho các trang web.

Tổng kết lại, thuật toán PageRank của Google là

$$\pi^{(k+1)T} = \pi^{(k)T}\mathbf{G} \tag{2.3.1}$$

và chính là phương pháp luỹ thừa áp dụng cho ma trận **G**.

Quay trở lại với ví dụ đầu tiên, với $\alpha = 0.9$, ma trận **G** khi đó sẽ là

$$\mathbf{G} = 0.9H + \left[0.9\begin{pmatrix}0\\1\\0\\0\\0\\0\end{pmatrix} + 0.1\begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix}\right].\frac{1}{6}\begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1\end{pmatrix}$$

$$= \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}$$

Và vector PageRank là vector phân phối dừng của ma trận **G** và bằng

$$\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$
$$\pi^T = \begin{pmatrix} 0.03721 & 0.05396 & 0.04151 & 0.3751 & 0.206 & 0.2862 \end{pmatrix}$$

Điều này tức là, với $\pi_1 = 0.03721$ có nghĩa là 3.721% thời gian người lướt web sẽ ở trang 1. Vì vậy, các trang trong mạng Hyperlink trên có thể xếp hạng theo thứ tự (4 6 5 2 3 1), tức là trang thứ 4 là trang quan trọng nhất và trang 1 là trang ít quan trọng nhất.

2.4 Ưu điểm của thuật toán PageRank

Phương pháp luỹ thừa là một trong những phương pháp lặp lâu đời và đơn giản nhất để tìm trị riêng trội và vector riêng trội của ma trận. Tuy nhiên phương pháp luỹ thừa lại có tốc độ hội tụ rất chậm. Các phương pháp khác như Gauss-Seidel, Jacobi, GMRES,v.v,... đều có tốc độ nhanh hơn phương pháp luỳ thừa. Thế nhưng Brin và Page vẫn lựa chọn phương pháp này vì những lý do sau đây.

Lý do đầu tiên đó là vì tính đơn giản của phương pháp luỹ thừa. Điều này khiến cho việc cài đặt và lập trình tương đối dễ dàng. Hơn nữa, phương pháp luỹ thừa cho ma trận đầy đủ $\mathbf{G}$ còn có thể biểu diễn qua ma trận H là ma trận rất thưa.

$$\begin{aligned}\pi^{(k+1)T} =&\pi^{(k)T}\mathbf{G}\\ =&\alpha\pi^{(k)T}S + \frac{1-\alpha}{n}\pi^{(k)T}ee^T\\ =&\alpha\pi^{(k)T}H + (\alpha\pi^{(k)T}a + 1 - \ \alpha)e^T/n\end{aligned} \tag{2.4.1}$$

Phép nhân $(\pi^{(k)T}H)$ được thực hiện trên ma trận rất thưa H, các ma trận S và H không cần được tính ra và lưu lại trên bộ nhớ máy tính, chỉ có các vector a và e là cần phải lưu lại trong quá trình tính toán. Và mỗi một phép nhân ma trận với vector có độ phức tạp là $O(n)$ vì ma trận H có trung bình khoảng 10 phần tử khác không mỗi hàng. Đây có lẽ là lý do Brin và Page sử dụng phương pháp luỹ thừa mà không phải các phương pháp khác.

Thứ hai, phương pháp luỹ thừa không cần phải lưu lại ma trận hệ số và không thay đổi các giá trị trên ma trận ban đầu. Điều này đặc biệt có lợi khi ma trận có kích thước quá lớn và việc lưu trữ và biến đổi nó có thể tốn rất nhiều bộ nhớ và thời gian tính toán. Các phương pháp tính trực tiếp trị riêng bằng việc biến đổi từng phần tử không thể áp dụng cho ma trận H với số phần tử có thể lên đến hàng nghìn tỷ.

Thứ ba, phương pháp luỹ thừa không tốn kém về mặt bộ nhớ. Ngoài việc phải lưu trữ ma trận H và vector đỉnh treo a, chỉ cần ghi nhớ lại duy nhất một vector là $\pi^{(k)T}$. Đây là một vector hoàn toàn dày đặc, tức là cần phải lưu lại n số thực. Số lượng trang trong danh mục Web của Google là khoảng 8.1 tỷ nên vấn đề về bộ nhớ không hề đơn giản. Các phương pháp khác như GMRES hoặc BICGSTAB mặc dù nhanh hơn nhưng lại cần phải lưu trữ nhiều vector. Phương pháp GMRES(10) yêu cầu phải lưu trữ 10 vector độ dài n ở mỗi vòng lặp, tương đương với khối lượng dữ liệu của cả ma trận H.

Lý do cuối cùng để Brin và Page lựa phương pháp luỹ thừa là về số vòng lặp cần thực hiện. Theo Brin và Page thì chỉ cần khoảng 50-100 vòng lặp để cho quá trình lặp hội tụ và cho ta một xấp xỉ đủ tốt của vector PageRank. Nhắc lại là mỗi vòng lặp của phương pháp luỹ thừa có độ phức tạp $O(n)$ vì ma trận H rất thưa. Như vậy, độ phức tạp của cả quá trình là $50)(n)$, là độ phức tạp tuyến tính. Thuật toán có thời gian chạy và độ phức tạp tính toán tuyến tính so với kích thước bài toán là rất nhanh và hiếm.

Để lý giải tại sao phương pháp luỹ thừa áp dụng cho ma trận $\mathbf{G}$ chỉ cần khoảng 50 vòng lặp

để cho ra kết quả hội tụ, ta nhắc lại tốc độ hội tụ của phương pháp luỹ thừa phụ thuộc vào tỷ số giữa hai giá trị riêng có chuẩn lớn nhất, là λ_1 và λ_2. Cụ thể hơn thuật toán có tốc độ hội tụ tương đương với tốc độ tỷ số $\left[\frac{|\lambda_2|}{|\lambda_1|}\right]^k$ tiến về 0. Đối với các ma trận xác suất như $\mathbf{G}$ thì $|\lambda_1| = 1$, vì vậy tốc độ hội tụ phụ thuộc vào $|\lambda_2|$. Nói chung việc tìm trị riêng λ_2 khi đã biết λ_1 là hoàn toàn có thể, và từ đó ta có thể đánh giá được tốc độ hội tụ của phương pháp luỹ thừa. Tuy nhiên trong bài toán PageRank, ta có một định lý đặc biệt

Định lý: Nếu ma trận xác suất S có tập các trị riêng $\{1, \lambda_2, \lambda_3, ..., \lambda_n\}$ thì khi đó, ma trận $\mathbf{G} = \alpha S + (1-\alpha)ev^T$ có tập các trị riêng là $\{1, \alpha\lambda_2, \alpha\lambda_3, ..., \alpha\lambda_n\}$ trong đó v^T là một vector xác suất

Hơn nữa, trong thực tế, $|\mu_2| \approx 1$, tức là $|\lambda_2(G)| \approx \alpha$. Vì vậy mà tham số α lý giải tại sao kết quả hội tụ chỉ sao 50 vòng lặp. Trong bài báo của mình, Brin và Page đặt $\alpha = 0.85$, mà $\alpha^{50} = 0.85^{50} \approx 0.000296$, chỉ ra rằng ở vòng lặp thứ 50, kết quả có độ chính xác khoảng 3-4 chữ số sau dấu phẩy. Mức độ chính xác này là chưa đủ để phân biệt rõ ràng giữa các phần tử của vector PageRank, tuy nhiên chỉ cần kết hợp với content score là vừa đủ để xếp hạng các trang web.

Trong chương này, em đã trình bày về cơ sở lý thuyết, ý tưởng, phương thức cài đặt và ưu điểm của thuật Toán PageRank. Trong chương tiếp theo, em sẽ trình bày chương trình cài đặt và kết quả thực nghiệm trên bộ dữ liệu thực tế.

Chương 3

Xây dựng chương trình và ứng dụng

Trong chương này em sẽ trình bày chương trình cài đặt thuật toán PageRank bằng ngôn ngữ lập trình Matlab.

Sau đây là chương trình cài đặt thuật toán PageRank trên ngôn ngữ lập trình Matlab

```
function [pi,time,numiter]=PageRank(pi0,H,n,alpha,epsilon)
rowsumvector=ones(1,n)*H;
nonzerorows=find(rowsumvector);
zerorows=setdiff(1:n,nonzerorows); l=length(zerorows);
a=sparse(zerorows,ones(l,1),ones(l,1),n,1);
k=0;
residual=1;
pi=pi0;
tic;
while (residual >= epsilon)
prevpi=pi;
k=k+1;
pi=alpha*pi*H + (alpha*(pi*a)+1-alpha)*((1/n)*ones(1,n));
residual=norm(pi-prevpi,1);
end
numiter=k;
time=toc;
end
```

Trong đó các tham số đầu vào:

- pi0 là vector xấp xỉ ban đầu của vector PageRank
- H là ma trận hyperlink dưới dạng ma trận thưa
- n là số phần tử hay là số trang web trong H

- alpha là tham số α
- epsilon là sai số hội tụ tính theo chuẩn 1

Kết quả trả ra

- pi là vector PageRank của ma trận H
- time là thời gian cần để tính vector pi
- numiter là số vòng lặp

Ta sẽ kiểm định chương trình trên 3 bộ dữ liệu là 3 đồ thị Hyperlink. Mô tả của 3 bộ dữ liệu có trong bảng sau

Tên	Số lượng đỉnh	Số lượng cạnh	Nguồn
Harvard500	500	2636	Trang web 1
web-Stanford	281.903	2.312.497	website 2
web-Google	875.713	5.105.039	website 3

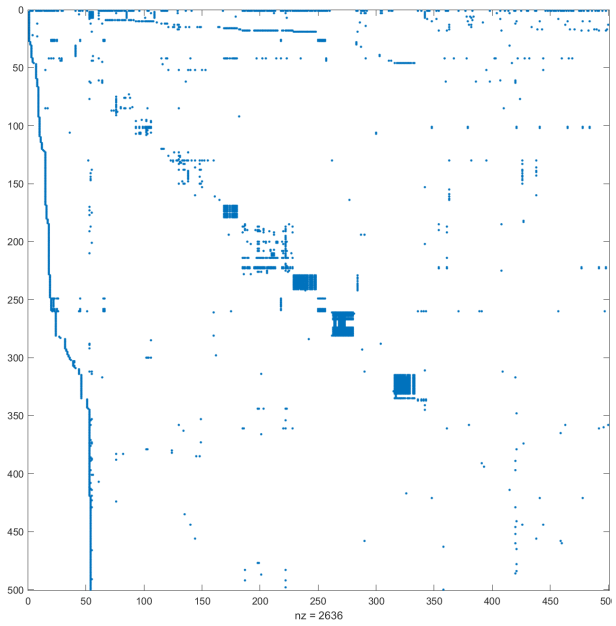
Bảng 3.1

Website 1: https://www.cise.ufl.edu/research/sparse/matrices/MathWorks/Harvard500.html

Website 2: https://snap.stanford.edu/data/web-Stanford.html

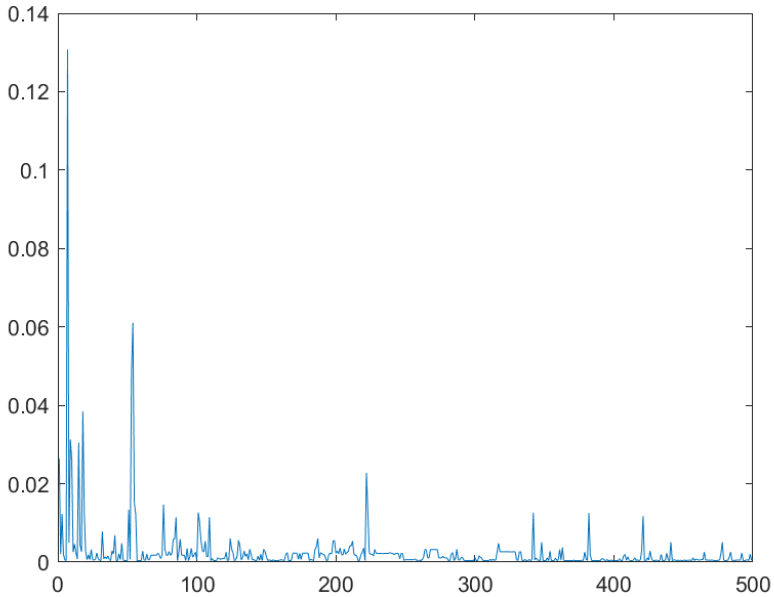
Website 3: https://snap.stanford.edu/data/web-Google.html

Đầu tiên là bộ dữ liệu Harvard500 bao gồm 500 đỉnh tương đương với 500 trang web nội bộ của đại học Harvard. Dữ liệu dầu vào là một ma trận thưa kích cỡ 500×500 có 2636 phần tử khác không ứng với 2636 cạnh. Sử dụng Matlab ta có thể minh hoạ độ thưa của ma trận trên



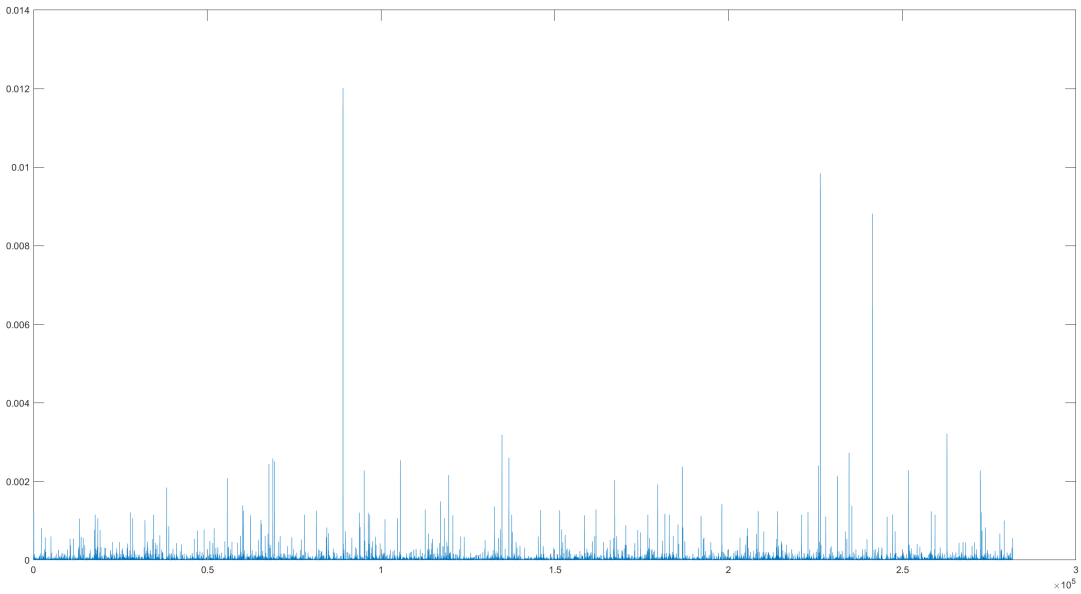
Hình 3.1: Bộ dữ liệu Harvard500

Với $alpha = 0.85, epsilon = 10^{-8}$, chạy chương trình cho ta vector PageRank. Thuật toán hội tụ trong 60 vòng lặp trong thời gian 4.56×10^{-4} giây.



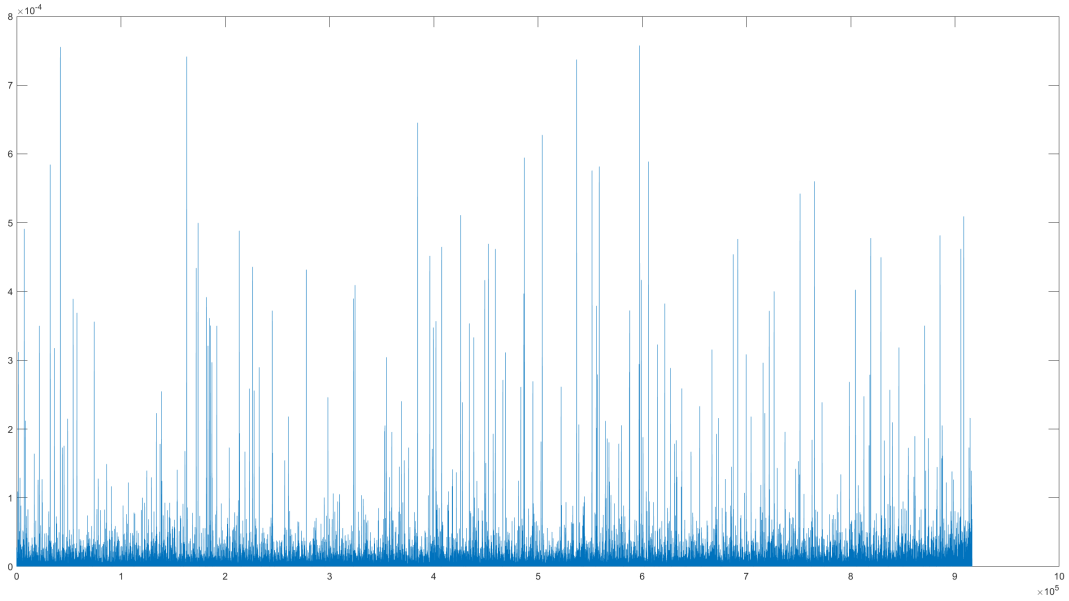
Hình 3.2: PageRank của bộ dữ liệu Harvard500

Tiếp theo là bộ dữ liệu web-Stanford bao gồm 281903 đỉnh và 2312497 cạnh. Bộ dữ liệu này là quá lớn để có thể lưu dưới dạng ma trận đầy. Với cùng các tham số trên, chạy chương trình ta thu được vector PageRank trong 76 vòng lặp và 1,3 giây



Hình 3.3: PageRank của bộ dữ liệu web-Stanford

Cuối cùng là bộ dữ liệu web-Google gồm có 875713 đỉnh và 5105039 cạnh. Bộ dữ liệu này do chính Google đưa ra vào năm 2002 trong một cuộc thi lập trình. Cũng như hai bộ dữ liệu trên, với các tham số không đổi, thuật toán thu được vector PageRank sau 74 vòng lặp và 4.95 giây



Hình 3.4: PageRank của bộ dữ liệu web-Google

Nhận xét: Ở hai bộ dữ liệu đầu tiên, vector PageRank chỉ có vài phần tử lớn trội hơn hẳn các phần tử còn lại. Điều này là do đây là các mạng của chỉ 1 tên miền đó là tên miền của trường đại học Harvard và Stanford. Trong các mạng này các trang web thông thường có đường dẫn đến trang chủ và một số ít các trang web quan trọng khác. Còn ở bộ dữ liệu của Google, các trang web thuộc về nhiều tên miền khác nhau, vì vậy nên thứ hạng các trang web cũng đồng đều hơn.

Chương 4

SEO - Search Engine Optimization

Trong chương 4 em sẽ trình bày về sức ảnh hưởng của Google đối với lĩnh vực SEO và các phương pháp SEO, cụ thể:

- Khái niệm về SEO
- Mối quan hệ giữa SEO và Google
- Một số phương pháp SEO phổ biến

Các nội dung được tham khảo nguồn: [1], [8], [9], [10], [11]

4.1 Mối quan hệ giữa Google và SEO

SEO - viết tắt của Search Engine Optimization - là quá trình nhằm làm nâng cao thứ hạng của trang web trên các công cụ tìm kiếm để giúp người dùng có thể tìm thấy trang web dễ dàng hơn trên bảng kết quả tìm kiếm. SEO được coi là một chiến lược marketing, giúp cho các doanh nghiệp có thể tiếp cận khách hàng dễ hơn. Khi thứ hạng của trang web trên kết quả tìm kiếm tăng, thì số lượng người đến trang web đó cũng tăng và lượng khác hàng tiềm năng cũng tăng theo. Là công cụ tìm kiếm phổ biến nhất hiện nay, cách thức Google xếp hạng các trang web có ảnh hưởng rất lớn tới SEO. Thuật ngữ "PR" ngoài được hiểu là "Public Relations" tức mối quan hệ với công chúng, còn được hiểu là "PageRank" chính là trái tim của Google mà ta vừa nghiên cứu.

Các công cụ tìm kiếm đời đầu bị lạm dụng và không chính xác, vì phụ thuộc quá nhiều vào các yếu tố như mật độ từ khoá mà hoàn toàn do người chủ trang web kiểm soát. Trong khi đó, Google sử dụng cả các yếu tố trên trang web (như mật độ từ khoá, meta tag, tiêu đề, đường dẫn và cấu trúc trang) và các yếu tố ngoài trang web (như PageRank và Hyperlink) để tránh khỏi bị lợi dụng như các công cụ tìm kiếm trước đó. Mặc dù PageRank khó bị lừa hơn nhưng những công cụ tạo đường dẫn và các mô hình thao túng Hyperlink đã tồn tại từ trước đó và chúng cũng áp dụng được với PageRank. Nhiều trang chỉ tập trung vào trao đổi, mua bán các đường dẫn, thường trên quy mô lớn. Một vài mô hình đó là *link farm*, tạo ra hàng nghìn trang web với mục đích duy nhất là link spamming.

Cho tới năm 2004, các công cụ tìm kiếm đã tích hợp nhiều yếu tố không được tiết lộ vào trong thuật toán xếp hạng của mình để giảm thiểu tác hại của các chiêu trò điều khiển và thao túng các đường dẫn. Vào năm 2005, Google bắt đầu cá nhân hoá kết quả tìm kiếm của người dùng bằng lịch sử tìm kiếm của họ. Tiếp đó vào năm 2007, Google công bố một chiến dịch chống lại các backlink nhằm chuyển PageRank cho các trang web muốn tăng PageRank. Vào tháng 12 năm 2009, Google thông báo họ sẽ sử dụng lịch sử tìm kiếm của tất cả người dùng để đưa ra kết quả tìm kiếm phù hợp. Từ đó đến nay Google vẫn liên tục thay đổi cuộc chơi khiến cho các người làm SEO phải liên tục thay đổi tư duy, chiến thuật và thích ứng với các quyết định của Google.

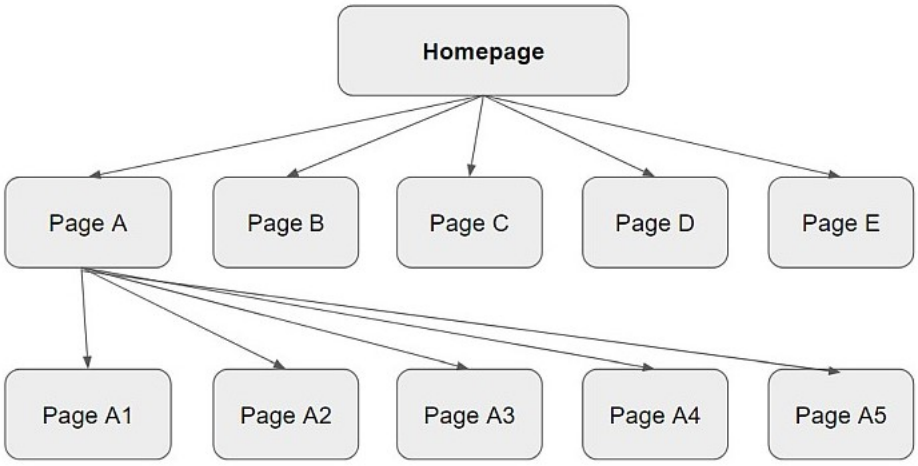
4.2 Một số phương pháp SEO phổ biến

Google luôn đề cao cảnh giác trước những hành vi cố ý lạm dụng PageRank, tuy nhiên vẫn có rất nhiều cách làm SEO được Google chấp nhận và ủng hộ.

Các phương pháp SEO dựa vào thuật toán PageRank

Đảm bảo trang web được Google đánh chỉ mục
Một trong những thành phần của các công cụ tìm kiếm hiện này chính là mô-đun crawler, có

nhiệm vụ thu thập các trang web để trả về trong các kết quả tìm kiếm. Vì vậy trước khi muốn làm tăng PageRank, ta phải đảm bảo trang web của mình được Google tìm thấy. Các spider dựa vào nhiều yếu tố trong đó có khoảng cách từ các trang đến thư mục gốc. Google khuyên các chủ trang web nên để các trang cách trang chủ khoảng 2-3 cú kích chuột. Các trang càng cách xa trang chủ càng có ít khả năng được Google tìm thấy.



Hình 4.1: Một website có khoảng cách tới trang chủ nhiều nhất là 2

Hơn nữa, ngày nay đa số người dùng tìm kiếm bằng các thiết bị di động như điện thoại thông minh, máy tính bảng. Google cũng vì thế mà ưu tiên phiên bản di động của các trang web hơn.

Ngăn chặn đánh chỉ mục với các trang không cần thiết

Để tránh các nội dung không mong muốn được đánh chỉ mục, các chủ trang web có thể thông báo các spider không đi đến các tệp hoặc các thư mục thông qua tệp robots.txt nằm ở thư mục gốc của tên miền. Hơn nữa, một trang có thể được loại ra khỏi danh sách tìm kiếm bằng thêm một thẻ meta (thông thường là <meta name = "robots" content = "noindex">). Khi spider đến một trang web, tệp "robots.txt" nằm ở thư mục gốc của trang web đó là tệp đầu tiên được xem. Tệp tin "robots.txt" sau đó truyền lại thông tin chỉ dẫn cho spider các trang nào không được đánh chỉ mục. Các trang được ngăn chặn đánh chỉ mục thường là các trang liên quan đến vấn đề đăng nhập như trang giỏ hàng của các trang web bán hàng hoặc các nội dung liên quan đến người dùng. Google cũng cảnh báo một số loại trang web sẽ được coi là spam nếu không được ngăn đánh chỉ mục

Đảm bảo không có trang mồ côi

Trang mồ côi (orphan page) là một trang không có bất kỳ liên kết nào dẫn đến nó. Vì vậy mà các trang này không thể được tìm thấy bởi các công cụ tìm kiếm bằng các spider tìm kiếm trên các liên kết nội bộ. Dưới dạng đồ thị, chúng là các đỉnh đứng một mình mà không có cạnh kề, vì vậy không tồn tại đường đi nào đi đến chúng. Các trang mồ côi vẫn có thể được đánh chỉ mục do có đường dẫn từ các trang ngoại lai hoặc được liệt kê trong sitemap. Các trang này với số lượng

nhỏ có thể không gây ra vấn đề, nhưng nếu ở số lượng lớn, chúng có thể làm giảm PageRank của các trang khác vì bản thân chúng không có PageRank để chia sẻ.

Đảm bảo không có trang cuối

Trang cuối (dead-end page) là các trang mà không có liên kết tới một trang web nào khác trong cùng một website tuy nhiên vẫn có thể chứa liên kết tới một trang ngoại lai. Khi người dùng hoặc các spider đến các trang web này họ không thể đi tiếp đến một trang nào nữa và buộc phải quay lại trang trước đó. Dưới dạng đồ thị, các trang này là các đỉnh treo. Điều này không chỉ ảnh hưởng đến trải nhiệm người dùng mà còn ảnh hưởng đến cả PageRank. Chúng được coi là phí phạm tài nguyên vì không chia sẻ PageRank với trang nào khác. Vì vậy mà các trang web theo tiêu chuẩn ngày nay đều có đường dẫn tới trang chủ và các trang quan trọng ngay trên đầu hoặc cuối trang web và sử dụng breadcrumb (mẩu bánh mì). Breadcrumb là một nhóm các đường dẫn được đặt cạnh nhau, thường nằm ở đầu trang giúp người dùng và công cụ tìm kiếm có thể hiểu được vị trí hiện tại và cấu trúc chung của website. Các đường dẫn này giúp tăng PageRank cho trang chủ và các trang quan trọng của website.

Hình 4.2: Các liên kết trên đầu trang và breadcrumb

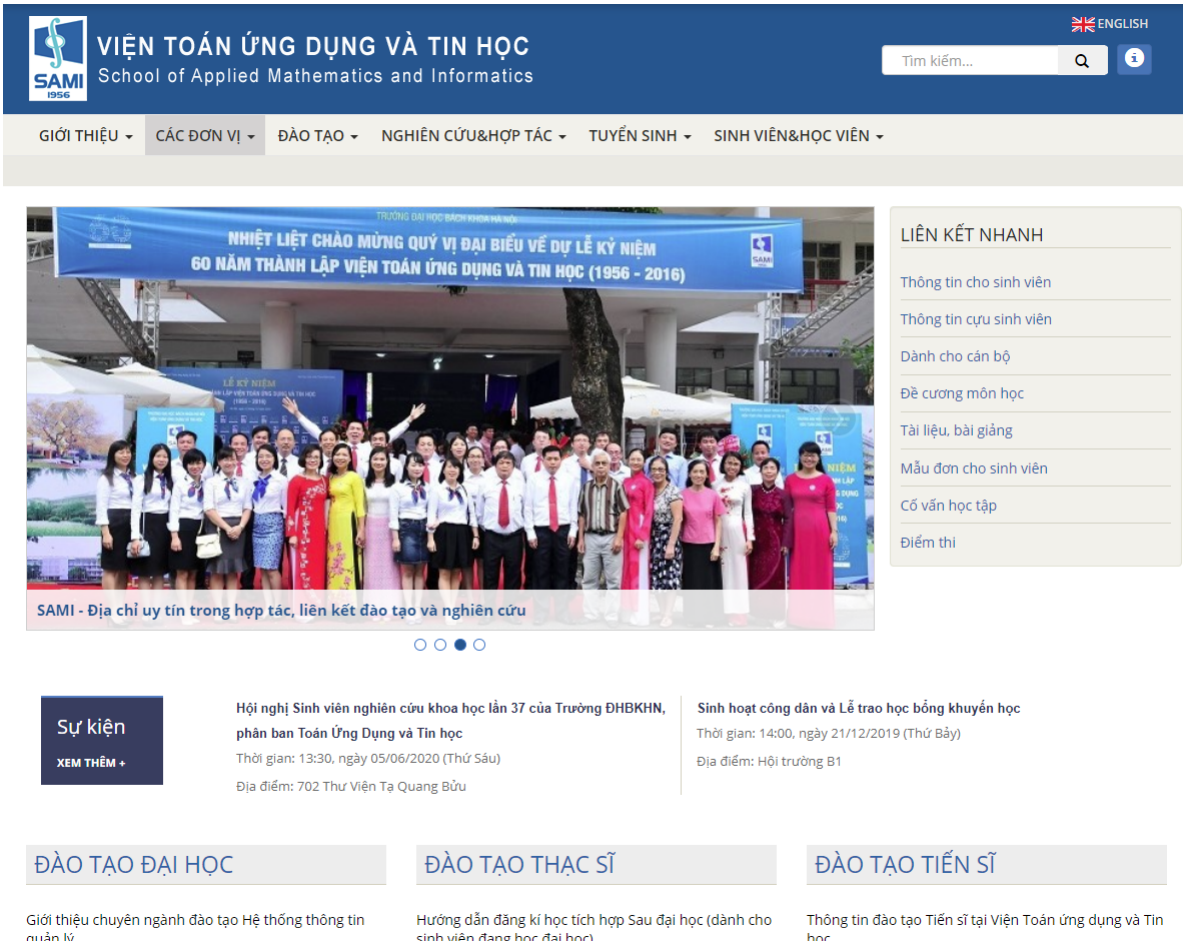
Liên kết với các trang web có uy tín

Các chủ website có thể đặt liên kết tới các website khác để chia sẻ PageRank cho nhau. Khi một website có đường dẫn đến nhiều website có PageRank cao thì PageRank của website đó cũng tăng lên và ngược lại. Chất lượng của backlink cũng rất quan trọng. Một đường dẫn của website uy tín có thể hơn nhiều đường dẫn của các website chất lượng thấp.

Các phương pháp SEO không dựa vào thuật toán PageRank

Thiết kế trang web thân thiện với SEO

Nhiều website chỉ tập trung vào tính năng và nội dung mà không tập trung vào thiết kế giao diện và trải nhiệm của người dùng. Một hệ thống menu rõ ràng, dễ sử dụng có thể giúp người dùng dễ tìm ra thông tin mà họ cần. Không nên để quá nhiều tiêu đề, sử dụng các danh sách thả xuống trong từng mục. Hơn nữa, các trang web có tốc độ nhanh sẽ được giữ chân người đọc được lâu hơn và các spider cũng ưu ái các trang web này hơn. Các trang web cũ và lạc hậu sẽ ít được chú ý hơn mặc dù nội dung có thể tốt



Hình 4.3: Thiết kế một trang web thân thiện với SEO

Tối ưu các từ khoá

SEO dựa vào từ khoá là một trong những hình thức phổ biến nhất của SEO. Việc sử dụng từ khoá trong tiêu đề có thể có ảnh hưởng ít nhiều tới PageRank. Nội dung bài viết cũng là một trong những yếu tố mà Google và các công cụ tìm kiếm đánh giá để xếp hạng website. Những từ khoá thường xuyên sử dụng nói cho công cụ tìm kiếm và cả người dùng biết website đó nói về cái gì. Không những thế, ta cần phải chọn đúng từ khoá, để thu hút đúng đối tượng mà ta mong muốn. Thông thường người dùng thường nghĩ đến các từ khoá ngắn gọn, thế nhưng đối với SEO, sử dụng các từ khoá ngắn và chung chung sẽ gặp phải rất nhiều cạnh tranh. Thay vào đó nên tập trung vào các từ khoá dài và cụ thể, chúng có ít cạnh tranh vào tạo cơ hội cho website dễ được tìm thấy hơn. Ví dụ

- Từ khoá "Đồ nội thất" có 100.000 kết quả
- Từ khoá "Đồ nội thất cao cấp" có 50.000 kết quả
- Từ khoá "Đồ gỗ nội thất cao cấp Châu Âu" có 1.000 kết quả

- Từ khoá "Đồ gỗ nội thất Châu Âu tại Hà Nội" có 1 kết quả

Sử dụng từ khoá làm các anchor text
Anchor text chính là đoạn văn bản chứa liên kết đến 1 trang khác. Việc sử dụng từ khoá làm anchor text sẽ đem lại cảm giác tiện lợi đối với người dùng. Khi đọc bài viết đến một từ khoá quan trọng mà người đọc muốn tìm hiểu thêm, người đọc khi đó chỉ cần kích chuột vào từ khoá đó sẽ đưa họ đến trang web khác có nội dung liên quan đến từ khoá đó. Các công cụ tìm kiếm cũng dựa vào đó mà đánh giá chất lượng bài viết.

Đảm bảo chất lượng về nội dung
Chỉ sử dụng nhiều từ khoá thôi là chưa đủ, ta còn cần phải đảm bảo chất lượng của các bài viết. Qua các năm tiêu chuẩn mà Google đưa ra ngày càng cao và những chủ website cũng phải cố gắng tiến tới tiêu chuẩn đó. Để làm được điều này các chủ website cần phải đảm bảo số lượng bài viết và mức độ phong phú, các bài viết cũ và không còn phù hợp cần được loại bỏ. Khi nhiều người cảm thấy nội dung có chất lượng, trang web có thể được nhắc đến ở các website khác, tăng số lượng backlink, dẫn đến tăng PageRank.

Các kỹ thuật không được Google chấp nhận
Các không được Google ủng hộ hay chấp nhận thường sử dụng các thủ đoạn hay mánh khoé để đánh lừa các công cụ tìm kiếm. Một phương pháp phổ biến và lâu đời đó là sử dụng các đoạn văn bản ẩn (hidden text), thường là các đoạn văn bản có màu chữ trùng với màu nền, hoặc nằm ở ngoài vùng hiển thị của màn hình. Phương pháp này có thể sử dụng thể giấu các đường dẫn đến các website muốn tăng PageRank hoặc để tăng mật độ từ khoá. Một trong những kỹ thuật khác đó là **cloaking**, là một thủ thuật che giấu nội dung trang web bằng việc trả về các trang web khác nhau cho các spider và cho người dùng. Nhờ vào đó phiên bản trang web trả về cho spider chỉ có mục đích duy nhất là làm tăng PageRank còn trang web trả về cho người dùng thì nội dung có chất lượng kém. Hiện nay các kỹ thuật này rất dễ bị Google phát hiện và khi bị phát hiện có thể bị phạt rất nặng. Các website có thể bị giảm thứ hạng, hoặc bị xoá khỏi danh mục của Google vĩnh viễn. Quá trình xử phạt có thể tự động bằng các thuật toán hoặc do các phản hồi của người dùng.

Kết luận

Trong báo cáo này em đã trình bày về cơ sở lý thuyết, ý tưởng thuật toán, phương thức cài đặt thuật toán PageRank của Google và ứng dụng của PageRank trong lĩnh vực SEO. Cụ thể hơn, em đã trình bày các nội dung:

1. Giới thiệu các mô hình thu thập thông tin cổ điển.
2. Các vấn đề trong thu thập thông tin trên Web.
3. Cở sở lý thuyết của thuật toán PageRank.
4. Ý tưởng và cài đặt thuật toán PageRank.
5. Cài đặt chương trình và đánh giá kết quả.
6. Ứng dụng của PageRank trong SEO.

Kết thúc đồ án em đã rút ra được cách thức hoạt động của các công cụ tìm kiếm và các phương pháp làm tăng thứ hạng của website trong các kết quả tìm kiếm. Đồ án mới chỉ dừng lại ở mức độ giới thiệu mà chưa đi sâu vào các ứng dụng khác của thuật toán PageRank. Một vài các ứng dụng đó có thể kể đến:

- Xếp hạng các bài đăng trên mạng xã hội.
- Gợi ý kết bạn trên mạng xã hội.
- Đưa ra các hệ thống gợi ý (Recommendation system).

Danh sách hình vẽ

Tài liệu tham khảo

[1] Amy N. Langville and Carl D. Meyer. Google's PageRank and Beyond: The Science of Search Engine Rankings, Princeton University Press, 2006.

[2] Nancy Blachman, Eric Fredricksen, and Fritz Schneider. How to Do Everything with Google. McGraw-Hill, 2003.

[3] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 33:107–17, 1998.

[4] Sergey Brin, Lawrence Page, R. Motwami, and Terry Winograd. The PageRank citation ranking: Bringing order to the Web. Technical Report 1999-0120, Computer Science Department, Stanford University, 1999.

[5] Carl D. Meyer. Matrix Analysis and Applied Linear Algebra. SIAM, Philadelphia, 2000.

[6] Tống Đình Quỳ, Xác suất thống kê, NXB Giáo dục, 2000.

[7] Yousef Saad. Iterative Methods for Sparse Linear Systems. SIAM, 2003.

[8] Beel, Jöran and Gipp, Bela and Wilde, Erik. Academic Search Engine Optimization (ASEO): Optimizing Scholarly Literature for Google Scholar and Co. Journal of Scholarly Publishing, 2010.

[9] Ortiz-Cordova, A. and Jansen, B. J. Classifying Web Search Queries in Order to Identify High Revenue Generating Customers. Journal of the American Society for Information Sciences and Technology, 2012.

[10] Hansell, Saul. Google Keeps Tweaking Its Search Engine. New York Times, 2007.

[11] Shari Thurow. The Most Important SEO Strategy. clickz.com. 2006.