Google PageRank

다음 소프트

검색기술부

전희원

2006.10.16

PageRank 정의

- **PageRank thesis**
  - **A page is important if it is pointed to by other important page**
  - **Inlink**
  - **OutLink**
  - 확률과 통계에 의한 **PageRank Vector**의 반복계산

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

Basic PageRank Algorithm

Original PageRank Formula

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

Unknown

- **r(P_i)**
  - 페이지 **P_i** 를 가르키는 모든 페이지의 **PageRank**의 합
- **B_p**
  - **P_i**를 가르키는 페이지의 집합
- **|P_j|**
  - **P_j**로부터의 **Outlink**의 갯수
- **r(P_j)**
  - 페이지 **P_j**를 가르키는 모든 페이지의 **PageRank**의 합

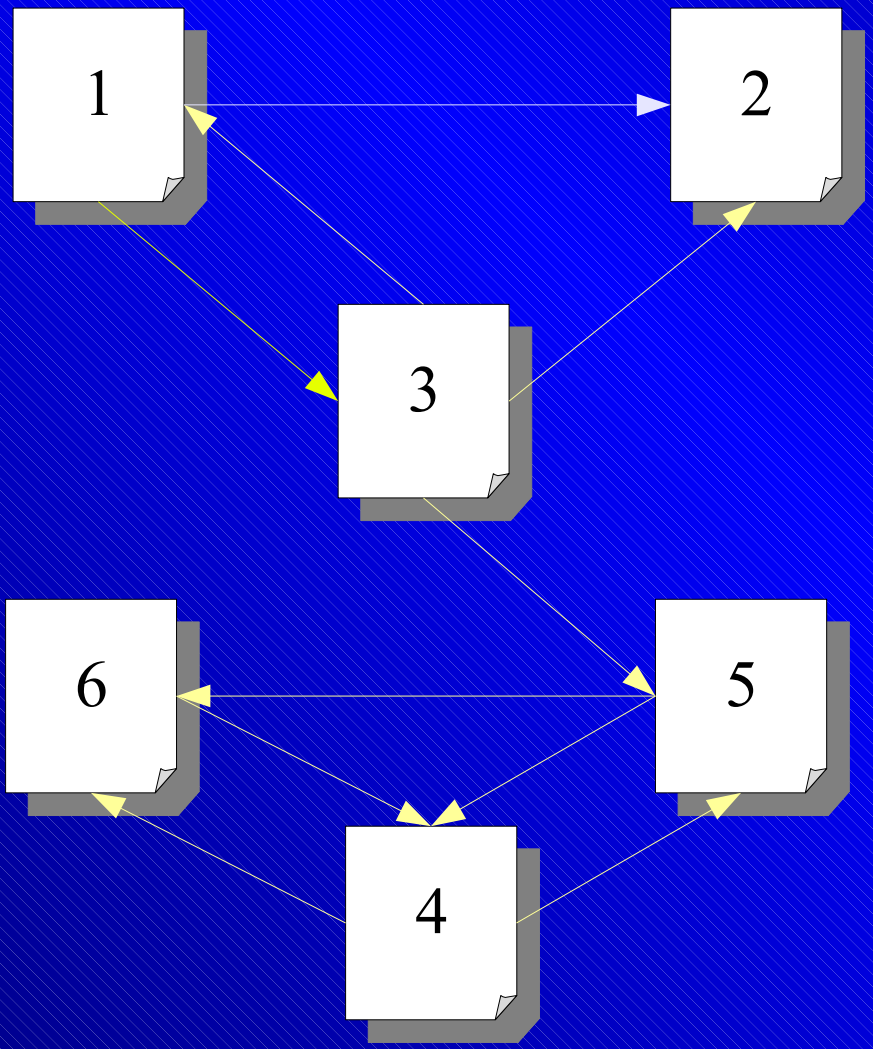
Iterative procedure

- 가정
  - 모든 페이지의 초기 **PageRank**는 **1/n** 이다.
    - **n : number of page int the world**
  - **Iteration** 방법으로 해결
    - **R_k+1(P_i) : P_i**의 **K+1**번째 **Iteration**
    - 따라서 **R_0(P_i) = 1 / n**

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

Iterative PageRank Algorithm

예제 (1)



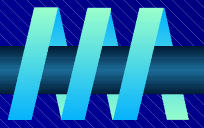
- 옛날에 전세계의 웹페이지가 **6**개였다**.**
- 이것들의 **PageRank**를 생각해보자

예제 (2)

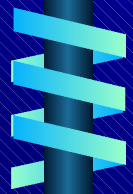
- 2번째까지의 **Iteration** 예

Iteration0	Iteration1	Iteration2	Rank at Iter 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

식을 더 직관적으로



- 그래프로 표현
  - 자료구조 책을 참고하자면 그래프 표현법은**?**
    - 인접행렬**(adjacency matrices)**
    - 인접리스트**(adjacency list)**
    - 인접다중리스트**(adjacency multilist)**
  - 직관적인 방법
    - 메트릭스로 표현
    - **Sparse matrix** 계산의 경우 인접리스트가 유리
    - 하지만 여기서는 고려하지 않겠음 **^^;**



예제의 Matrix 표현

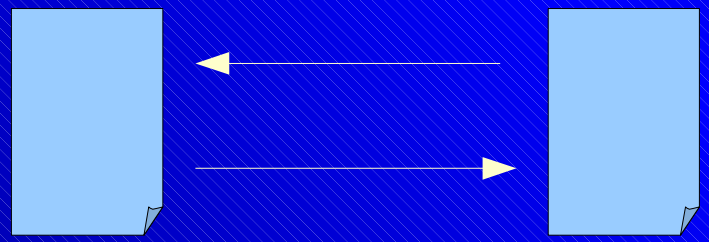
$$H = \begin{array}{c} \\ P1 \\ P2 \\ P3 \\ P4 \\ P5 \\ P6 \end{array} \begin{array}{c} \begin{array}{cccccc} P1 & P2 & P3 & P4 & P5 & P6 \end{array} \\ \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

- "∏" is n column Vector
- Alias PagerankVector

$$\prod^{(k+1)T} = \prod^{(k)T} H$$

이슈사항

- **Iteration**은 언제까지 인가?
- 수렴의 보장?
- ∏ 초기값의 영향은 얼마나되나?
- **Iteration** 횟수는 예상할 수 있나?
- **RankSink, Positive Value** 문제
  - 예제의 13번째의 ∏값
    - **(0, 0, 0, 2/3, 1/3, 1/5)**
- Cycle 문제



또 한번의 개선(1) - Makov Chain

- **Makov Chain**
  - **H**는 **Transition probability matrix**의 특징을 가지고 있다.
  - **Makov Chain**이 만족하는 조건을 충족 시키면 **Sink, Cycle** 문제를 해결할 수 있다.
    - **Stochastic**
    - **Irreducible**
    - **Aperiodic**
  - 그래서 **H**에 대한 보정 작업이 필요하다.
- **Brin** 과 **Page**가 제안한 수정모델
  - **Random surfer**
    - **Dangling node and teleport**
  - **H**의 변형모델인 **Stochastic matrix** 제안

또 한번의 개선(2) – stochastic matrix

- **S = H + a*((1/n)*e^t)**
  - a벡터에서 (a_i = 1 If i == 단말노드 else a_i = 0)

$$S = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

updated

또 한번의 개선(3) – Google Matrix

- G = α * S + (1 – α) 1/n * e*e^T
- α의 의미는?
  - α = 0.6일때?

- PageRank method의 결과

$$\prod^{(k+1)T} = \prod^{(k)T} G$$

Example(1)

- 앞 페이지의 예제를 예로 들면

$$\boldsymbol{G} = .9\cdot\boldsymbol{H}+\left(.9\cdot\begin{pmatrix}0\\1\\0\\0\\0\\0\end{pmatrix}+.1\cdot\begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix}\right)\cdot 1/6\cdot(1\,1\,1\,1\,1\,1)$$

$$G=\begin{bmatrix}1/61 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60\\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6\\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60\\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15\\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15\\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60\end{bmatrix}$$

Example(2)

- ∏^T = (.03721 .05396 .04151 .3751 .206 .2863)
  - Ranked to (6 4 5 1 3 2)
  - ∏_1 = 0.03721 의 의미
    - 3.721 % 의 시간을 Page1을 보는데 쓴다.

- 프로그래머를 위한 식 정리
  - ∏^(k+1)T = α * (∏^(k)T) * H + (α (∏^(k)T) a + 1 - α)e^T/n
  - 구현 예제
    - Http://www.freesearch.pe.kr/545

여러 부연설명

- **α**
  - **Page 와 Brin의 논문에 의하면 0.85**
  - 대략 **Iteration : 50**
  - **α^50 = (0.85)^50 ≈ 0.000296 (2~3 places of accuracy)**
  - 약 **3**억 페이지에서 실험한 결과
  - **10^(-10)** 정도의 정확도를 계산
- **Vector – Matrix** 계산 복잡도
  - **O(n^2)**
  - **Sparse Matrix (**평균 **10**개의 링크일 경우**)**
    - **O(nnz(H)) -> O(n) -> O(10n)**
  - **Other Iteration Method**
    - **GMRES, BICGSTAB**
      - 저장공간의 많아야 한다.
    - 구글 식에는 **(∏, H, a, PageRank Vector : 2)**

끝

- **Google PageRank and Beyond**
- **Next Read**
  - Combating Web Spam with TrustRank
    - http://www.vldb.org/conf/2004/RS15P3.PDF

- 다음 세미나 주제?
  - **HIT** 랭킹 알고리즘
  - **TrustRank** 알고리즘

수고하셨습니다.