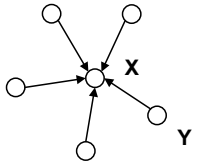
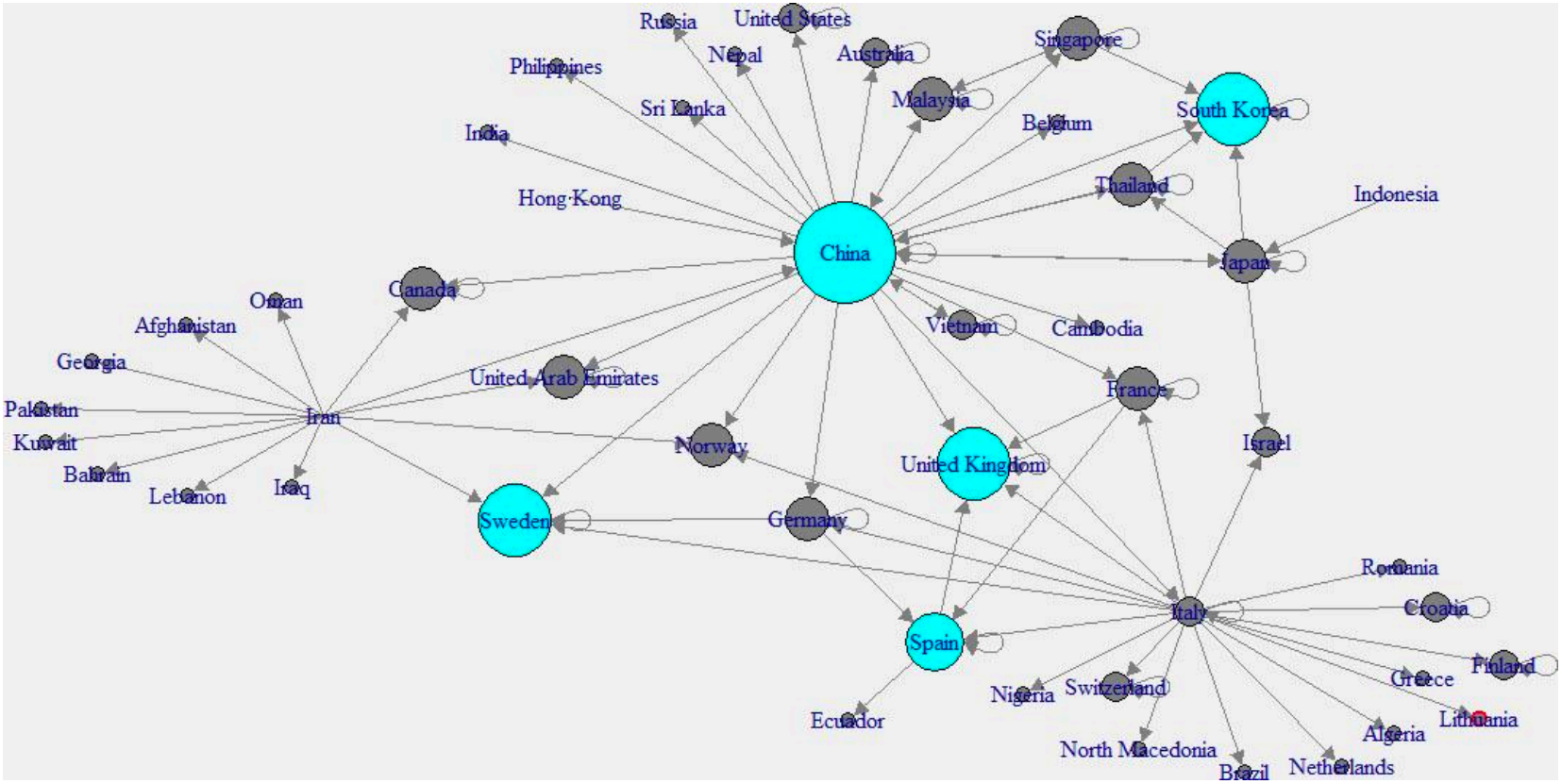
Centralité dans le cas de réseaux orientés

- Graphe du web
- Réseau trophique (chaînes alimentaires)
- Réseaux d'influence
- Réseaux héréditaires
- Réseaux de citations
- Réseaux neuronaux

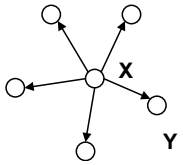
Centralité de degré entrant

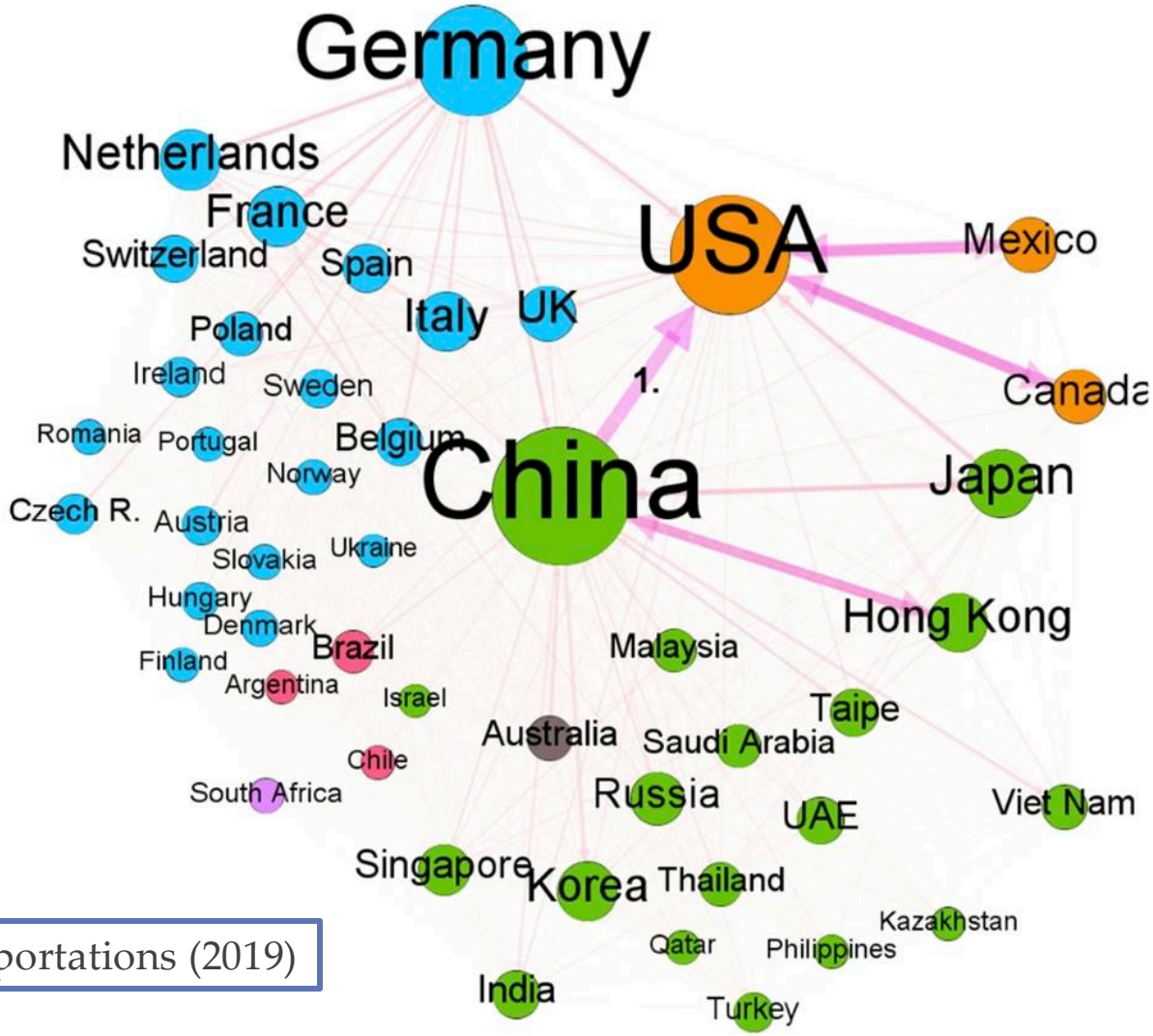




Déplacement des personnes ayant contractés le COVID-19 (janvier-mars 2020)

Centralité de degré sortant





International exportations (2019)

La centralité de Proximité dans les réseaux orientés

- On choisit une direction (entrant / sortant)
  - Proximité entrante (e.g. prestige dans les réseaux de citations)
  - Proximité sortante (e.g. prestige dans les réseaux d'influence)

- La **proximité entrante d'un nœud x** : somme des longueurs des plus courts chemins entre tous les autres nœuds et x.

$$C_c(x) = \frac{1}{\sum_{y \neq x} \mathrm{dist}(y, x)}$$

- Normalisation : $\overline{C_c} = (n-1)C_c$

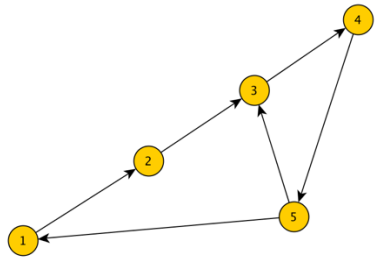
- Dans le cas où il n'y a pas de chemin dirigé entre x et y :

$$C_c(x) = \sum_{y \neq x} \frac{1}{\mathrm{dist}(y, x)} \quad \text{avec la convention } \frac{1}{\infty} = 0$$

La centralité d'intermédiarité dans les réseaux orientés

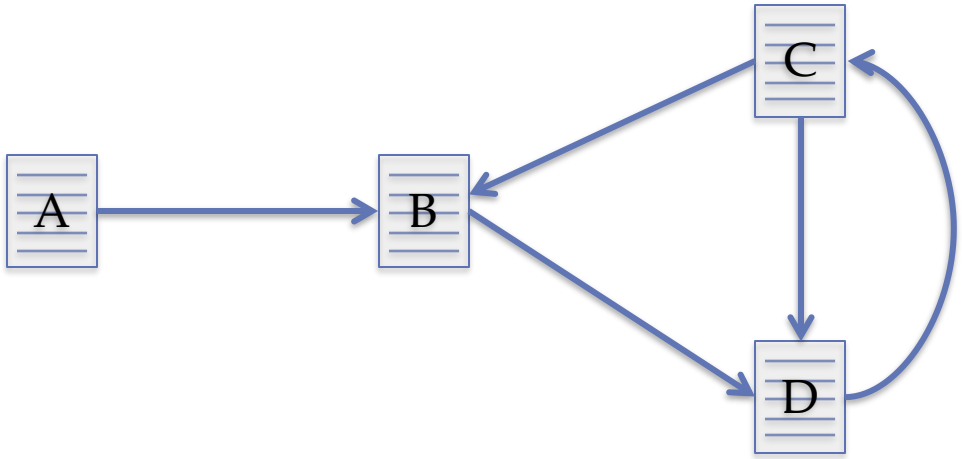
- On considère les chemins orientés
- $$C_B(i) = \sum_{i \neq j \neq k} \frac{\#\text{ plus courts chemins entre } j \text{ et } k \text{ passant par } i}{\#\text{ plus courts chemins entre } j \text{ et } k}$$
- Différence au moment de la normalisation :
  - Cas non orienté (rappel) : $\overline{C_B(i)} = \frac{2\,C_B(i)}{(n-1)(n-2)}$
  - Cas orienté : $\overline{C_B(i)} = \frac{C_B(i)}{(n-1)(n-2)}$

Un nœud qui est sur le chemin ij
n'est pas forcément sur le chemin ji



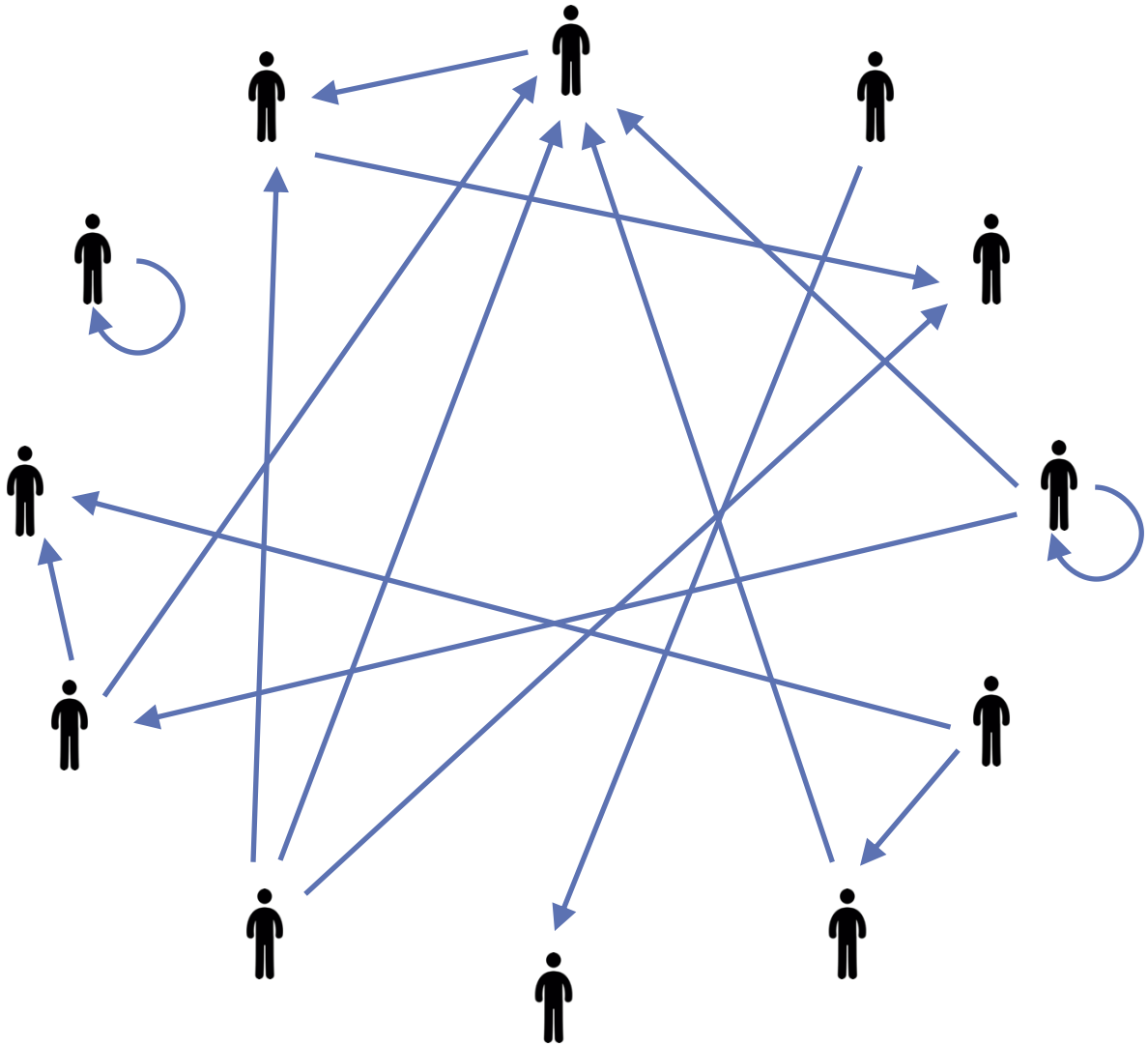
Et pour les pages web ?
Comment les classer ?

- Le web est un graphe :
  - Les nœuds sont les pages
  - Les arêtes sont les liens hypertexte présents sur les pages web

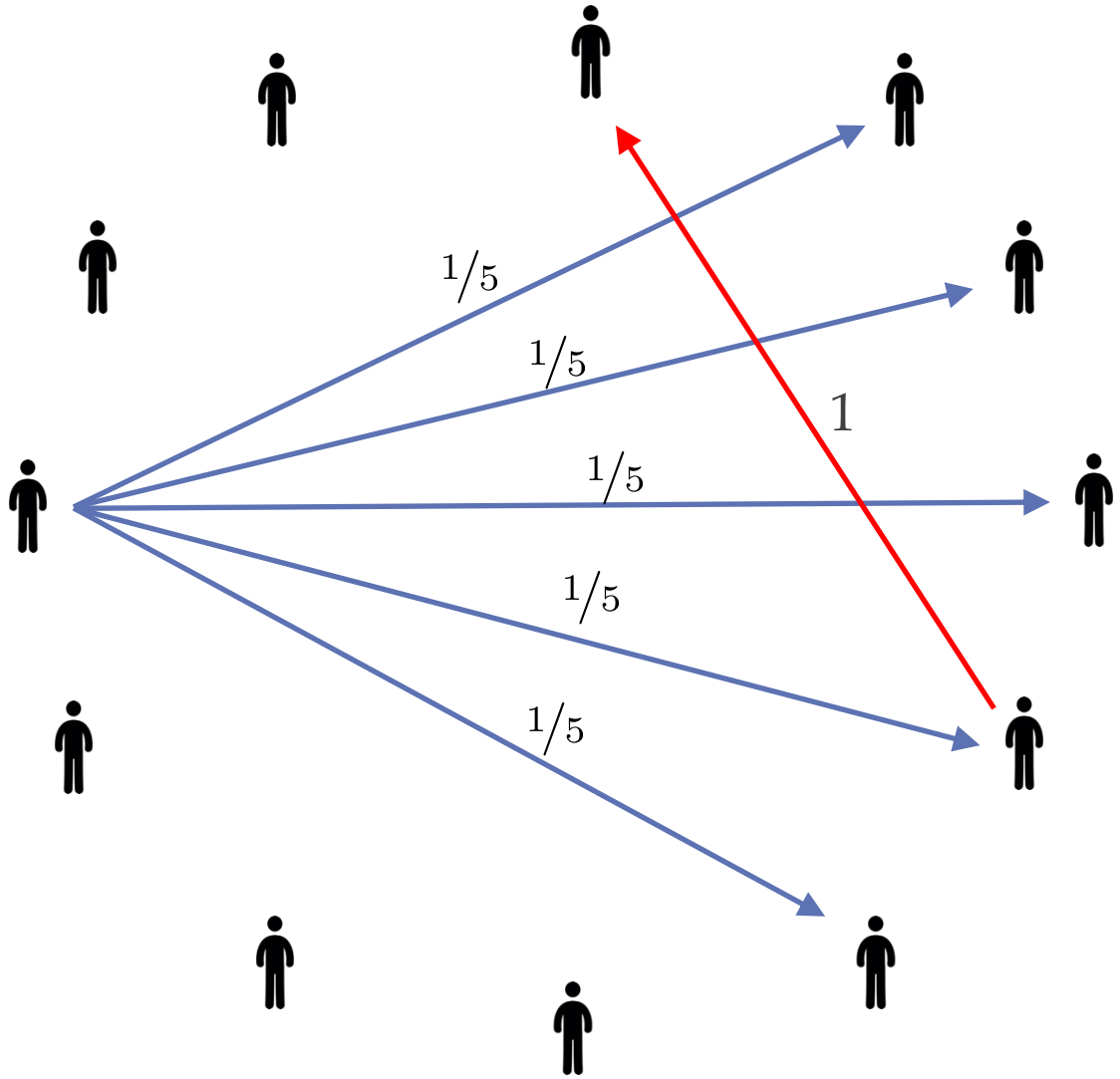


- Comment caractériser l'importance d'une page ? **Nombre de liens entrants** ? Proximité ? Intermédiarité ? Explorer son contenu ?

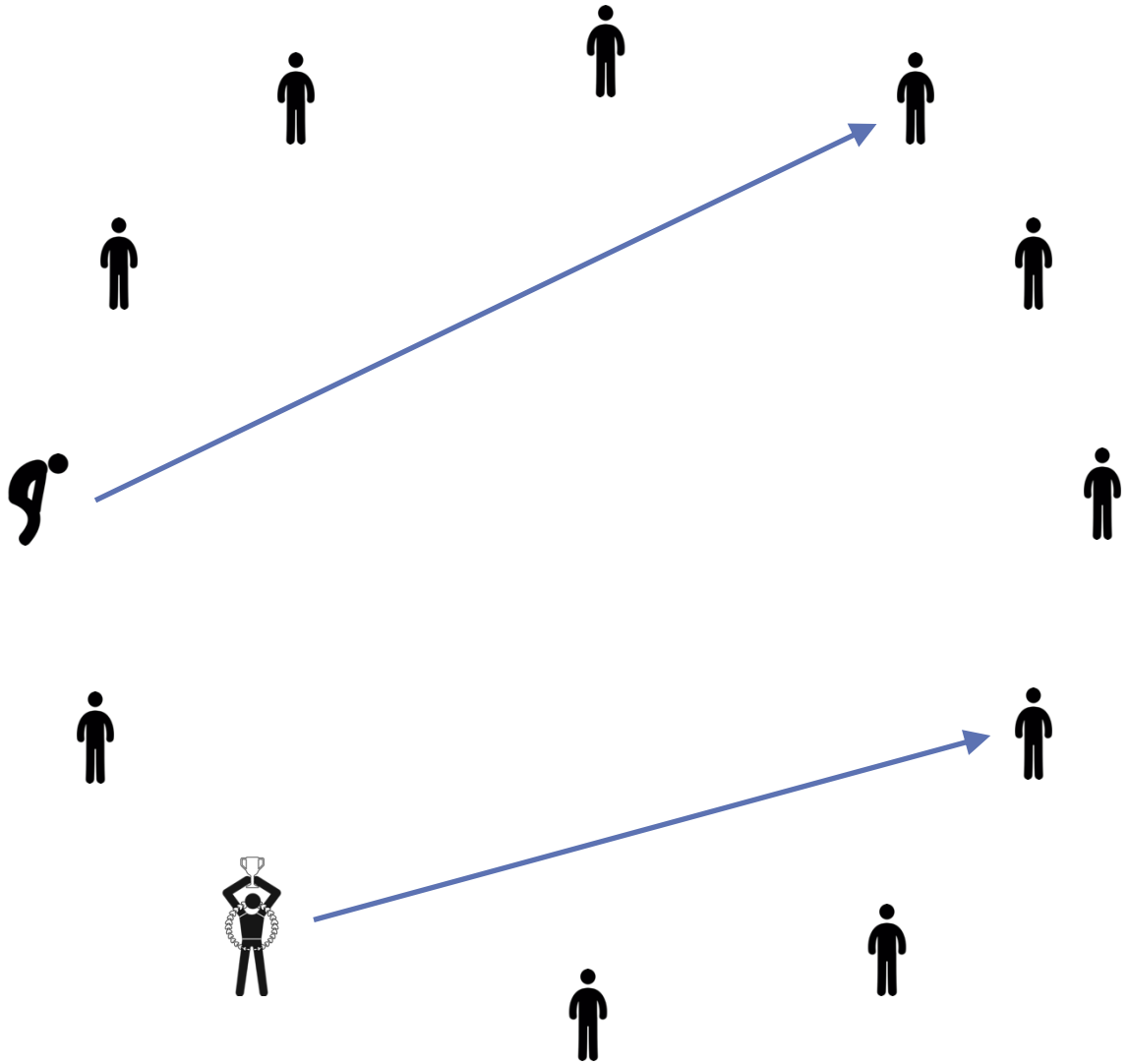
Exemple : Qui est bon en sport ?



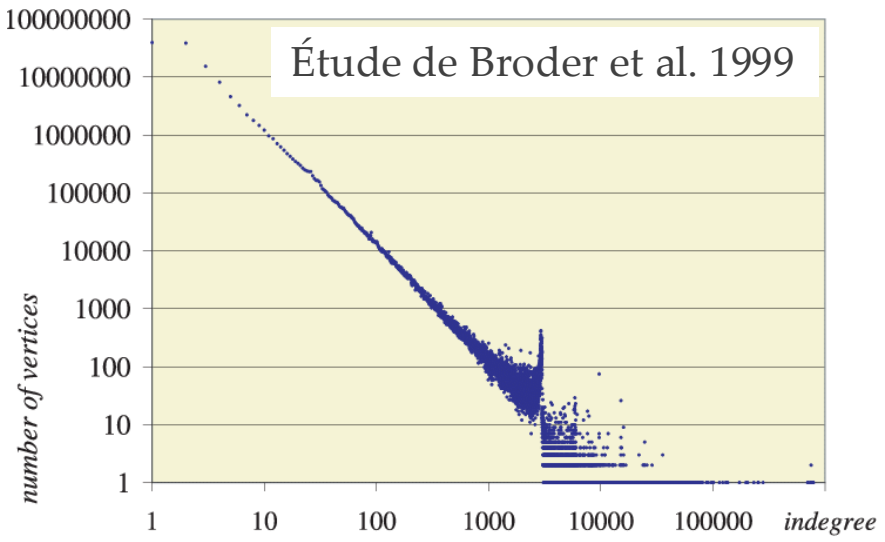
Les votes ont-ils tous la même importance ?



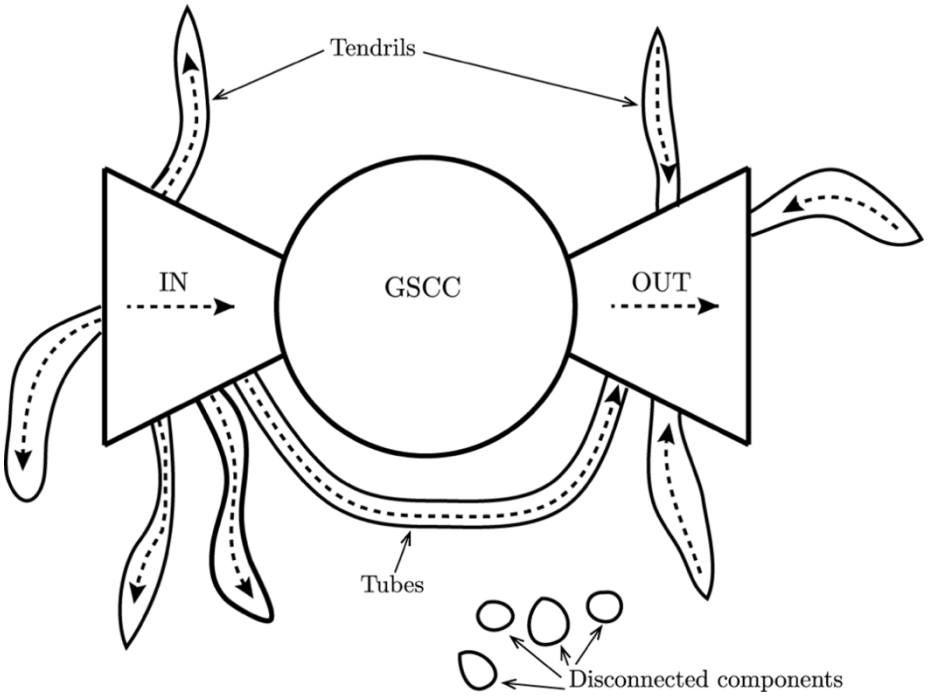
Les votes ont-ils tous la même importance ?



Structure du web à la fin des années 90



Sur un graphe de 2×10^8 sommets et 1.5×10^9 arcs obtenu à partir d'un « crawl » d'Altavista



- Structure du web en « nœud papillon ».
- Plus de la moitié des sommets sont hors de la CFC !
- Aujourd'hui : 4,65 milliards de pages « visibles » (estimé à seulement 10% du total)

Brin, Page (1998) - The Anatomy of a Large-Scale Hypertextual Web Search Engine

The Anatomy of a Large-Scale Hypertextual Web Search Engine

Sergey Brin and Lawrence Page

Computer Science Department,
Stanford University, Stanford, CA 94305, USA
sergey@cs.stanford.edu and page@cs.stanford.edu

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/
To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and web proliferation, creating a web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale web search engine -- the first such detailed public description we know of to date. Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords

World Wide Web, Search Engines, Information Retrieval, PageRank, Google

1. Introduction

(Note: There are two versions of this paper -- a longer full version and a shorter printed version. The full version is available on the web and the conference CD-ROM.)
The web creates new challenges for information retrieval. The amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as Yahoo! or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead automated search engines. We have built a large-scale search engine which addresses many of the problems of existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. We chose our system name, Google, because it is a common spelling of googol, or 10^{100} and fits well with our goal of building very large-scale search engines.

1.1 Web Search Engines -- Scaling Up: 1994 - 2000

Search engine technology has had to scale dramatically to keep up with the growth of the web. In 1994, one of the first web search engines, the World Wide Web Worm (WWWW) [McBryan 94] had an index of 110,000 web pages and web accessible documents. As of November, 1997, the top search engines claim to index from 2 million (WebCrawler) to 100 million web documents (from Search Engine Watch). It is foreseeable that by the year 2000, a comprehensive index of the Web will contain over a billion documents. At the same time, the number of queries search engines handle has grown incredibly too. In March and April 1994, the World Wide Web Worm received an average of about 1500 queries per day. In November 1997, Altavista claimed it handled roughly 20 million queries per day. With the increasing number of users on the web, and automated systems which query search engines, it is likely that top search engines will handle hundreds of millions of queries per day by the year 2000. The goal of our system is to address many of the problems, both in quality and scalability, introduced by scaling search engine technology to such extraordinary numbers.

1.2. Google: Scaling with the Web

Creating a search engine which scales even to today's web presents many challenges. Fast crawling technology is needed to gather the web documents and keep them up to date. Storage space must be used efficiently to store indices and, optionally, the documents themselves. The indexing system must process hundreds of gigabytes of data efficiently. Queries must be handled quickly, at a rate of hundreds to thousands per second.

These tasks are becoming increasingly difficult as the Web grows. However, hardware performance and cost have improved dramatically to partially offset the difficulty. There are, however, several notable exceptions to this progress such as disk seek time and operating system robustness. In designing Google, we have considered both the rate of growth of the Web and technological changes. Google is designed to scale well to extremely large data sets. It makes efficient use of storage space to store the index. Its data structures are optimized for fast and efficient access (see section 4.2). Further, we expect that the cost to index and store text or HTML will eventually decline relative to the amount that will be available (see Appendix B). This will result in favorable scaling properties for centralized systems like Google.

1.3 Design Goals

1.3.1 Improved Search Quality

Our main goal is to improve the quality of web search engines. In 1994, some people believed that a complete search index would make it possible to find anything easily. According to Best of the Web 1994 -- Navigators, "The best navigation service should make it easy to find almost anything on the Web (once all the data is entered)." However, the Web of 1997 is quite different. Anyone who has used a search engine recently, can readily testify that the completeness of the index is not the only factor in the quality of search results. "Junk results" often wash out any results that a user is interested in. In fact, as of November 1997, only one of the top four commercial search engines finds itself (returns its own search page in response to its name in the top ten results). One of the main causes of this problem is that the number of documents in the indices has been increasing by many orders of magnitude, but the user's ability to look at documents has not. People are still only willing to look at the first few tens of results.

Brin, Page (1998) - The Anatomy of a Large-Scale Hypertextual Web Search Engine

Because of this, as the collection size grows, we need tools that have very high precision (number of relevant documents returned, say in the top tens of results). Indeed, we want our notion of "relevant" to only include the very best documents since there may be tens of thousands of slightly relevant documents. This very high precision is important even at the expense of recall (the total number of relevant documents the system is able to return). There is quite a bit of recent optimism that the use of more hypertextual information can help improve search and other applications [Marchiori 97] [Spertus 97] [Weiss 96] [Kleinberg 98]. In particular, link structure [Page 98] and link text provide a lot of information for making relevance judgments and quality filtering. Google makes use of both link structure and anchor text (see Sections 2.1 and 2.2).

1.3.2 Academic Search Engine Research

Aside from tremendous growth, the Web has also become increasingly commercial over time. In 1993, 1.5% of web servers were on .com domains. This number grew to over 60% in 1997. At the same time, search engines have migrated from the academic domain to the commercial. Up until now most search engine development has gone on at companies with little publication of technical details. This causes search engine technology to remain largely a black art and to be advertising oriented (see Appendix A). With Google, we have a strong goal to push more development and understanding into the academic realm.

Another important design goal was to build systems that reasonable numbers of people can actually use. Usage was important to us because we think some of the most interesting research will involve leveraging the vast amount of usage data that is available from modern web systems. For example, there are many tens of millions of searches performed every day. However, it is very difficult to get this data, mainly because it is considered commercially valuable.

Our final design goal was to build an architecture that can support novel research activities on large-scale web data. To support novel research uses, Google stores all of the actual documents it crawls in compressed form. One of our main goals in designing Google was to set up an environment where other researchers can come in quickly, process large chunks of the web, and produce interesting results that would have been very difficult to produce otherwise. In the short time the system has been up, there have already been several papers using databases generated by Google, and many others are underway. Another goal we have is to set up a Spacelab-like environment where researchers or even students can propose and do interesting experiments on our large-scale web data.

2. System Features

The Google search engine has two important features that help it produce high precision results. First, it makes use of the link structure of the Web to calculate a quality ranking for each web page. This ranking is called PageRank and is described in detail in [Page 98]. Second, Google utilizes link to improve search results.

2.1 PageRank: Bringing Order to the Web

The citation (link) graph of the web is an important resource that has largely gone unused in existing web search engines. We have created maps containing as many as 518 million of these hyperlinks, a significant sample of the total. These maps allow rapid calculation of a web page's "PageRank", an objective measure of its citation importance that corresponds well with people's subjective idea of importance. Because of this correspondence, PageRank is an excellent way to prioritize the results of web keyword searches. For most popular subjects, a simple text matching search that is restricted to web page titles performs admirably when PageRank prioritizes the results (demo available at google.stanford.edu). For the type of full text searches in the main Google system, PageRank also helps a great deal.

2.1.1 Description of PageRank Calculation

Academic citation literature has been applied to the web, largely by counting citations or backlinks to a given page. This gives some approximation of a page's importance or quality. PageRank extends this idea by not counting links from all pages equally, and by normalizing by the number of links on a page. PageRank is defined as follows:

> *We assume page A has pages T1...Tn which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:*
>
> $PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))$
>
> *Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one.*

PageRank or *PR(A)* can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web. Also, a PageRank for 26 million web pages can be computed in a few hours on a medium size workstation. There are many other details which are beyond the scope of this paper.

2.1.2 Intuitive Justification

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page. The probability that the random surfer visits a page is its PageRank. And, the *d* damping factor is the probability at each page the "random surfer" will get bored and request another random page. One important variation is to only add the damping factor *d* to a single page, or a group of pages. This allows for personalization and can make it nearly impossible to deliberately mislead the system in order to get a higher ranking. We have several other extensions to PageRank, again see [Page 98].

Another intuitive justification is that a page can have a high PageRank if there are many pages that point to it, or if there are some pages that point to it and have a high PageRank. Intuitively, pages that are well cited from many places around the web are worth looking at. Also, pages that have perhaps only one citation from something like the Yahoo! homepage are also generally worth looking at. If a page was not high quality, or was a broken link, it is quite likely that Yahoo's homepage would not link to it. PageRank handles both these cases and everything in between by recursively propagating weights through the link structure of the web.

a great deal.

2.1.1 Description of PageRank Calculation

Academic citation literature has been applied to the web, largely by counting citations or backlinks to a given page. This gives some approximation of a page's importance or quality. PageRank extends this idea by not counting links from all pages equally, and by normalizing by the number of links on a page. PageRank is defined as follows:

> *We assume page A has pages T1...Tn which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:*
>
> $PR(A) = (1-d) + d\ (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))$
>
> *Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one.*

PageRank or *PR(A)* can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web. Also, a PageRank for 26 million web pages can be computed in a few hours on a medium size workstation. There are many other details which are beyond the scope of this paper.

2.1.2 Intuitive Justification

PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page. The probability that the random surfer visits a page is its PageRank. And, the *d* damping factor is the probability at each page the "random surfer" will get bored and request another random page. One important variation is to only add the damping factor *d* to a single page, or a group of pages. This allows for personalization and can make it nearly impossible to deliberately mislead the system in order to get a higher ranking. We have several other extensions to PageRank, again see [Page 98].

Another intuitive justification is that a page can have a high PageRank if there are many pages that point to it, or if there are some pages that point to it and have a high PageRank. Intuitively, pages that are well cited from many places around the web are worth looking at. Also, pages that have perhaps only one citation from something like the Yahoo! homepage are also generally worth looking at. If a page was not high quality, or was a broken link, it is quite likely that Yahoo's homepage would not link to it.

Brin, Page (1998) - The Anatomy of a Large-Scale Hypertextual Web Search Engine

- Modèle de S. Brin et L. Page :
  - Accorder plus d'importance aux pages référencées par des pages qui font autorité.
  - Accorder moins de crédit à une référence si elle provient d'une page avec de nombreux liens.

PageRank (Page, Brin 1998)

- Page Rank : Pensé pour modéliser le comportement d'un utilisateur qui surfe de liens en liens (ne clique jamais sur « back ») et peut éventuellement aller ailleurs en saisissant une URL.

- Processus vu comme une marche aléatoire sur le graphe.

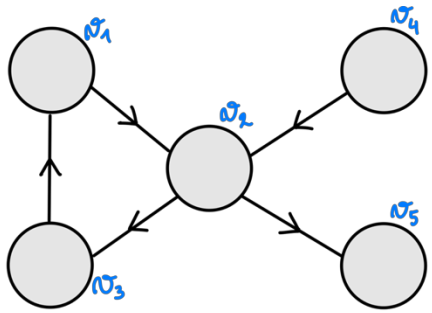
- Le score de chaque nœud est la probabilité d'être sur ce nœud au cours la marche aléatoire : $\sum_i R(i) = 1$

PageRank (Page, Brin 1998)

Page Rank = Centralité de vecteur propre dans le cas orienté :

$$R(i) = \frac{1}{\lambda} \sum_{j \text{ voisin entrant de i}} R(j)$$

Matrice d'adjacence = liens sortants.



$$A^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

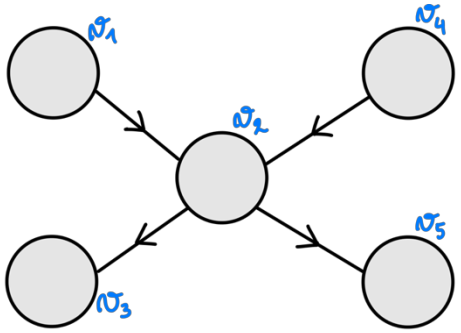
Pour le Page Rank, importance donnée aux pages ayant beaucoup de liens entrants : transposition de la matrice d'adjacence.

Notation matricielle : $R = \frac{1}{\lambda} A^T R$

PageRank (Page, Brin 1998)

Plusieurs problèmes se posent :

1. Nœuds absorbants (puits)
2. Cycles
3. Nœuds sources



1. Les nœuds absorbants (puits)

$$R = \begin{pmatrix} 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \end{pmatrix} \rightarrow A^T R = \begin{pmatrix} 0 \\ 2/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix} \rightarrow \frac{5}{4} A^T R = \begin{pmatrix} 0 \\ 1/2 \\ 1/4 \\ 0 \\ 1/4 \end{pmatrix}$$

$$R = \begin{pmatrix} 0 \\ 1/2 \\ 1/4 \\ 0 \\ 1/4 \end{pmatrix} \rightarrow A^T R = \begin{pmatrix} 0 \\ 0 \\ 1/4 \\ 0 \\ 1/4 \end{pmatrix} \rightarrow 2A^T R = \begin{pmatrix} 0 \\ 0 \\ 1/2 \\ 0 \\ 1/2 \end{pmatrix}$$

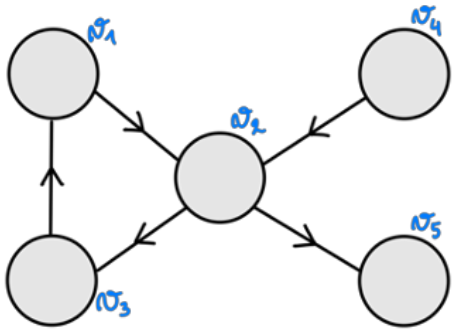
$$A^T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$R = \begin{pmatrix} 0 \\ 0 \\ 1/2 \\ 0 \\ 1/2 \end{pmatrix} \rightarrow A^T R = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

PageRank (Page, Brin 1998)

Plusieurs problèmes se posent :

1. Nœuds absorbants (puits)
2. Cycles
3. Nœuds sources



2. Les Cycles

$$A^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$R = \begin{pmatrix} 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \end{pmatrix} \Rightarrow A^T R = \begin{pmatrix} 1/5 \\ 2/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix}$$

$$R = \begin{pmatrix} 1/5 \\ 2/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix} \Rightarrow A^T R = \begin{pmatrix} 1/5 \\ 1/5 \\ 2/5 \\ 0 \\ 1/5 \end{pmatrix}$$

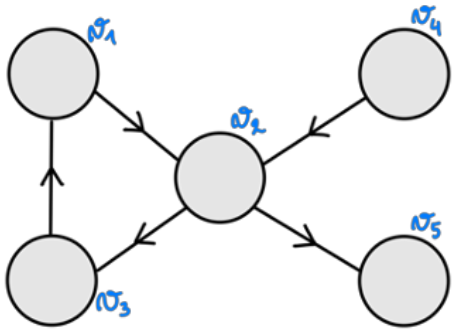
$$R = \begin{pmatrix} 1/6 \\ 1/6 \\ 1/3 \\ 0 \\ 1/3 \end{pmatrix} \Rightarrow A^T R = \begin{pmatrix} 1/3 \\ 1/6 \\ 1/6 \\ 0 \\ 1/6 \end{pmatrix} \Rightarrow \frac{6}{5} A^T R = \begin{pmatrix} 2/5 \\ 1/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix}$$

$$R = \begin{pmatrix} 2/5 \\ 1/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix} \Rightarrow A^T R = \begin{pmatrix} 1/5 \\ 2/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix}$$

PageRank (Page, Brin 1998)

Plusieurs problèmes se posent :

1. Nœuds absorbants (puits)
2. Cycles
3. Nœuds sources



3. Les nœuds sources

$$A^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$R = \begin{pmatrix} 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \end{pmatrix} \Rightarrow A^T R = \begin{pmatrix} 1/5 \\ 2/5 \\ 1/5 \\ 0 \\ 1/5 \end{pmatrix}$$

PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

1. Première adaptation :

1 page = 1 score distribué de manière équitable

$$D^{-1} = \begin{pmatrix} \frac{1}{d^+(v_1)} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{d^+(v_2)} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{d^+(v_n)} \end{pmatrix}$$

Probabilité de suivre un lien à partir de $v_i = \dfrac{1}{d^+(v_i)}$ si $d^+(v_i) \neq 0$;

Sinon 0.

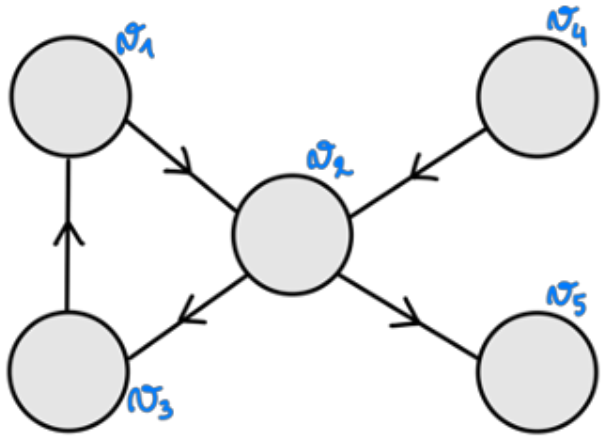
PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

1. Première adaptation :

1 page = 1 score distribué de manière équitable

Exemple :



$$D^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

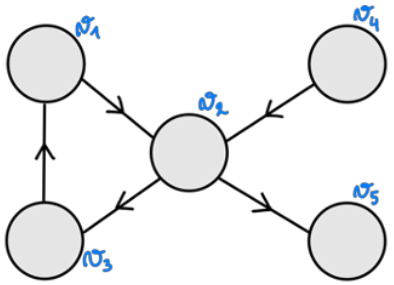
PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

1. Première adaptation :

1 page = 1 score distribué de manière équitable

La matrice de passage se définit comme : $P = D^{-1}A$



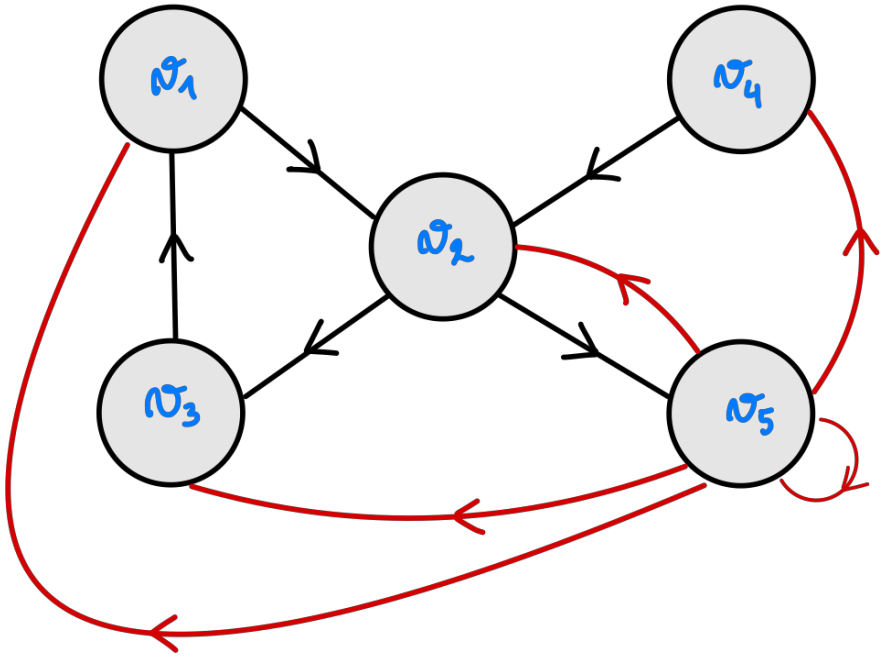
$$P = D^{-1}A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

2. Deuxième adaptation :

Régler le problème des nœuds absorbants : ajout de liens fictifs



PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

2. Deuxième adaptation :

Régler le problème des nœuds absorbants : ajout de liens fictifs

Soit s un vecteur-colonne de taille n indiquant quels sont les puits.

Soit u le vecteur-colonne unitaire.

$$s = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad u = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad u^T = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\frac{1}{n} s u^T = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{1}{n} & \frac{1}{n} & \frac{1}{n} & \frac{1}{n} & \frac{1}{n} \end{pmatrix}$$

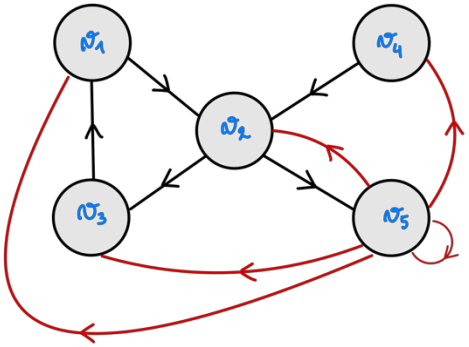
PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

2. Deuxième adaptation :

Régler le problème des nœuds absorbants : ajout de liens fictifs

La matrice de passage devient : $P = D^{-1}A + \frac{1}{n}su^T$



$$P = D^{-1}A + \frac{1}{n}su^T = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix}$$

Toutes les lignes somment à 1 !

PageRank (Page, Brin 1998)

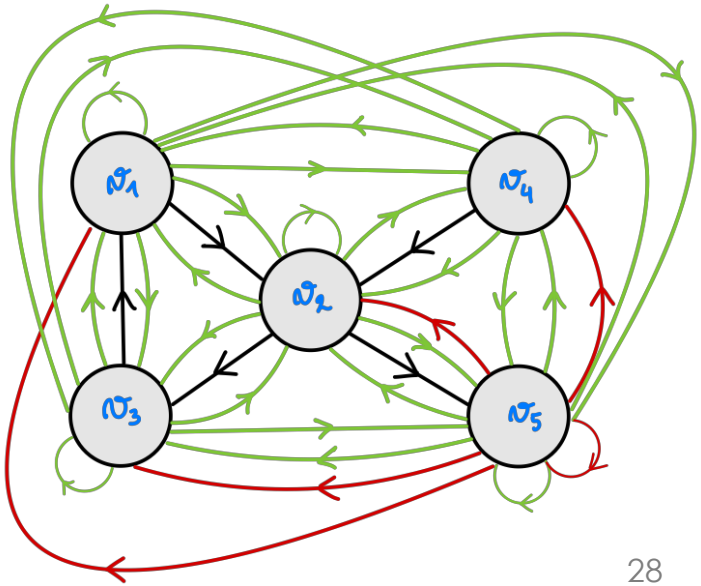
S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

3. Troisième adaptation :

Régler le problème des nœuds source : ajout de liens fictifs

Pages accessibles uniquement en saisissant leur adresse.

- Avec probabilité α, je clique sur un « lien »
- Avec probabilité $1-\alpha$, je me rends sur une page au hasard (i.e. je saisis une adresse).
- α : Facteur de zap (Dumping factor)



- Liens noirs : $\frac{\alpha}{d^+(v_i)}$
- Liens verts : $\frac{1-\alpha}{n}$
- Liens rouge : $\frac{\alpha}{n}$

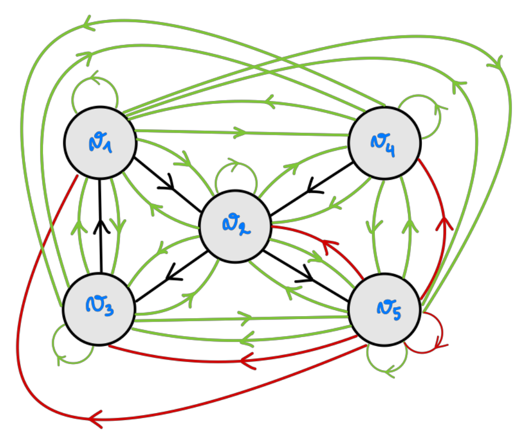
PageRank (Page, Brin 1998)

S. Brin et L. Page adapte l'algorithme de puissance itérée en 1998 en remplaçant la matrice d'adjacence par une matrice de « passage » ad hoc.

$$P = \alpha\left(D^{-1}A + \frac{1}{n} s u^T\right) + (1-\alpha)\frac{uu^T}{n}$$

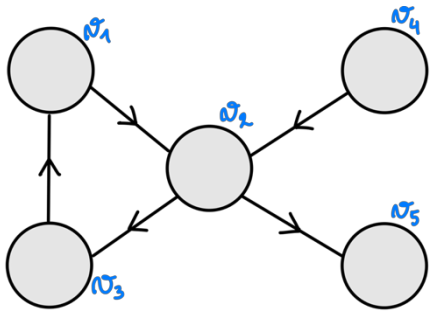
$$= \begin{pmatrix} 0 & \alpha & 0 & 0 & 0 \\ 0 & 0 & \frac{\alpha}{2} & 0 & \frac{\alpha}{2} \\ \alpha & 0 & 0 & 0 & 0 \\ 0 & \alpha & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{\alpha}{5} & \frac{\alpha}{5} & \frac{\alpha}{5} & \frac{\alpha}{5} & \frac{\alpha}{5} \end{pmatrix} + \begin{pmatrix} \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{1-\alpha}{5} & \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{\alpha}{2} + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{\alpha}{2} + \frac{1-\alpha}{5} \\ \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix}$$



PageRank (Page, Brin 1998)

Page Rank = Centralité de vecteur propre dans le cas orienté :



$$P = \begin{pmatrix} \frac{1-\alpha}{5} & \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{\alpha}{2} + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{\alpha}{2} + \frac{1-\alpha}{5} \\ \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1-\alpha}{5} & \alpha + \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} & \frac{1-\alpha}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix}$$

Notation matricielle : $R = \frac{1}{\lambda} P^T R$

PageRank (Page, Brin 1998)

- Résolution à l'aide de l'algorithme de puissance itérée (comme pour la centralité de vecteur propre dans le cas non orienté)

- Facteur de zap (dumping factor) : Google a choisi $\alpha = 0.85$.
  Bon compromis pour une convergence rapide.
  Exemple avec 322 millions de liens : converge en 52 itérations.