LC029 정보검색

2022년 12월 1일

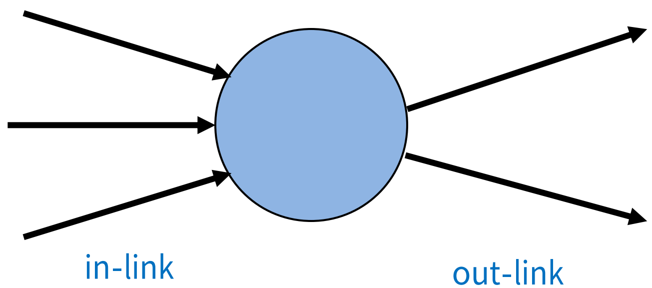
Chapter 21 : Link Analysis

Link Analysis

- Citation Analysis 피인용
  - 피인용 횟수로 논문의 권위 평가
  - The number of citations is an indicative of authority.
  - Quantify the influence of papers by analyzing the pattern of citations amongst them.
- Link Analysis for web search 이 점수는 static함
  - Useful for ranking web search results. 검색 결과 랭킹에 도움
  - Useful to determine what page(s) to crawl next.
    다음 크롤링 대상 선정에 도움

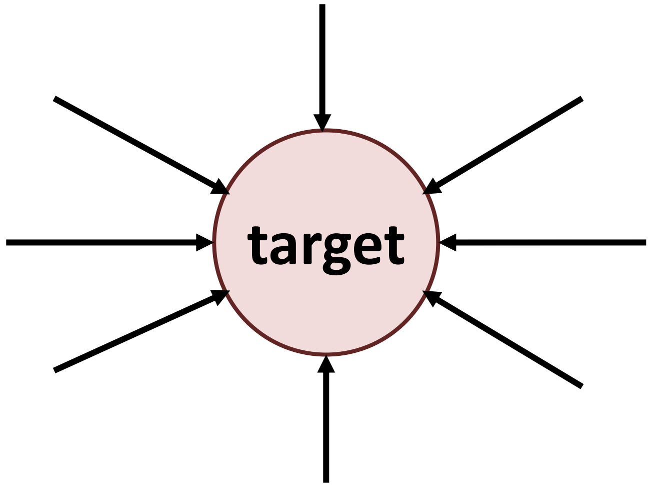
Link Analysis

- Simple Approach
  - 간단하게 링크 개수로 점수 매기기
  - Use link counts as simple measures of popularity.
  - Undirected popularity: 무방향 점수 : in + out
    - score = the number of in-links and out-links (3+2=5).
  - Directed popularity: 방향 점수 : in-link
    - Score = number of in-links (3).



Link Analysis

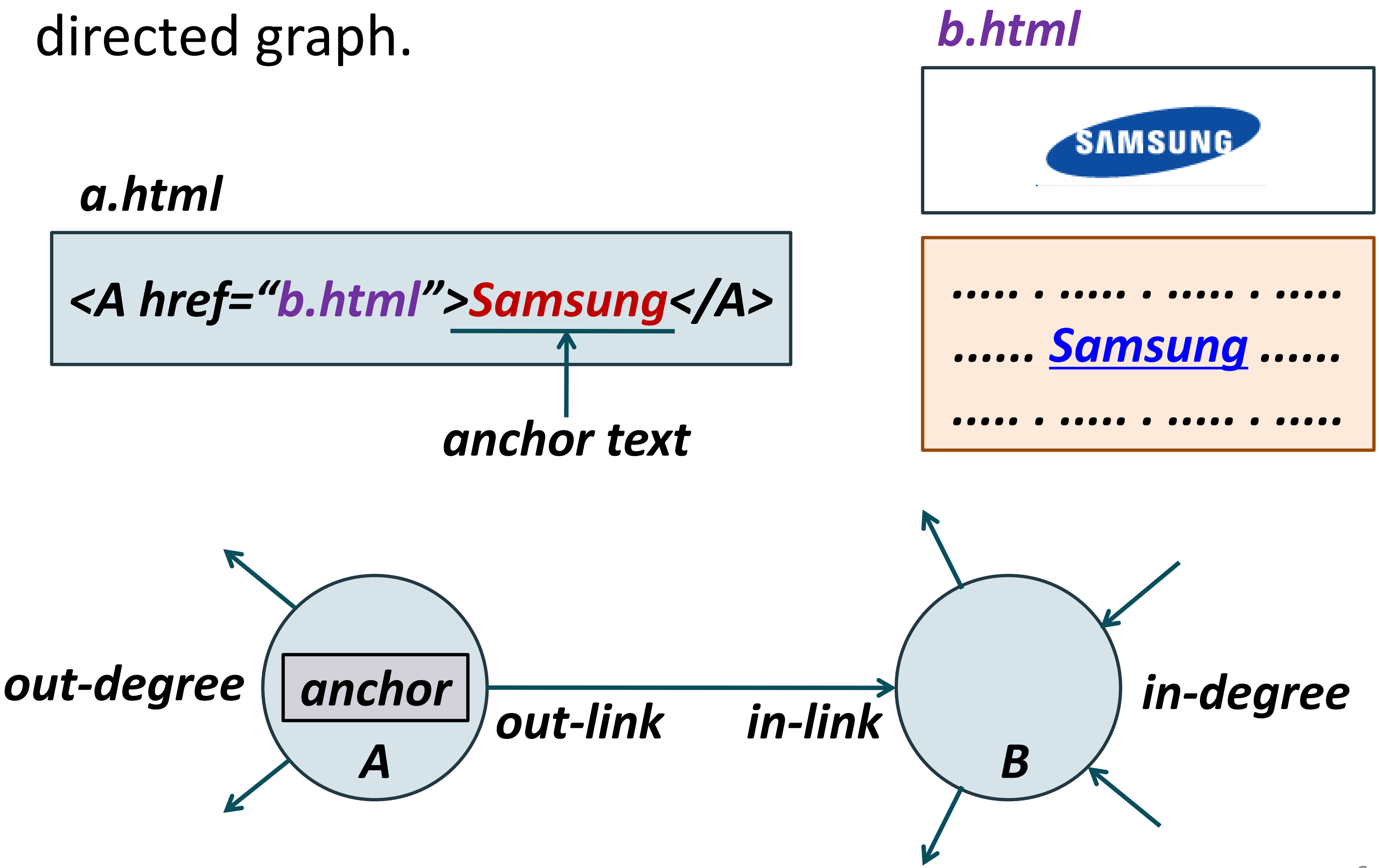
- Query Processing 쿼리 처리
  - Retrieve all pages satisfying the query. 쿼리를 만족하는 모든 페이지 반환
  - Order retrieved documents by their link popularity. 링크 점수 순으로 랭킹
- Problem : Link Spam 악용 : in-link 늘리기
  - Set up multiple web pages pointing to a target web page.



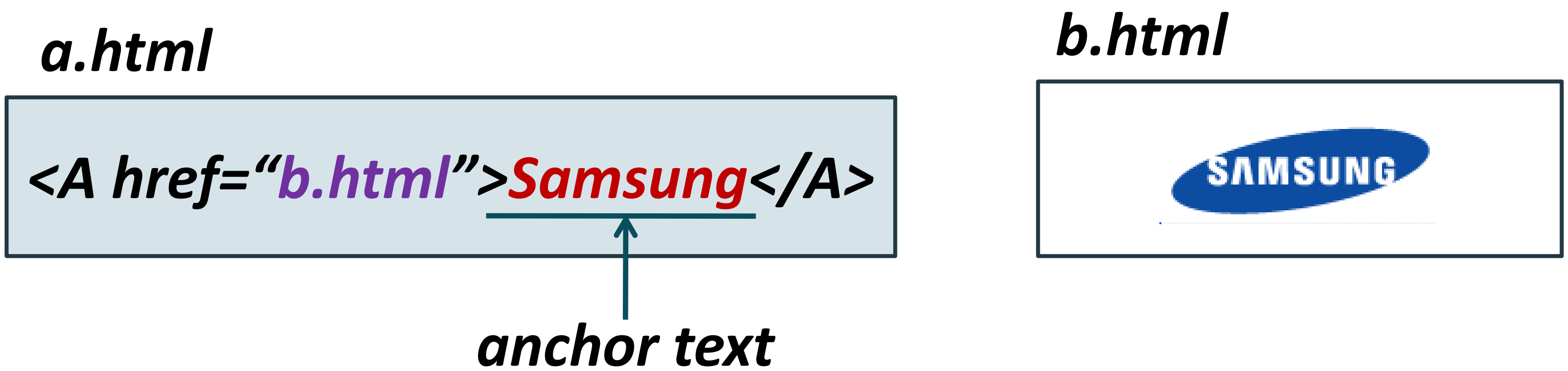
Web as a Graph

Web Graph

- Web documents have hyperlinks between them as a directed graph.



The Web as a Directed Graph



Assumption 1: The **anchor text** of the hyperlink describes the target page (textual context).

Assumption 2: A hyperlink between pages denotes author perceived relevance (quality signal).

가정 1 : 앵커 텍스트는 타겟 페이지와 관련이 있다
가정 2 : 페이지 사이의 하이퍼링크는 작성자가 인식한 관련성(품질 신호)을 나타냅니다.

Anchor Text

- A million pieces of anchor text with “IBM” send a strong signal. 특정 페이지를 가리키는 앵커 텍스트에 같은 단어가 수없이 쓰임

<A>IBM</A> <A>ibm</A> <A>IBM home page</A>

www.ibm.com

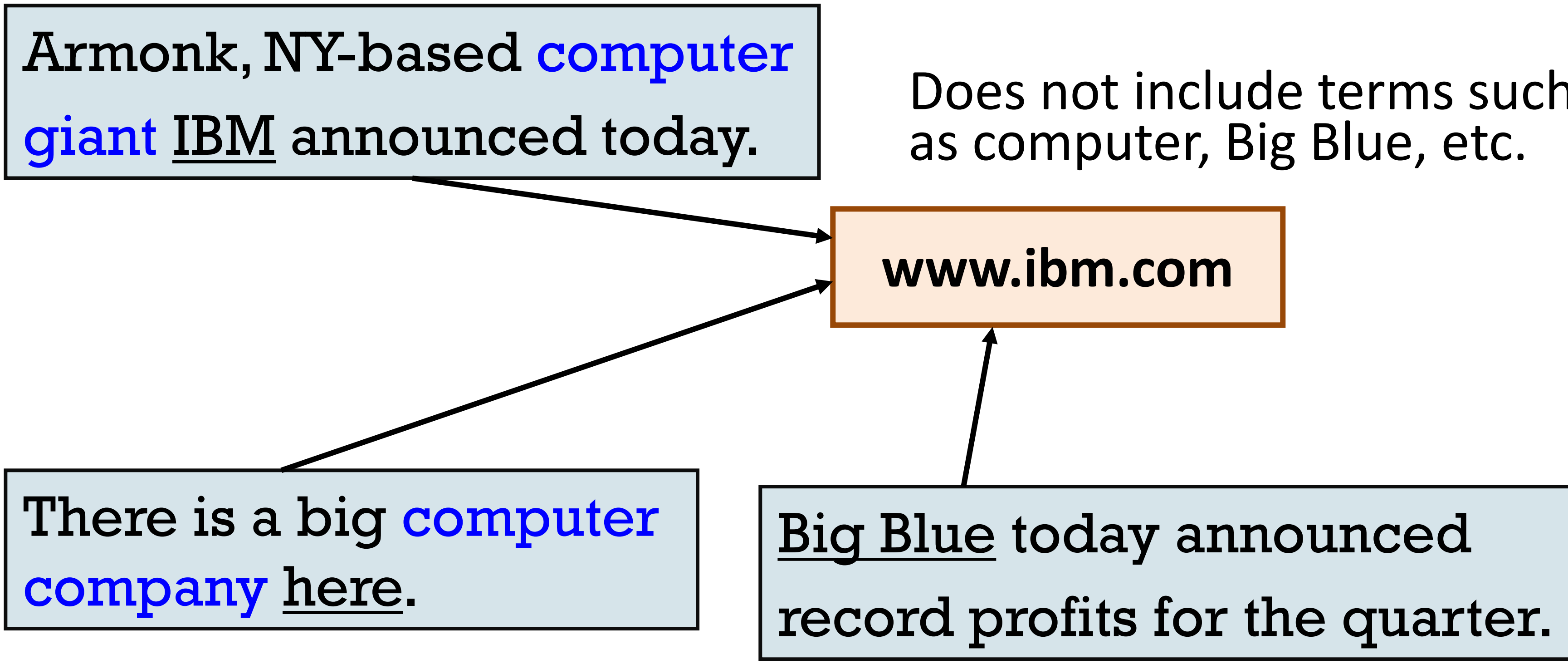
- For **IBM** how to distinguish between:
  - IBM’s home page
  - Rival’s spam page

가능성은 두 가지
- 그 단어와 진짜 관련이 있음
- 경쟁사가 가짜 링크를 만들어 악의적인 정보를 노출하고자 함

이 문제를 해결하는 방법은 배우지 않지만 주의하라고 알려주는 거다

Anchor Text

- When indexing a document *D*, include anchor text or text surrounding anchor text.



Anchor Text

www.ibm.com

There is a big computer company here.

Armonk, NY-based computer giant IBM announced today.

Big Blue today announced record profits for the quarter.

term 1 → …	www.ibm.com	…
term 2 → …	www.ibm.com	…
term n → …	www.ibm.com	…
computer company → …	www.ibm.com	…
computer giant → …	www.ibm.com	…
big blue → …	www.ibm.com	…

외부 하이퍼링크의 앵커 텍스트를 인덱스에 추가해
실제로는 해당 페이지에 없는 단어로 검색해도
그 페이지를 찾을 수 있게 함

Indexing Anchor Text

- Score anchor text with weight 앵커 텍스트 가중치 매기기
  - <u>Depending on the authority</u> of the anchor page's website
    e.g. if we were to assume that content from **cnn.com** or
    yahoo.com is authoritative, then trust the anchor text from them 페이지의 권위 참고하기
  - <u>Based on frequency</u> with a penalty for terms that occur very often 너무 자주 등장하는 단어에는 페널티 주기
    e.g. penalty for terms such as **click**, **here**, …

PageRank Scoring

PageRank Scoring

- PageRank
  - Score between 0 and 1, determined by link analysis.
  - The score of a page will depend on the link structure of the web graph. 한 링크의 랭크를 매기기 위해 해당 링크의 연결된 구조 전체를 본다
  - The score is a query-independent measure of the static quality of each web page. 이 점수는 쿼리와 독립적이고 static함
  - The score is combined with other scores such as cosine similarity, term proximity, etc. 링크 점수를 다른 점수와 합친다
  - The composite score is used to provide a ranked list of results for the query.
  - PageRank is used in Google, but so are many other clever heuristics.

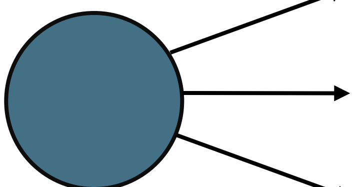
PageRank Scoring

웹 페이지에서 무작위로 걷는 서퍼를 상상해 보세요

- Imagine a surfer doing a random walk on web pages:
  - Start at a random page. 랜덤 페이지에서 시작
  - At each step, go out of the current page along one of the out-links on that page, equiprobably.

각 단계에서 해당 페이지의 아웃링크 중 하나를 따라
현재 페이지 밖으로 나가십시오.

각 단계마다 지금 페이지에 연결된 out-link 중 하나를
무작위로 골라 이동



"정상 상태"에서는 각 페이지의 방문률이 장기적입니다.
→ 이 비율은 PageRank로 사용됩니다.

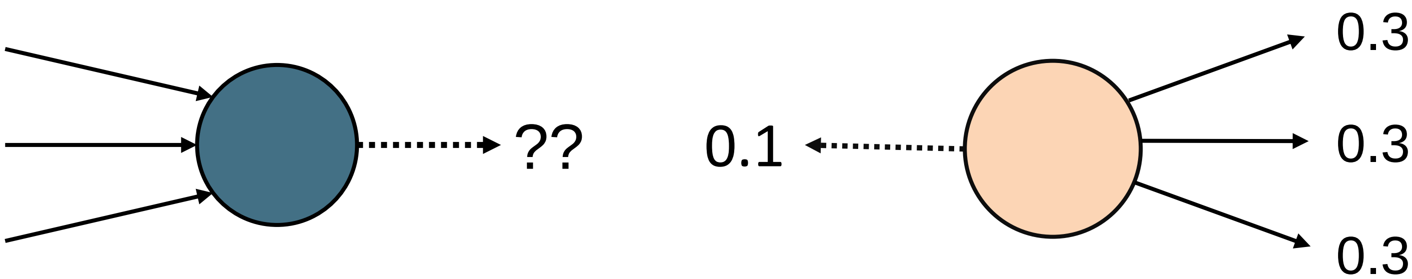
  - "In the steady state", each page has a long-term visit rate.
    → This rate is used as the **PageRank**.

위 과정을 꾸준히 아주 오랫동안 하면 각 페이지의 방문 비율을 알 수 있고, 그게 페이지 랭크다

링크가 적게 된 사이트는 그럴 이유가 있는 거다(가치가 낮다)

Teleport Operation

- The web is full of dead-ends. in만 있고 out이 없는 페이지가 있다
  - Random walk can get stuck in dead-ends.

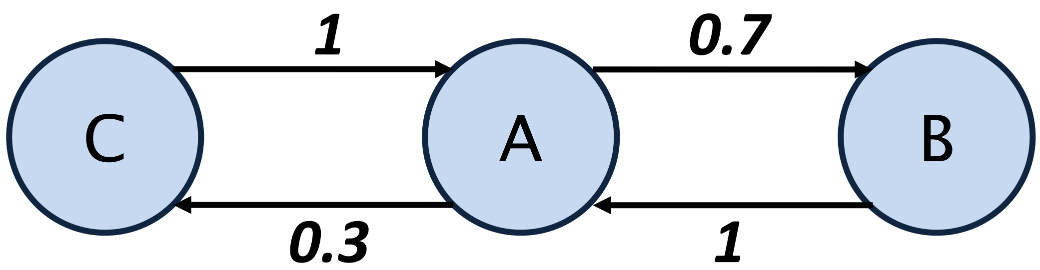


- At a **dead end**, jump to a random web page.
  - Now cannot get stuck locally. 막다른 길에 오면 랜덤 페이지로 점프
- At any **non-dead end**, with probability α, jump to a random web page. out이 있어도 누르기 싫으면 랜덤 페이지로 점프할 수 있다
  - With remaining probability (1- α), go out on a random link.
  - Here α is a parameter.

Markov Chains

Markov Chain

- Consist of *N* **states**, with **transition probabilities** between the states. 상태 간 전환 확률이 있는 N개의 상태로 구성됩니다.



- The Markov chain can be in one of the *N* states at any given **time step**. 마르코프 연쇄는 임의의 시간 단계에서 N개의 상태 중 하나에 있을 수 있다.
- The probability distribution of next state <u>depends only on the current state</u>, not on how the Markov chain arrived at the current state.

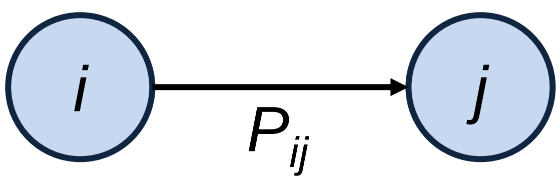
다음 상태의 확률 분포는 마르코프 연쇄가 어떻게 현재 상태에 도달했는지가 아니라 현재 상태에만 의존한다.
: 현재 상태에 오기까지의 경로에 대한 확률은 고려하지 않는다

Markov Chain

- A Markov chain is characterized by an $N \times N$ **transition probability matrix *P***, each of whose entries belongs to [0..1]. 마르코프 연쇄는 각 항목이 [0..1]에 속하는 N x N 전이 확률 행렬 P로 특징지어진다.

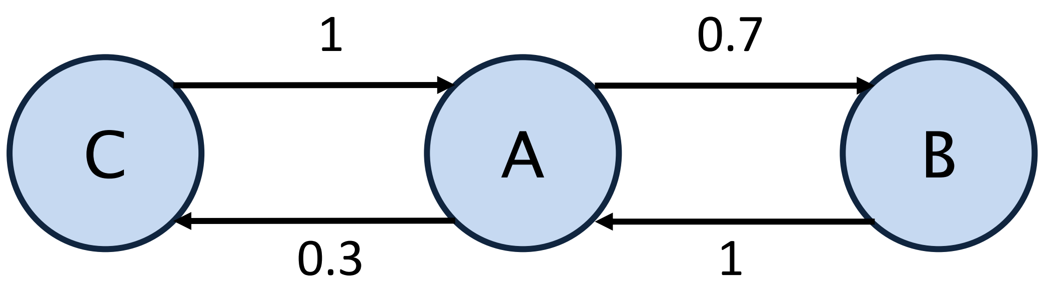
- For $1 \leq i, j \leq n$, the matrix entry P_{ij} tells us the probability of *j* being the next state, given we are currently in state *i*.

1 ≤ i, j ≤ n의 경우, 행렬 엔트리 P_ij는 우리가 현재 상태 i에 있을 때 j가 다음 상태일 확률을 알려준다.



Markov Chain

- Markov Chain



- Transition **Probability Matrix *P***

	A	B	C	
A	***0***	***0.7***	***0.3***	***= 1.0***
B	***1***	***0***	***0***	
C	***1***	***0***	***0***	

$$\forall i, \sum_{j=1}^{N} P_{ij} = 1$$

Probability Vector *of state* A

= (*0, 0.7, 0.3*)

which tells us where we go next from state A

Probability Vector

- A probability vector $\boldsymbol{x} = (x_1, \ldots, x_N)$ tells us where we are at any point.
  e.g. (0, 0, 0, …, 1, …, 0, 0, 0) means we're in state *i*.
  1 *i* *N*

- More generally, the probability vector $\boldsymbol{x} = (x_1, \ldots, x_N)$ means that we are in state *i* with probability x_i.
  e.g. (0.1, 0.3, 0.2, 0, 0.1, 0.3, 0)

  Note that $\sum_{i=1}^{n} x_i = 1.$

한 줄의 합은 항상 1이어야 함

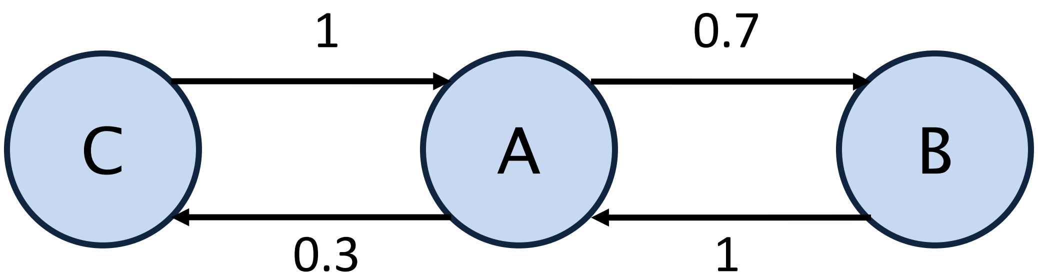
벡터를 이용해 현재 상태도 나타낼 수 있고
다음 순간 있을 상태도 나타낼 수 있음

Change in Probability Vector

- If the probability vector is $\boldsymbol{x} = (x_1, \ldots, x_N)$ at this step, what is it at the next step?
- Recall that row *i* of the transition probability matrix $\boldsymbol{P}$ tells us where we go next from state *i*.
- So, from $\boldsymbol{x}$, our next state is distributed as $\boldsymbol{xP}$.

현재 상태가 x일 때 다음 상태는 전이 확률 행렬에 곱해서 구하면 된다.

Change in Probability Vector



	A	B	C
A	***0***	***0.7***	***0.3***
B	***1***	***0***	***0***
C	***1***	***0***	***0***

transition probability matrix *P*

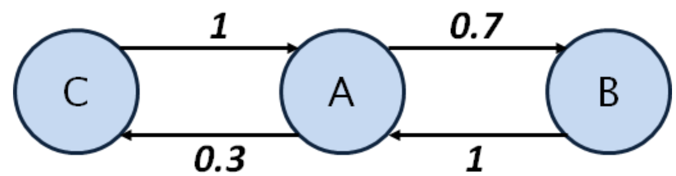
- Let ***x*** = (1, 0, 0). (We are in the state A.)
- Our next state is distributed as ***xP***.

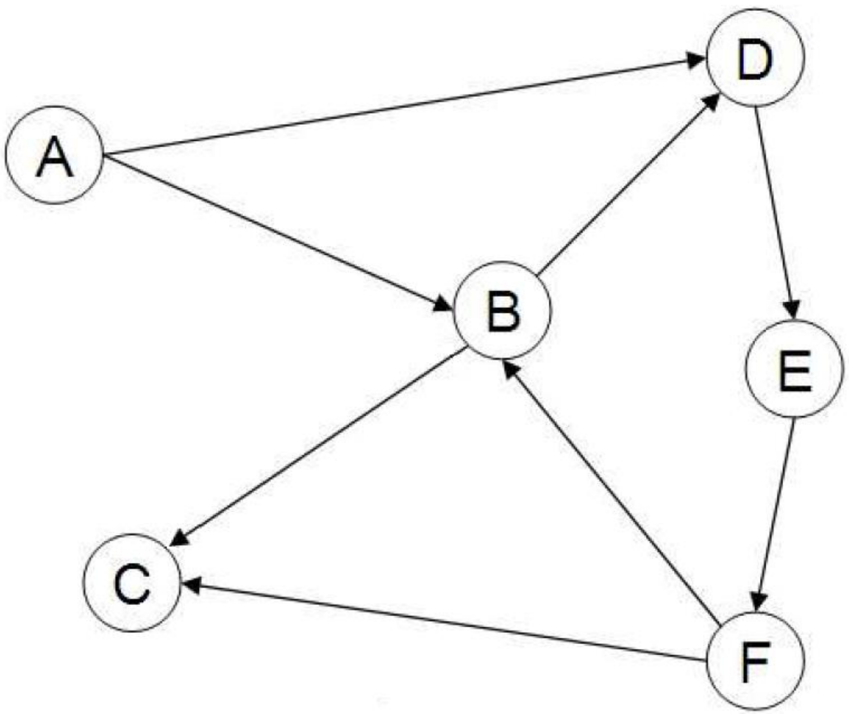
$$(1, 0, 0)\begin{pmatrix} 0 & 0.7 & 0.3 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} = (0, 0.7, 0.3)$$

현재 상태가 x일 때 다음 상태는 전이 확률 행렬에 곱해서 구하면 된다.

Markov Chain

- We will use the Markov chain to represent the web.
  - Each web page corresponds to a state in the Markov chain.



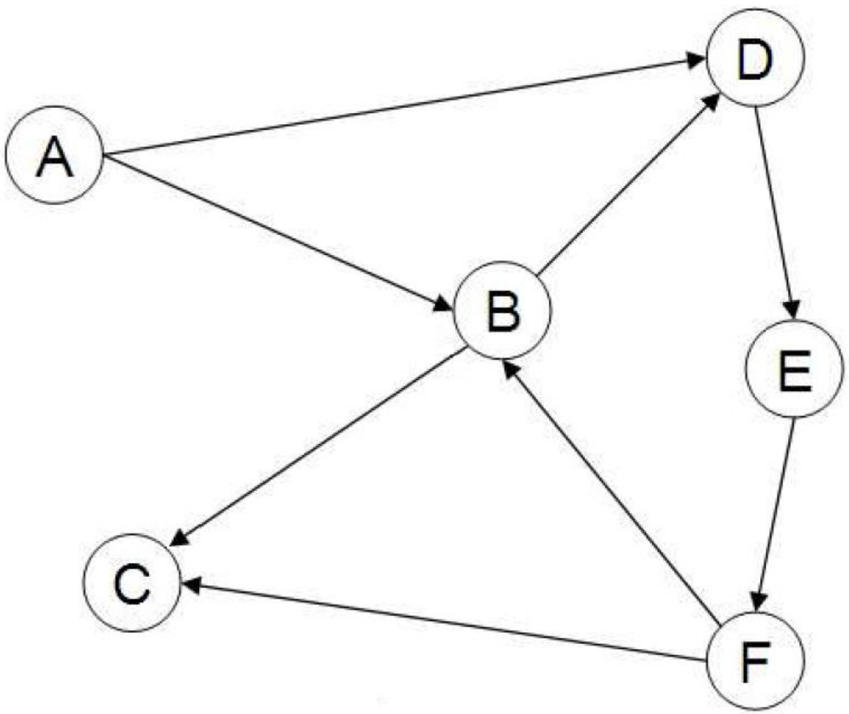


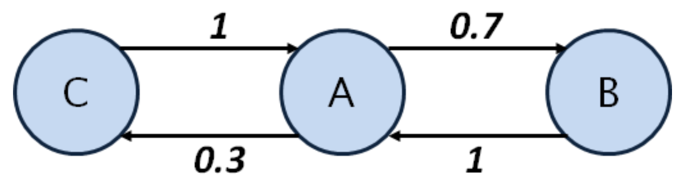
생긴 건 마르코프 체인과 비슷하지만
각 간선을 탈 확률이 없고
경로와 무관하지도 않기 때문에
마르코프 체인은 아님

← ***Web Graph***

Markov Chain

- We will use the Markov chain to represent the web.
  - If we start at page *i*, where are we at next step?





A	B	C
0	*0.7*	*0.3*
1	*0*	*0*
1	*0*	*0*

transition probability matrix *P*

← ***Web Graph***

Derivation of *P*

- Let ***A*** be an adjacency matrix of the web graph:

  A_{ij} = 1 if there is a hyperlink from page i to page j.

  A_{ij} = 0 otherwise. 링크의 유무만 표시한 인접 행렬 만들기

- Derivation of the transition probability matrix ***P*** from ***A***:
  - If a row of ***A*** has no 1's, replace each element by 1/*N*.

    만약 한 행에 1이 아예 없다면 1/N으로 바꿈

    Teleport operation

  - Otherwise, 1이 있는 행이라면

    각 행의 1에 1의 개수로 나눈 값을 대입 후 랜덤 점프 확률 곱합

    1. Divide each 1 in ***A*** by the number of 1's in its row.

    2. Multiply the resulting matrix by 1 - α.

    Random Surf

    3. Add α/*N* to every entry of the resulting matrix.

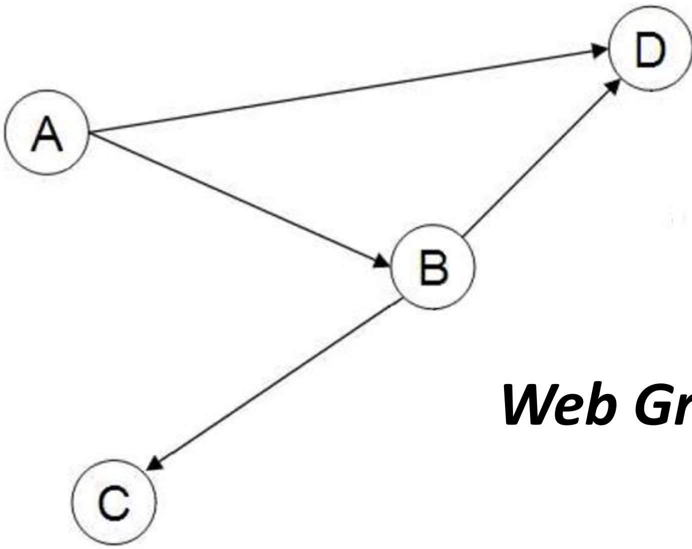
    랜덤 점프 확률을 1/N 해서 모든 행에 더함

    Teleport operation

- The resulting matrix is ***P***.

  위 과정을 모두 거치면 마르코프 체인이 됨

Derivation of ***P***



Web Graph

adjacency matrix

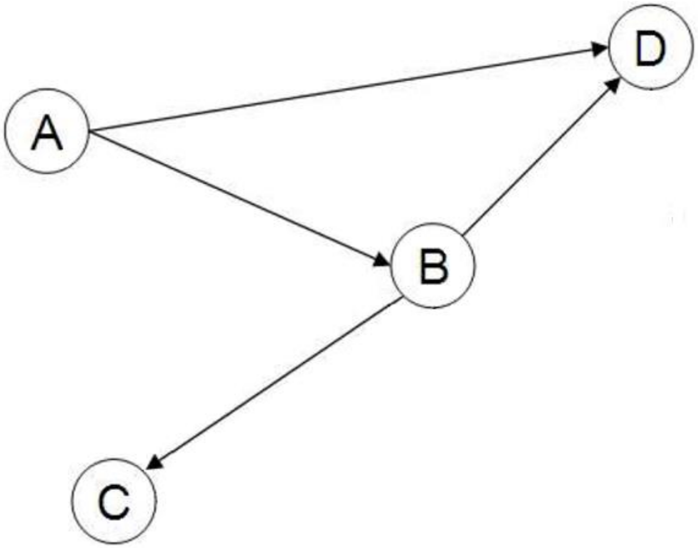
	A	B	C	D
A	*0*	*1*	*0*	*1*
B	*0*	*0*	*1*	*1*
C	*0*	*0*	*0*	*0*
D	*0*	*0*	*0*	*0*

	A	B	C	D
A	*0*	*0.45*	*0*	*0.45*
B	*0*	*0*	*0.45*	*0.45*
C	*0.25*	*0.25*	*0.25*	*0.25*
D	*0.25*	*0.25*	*0.25*	*0.25*

	A	B	C	D
A	*.025*	*.475*	*.025*	*.475*
B	*.025*	*.025*	*.475*	*.475*
C	*.250*	*.250*	*.250*	*.250*
D	*.250*	*.250*	*.250*	*.250*

transition probability matrix

Derivation of ***P***



	A	B	C	D
A	*.025*	*.475*	*.025*	*.475*
B	*.025*	*.025*	*.475*	*.475*
C	*.250*	*.250*	*.250*	*.250*
D	*.250*	*.250*	*.250*	*.250*

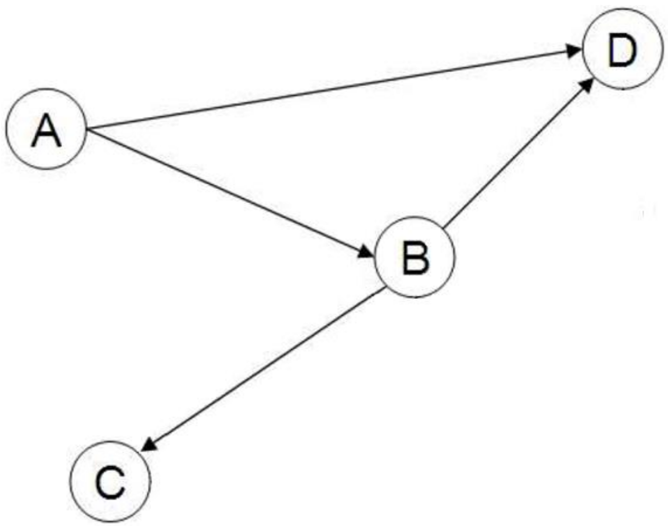
transition probability matrix

- With the given web graph, there is no path from C or D to any other node. 원래 웹 그래프에는 C, D에서 다른 곳으로 이동할 수 있는 길이 없지만
- The transition probability matrix, however, has transition probability > 0 from C or D to any other node.
  (We applied the Teleport Operation.)

랜덤 점프를 포함했기 때문에 마르코프 체인에서는 모든 페이지가 0 이상의 값을 갖는다

Ergodic Markov Chain

- A Markov chain is **ergodic** if 만약 체인에 0이 하나도 없으면
  - You have a path from any state to any other.
  - For any start state, after a finite transient time T_0, the probability of being in any state at a fixed time t ($> T_0$) is nonzero. 임의의 시작 상태에서, 유한 과도 시간 T_0 이후에, 고정 시간 t (> T_0)에서 임의의 상태에 있을 확률은 0이 아니다.



	A	B	C	D
A	.025	.475	.025	.475
B	.025	.025	.475	.475
C	.250	.250	.250	.250
D	.250	.250	.250	.250

Ergodic Markov Chain

- For any ergodic Markov chain, there is a unique **long-term visit rate** for each state.
  - *Steady-state Probability Distribution*.
- Over a long time-period, we visit each state in proportion to this rate.
- It doesn't matter where we start.
- This *steady-state probability* for a state is the **PageRank** of the corresponding web page.

0이 없는 마르코프 체인에서는
장기적으로 각 페이지에 대한 방문 확률이
전이 행렬대로 수렴한다
고로 이것을 페이지 랭크로 사용한다

0이 있으면 수렴하지 않는다

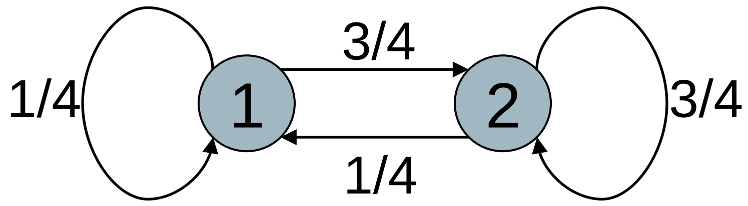
Steady State Probability Vector

- The **steady state probability vector** is a vector of probabilities $\boldsymbol{a} = (a_1, \ldots, a_N)$ where a_i is the probability that we are in state i.
- If our current position is described by $\boldsymbol{a}$, then the next step is distributed as $\boldsymbol{aP}$ where P is a Transition Probability Matrix.
- But $\boldsymbol{a}$ is the steady state, so $\boldsymbol{a} = \boldsymbol{aP}$.
- Solving this matrix equation gives us $\boldsymbol{a}$.

수렴한 상태의 각 페이지에 있을 확률 벡터 a가 있고 지금 어딘가의 페이지에 있을 때
다음 상태를 알려면 현재 상태 벡터(=a)를 전이 행렬에 곱해야 하는데
이미 수렴했기 때문에 곱해봐야 자기 자신이 그대로 나온다
다음 페이지에서 이 a를 구하고자 한다

How do we compute this vector?

- Consider the Markov chain below.



	1	2
1	***1/4***	***3/4***
2	***1/4***	***3/4***

Transition Probability Matrix P

1) Let ***a*** = (*x*, *y*) where x + y = 1.0
2) Since ***a*** = ***aP***, (*x*, *y*) = (*x/4*+*y/4*, 3*x/4*+3*y/4*).

- From (1) and (2), x = 1/4, y = 3/4.
- Therefore, ***a*** = (*1/4, 3/4*).

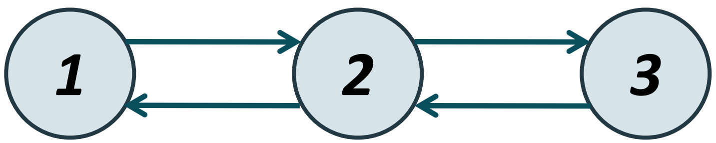
How do we compute this vector?

- **Power Iteration Method**
  - Recall, regardless of where we start, we eventually reach the steady state $\boldsymbol{a}$.
  - Start with any distribution, say $\boldsymbol{x} = (1, 0, \ldots, 0)$.
  - After one step, we're at $\boldsymbol{xP}$.
  - After two steps at $\boldsymbol{xP}^2$, then $\boldsymbol{xP}^3$ and so on.
  - For large k, we are at $\boldsymbol{xP}^k$ where $\boldsymbol{xP}^k \rightarrow \boldsymbol{a}$.
  - So, we can compute the steady-state probability vector with the multiplication of $\boldsymbol{x}$ by increasing powers of $\boldsymbol{P}$ until the product looks stable.

수없이 곱해서 수렴 벡터 a 찾기
- 아무데서나 시작한다
- 충분히 많이, p를 곱한다
- 충분히 많이 곱한 p는 a에 수렴한다
→ p의 변화량이 아주 적을 때 곱셈을 멈추면 됨

Computation of the Vector : Example

- Consider the web graph.



adjacency matrix

	1	2	3
1	0	1	0
2	1	0	1
3	0	1	0

- Assume α = 0.5, then α/N = 1/6.

	1	2	3
1	0	1	0
2	1/2	0	1/2
3	0	1	0

P →

	1	2	3
1	1/6	2/3	1/6
2	5/12	1/6	5/12
3	1/6	2/3	1/6

transition probability matrix

Computation of the Vector : Example

- Assume that the surfer starts in state 1, then the initial probability distribution vector $\boldsymbol{x_0} = (1, 0, 0)$.
- After one step, the distribution is

$$\boldsymbol{x_0 P} = (1, 0, 0)\begin{pmatrix} 1/6, 2/3, 1/6 \\ 5/12, 1/6, 5/12 \\ 1/6, 2/3, 1/6 \end{pmatrix} = (1/6, 2/3, 1/6) = \boldsymbol{x_1}$$

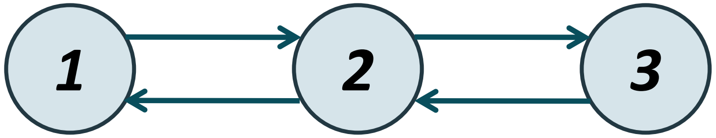
$$\boldsymbol{x_1 P} = (1/3, 1/3, 1/3) = \boldsymbol{x_2}$$

$$\boldsymbol{x_2 P} = (1/4, 1/2, 1/4) = \boldsymbol{x_3}$$

$$\boldsymbol{x_3 P} = (7/24, 5/12, 7/24) = \boldsymbol{x_4}$$

$$\ldots$$

$$\boldsymbol{x} = (5/18, 4/9, 5/18)$$



PageRank Summary

- Preprocessing: 전처리 : 수렴한 전이 확률 벡터 만들어두기
  - Given graph of links, build a Transition Probability Matrix ***P***.
  - From ***P***, compute the Steady State Probability Vector ***a***.
  - The entry a_i is a number between 0 and 1, which is the PageRank of page *i*.
  - PageRank is a *query-independent* measure of the static quality of each page.
- Query processing: 쿼리 처리 : 검색 결과 랭크에 전이 확률 벡터 점수 포함하기
  - Retrieve pages meeting query.
  - Rank them by their PageRank and other query-dependent scores such as cosine similarity, term proximity, etc.

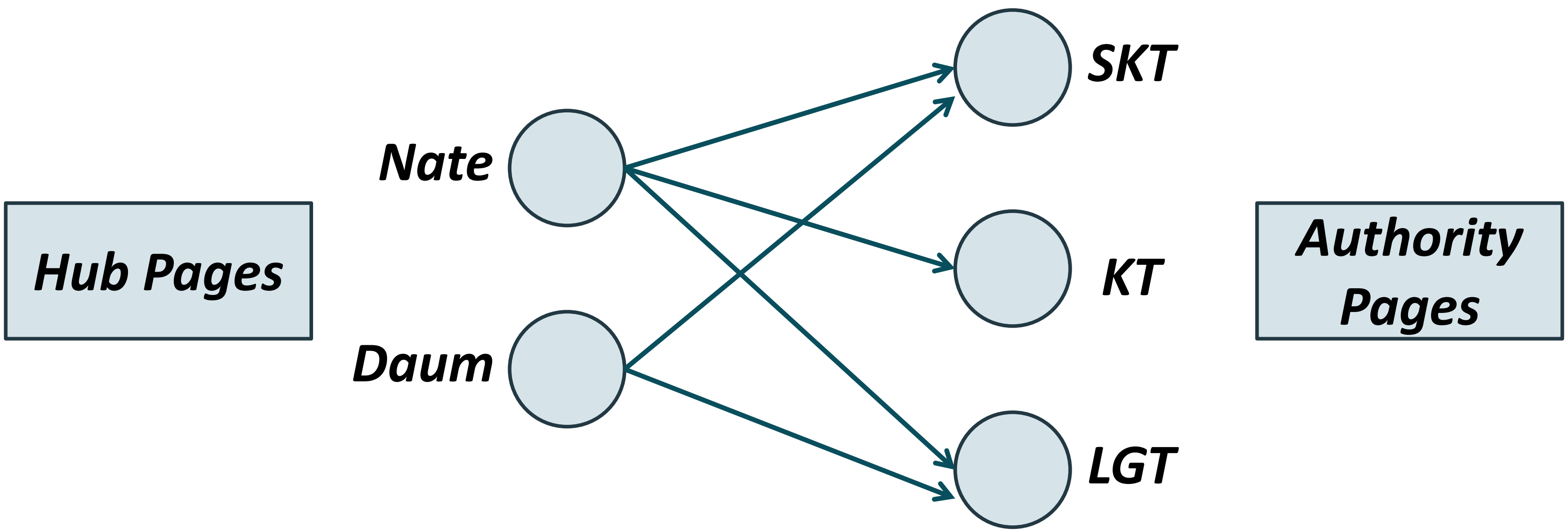
Hyperlink-Induced Topic Search

Hyperlink-Induced Topic Search (HITS)

- *In response to a query*, we'd like to find two sets of inter-related pages (instead of an ordered list of pages that match the query)
  - ***Hub pages*** 특정 부류의 페이지를 모아 나열한 페이지
    that contain good lists of links on a subject.
  - ***Authority pages*** 허브 페이지의 링크를 많이 받은 페이지
    that occur recurrently on good hubs for the subject.

Hub Pages and Authority Pages

- A good hub page for a topic *points* to many authority pages for that topic.
- A good authority page for a topic is *pointed* to by many hub pages for that topic.

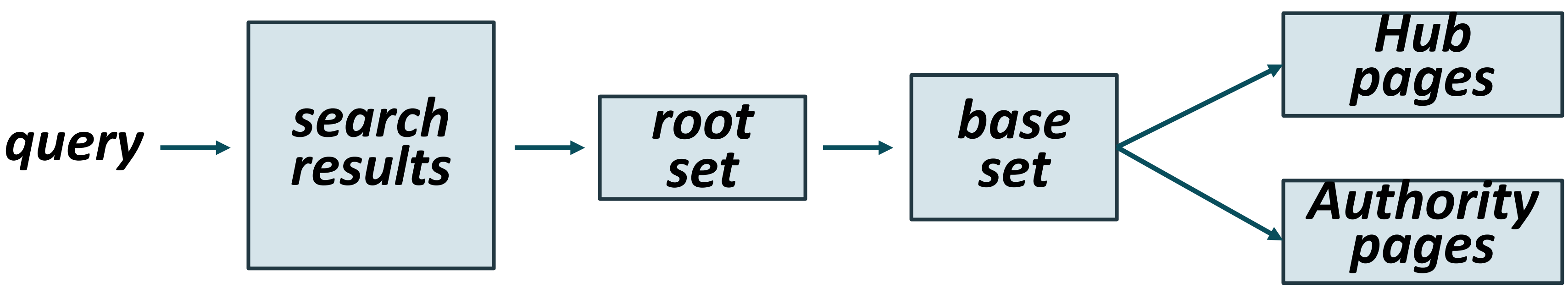


허브 페이지와 권위 페이지 찾기

How to find Hub and Authority Pages?

- Extract from the web a **base set** of pages that *could* be good hub or authority pages.
- From these, identify a small set of hub pages and authority pages.

  → use iterative algorithm



뭔가 검색함 → 검색 결과 받음 → 그 중 몇 개를 root 셋으로 함
→ root 셋을 확장해 base 셋 만듦 → 허브와 권위 페이지 찾는 데 씀

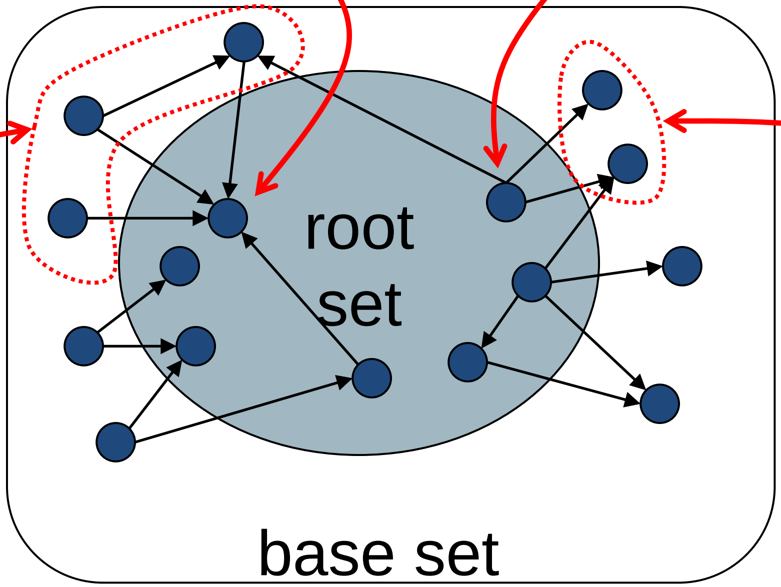
Base Set

- Given text query (say ***swine flu***), retrieve pages containing ***swine flu.*** 뭔가 검색하고 결과를 받음
  - Call them the **root set** of pages. 그 결과들을 root 셋으로 함
- Add in any page that either 다음의 조건을 만족하는 페이지를 추가함
  - points to a page in the root set, or root 셋을 가리키거나
  - is pointed to by a page in the root set. root 셋이 가리키는 것
- Call them the **base set.** 추가된 페이지를 포함해 base 셋이라 함
- Use the base set for **computing hub and authority scores**.

Base Set : Visualization

If this is good authority page

If this is good hub page



then these are good authority pages.

then these are good hub pages.

Assembling the Base Set

- **Root set** typically consists of 200-1000 pages rather than all pages matching the text query.
- **Base set** may have up to 5000 pages.
- How do you decide the base set?
  - Follow out-links by parsing root set pages (or from *connectivity server*)
  - Get in-links from a *connectivity server*

Distilling Hub and Authority Pages

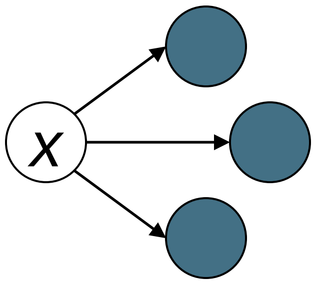
- Compute, for each page *x* in the base set, a **hub score** *h(x)* and an **authority score** *a(x)*.
  - Initialize: for all *x*, $h(x) \leftarrow 1$; $a(x) \leftarrow 1$;
  - **Iteratively update** all *h(x)*, *a(x)*;
- After iterations choose pages with
  - highest *h()* scores as hub pages
  - highest *a()* scores as authority pages

허브 점수와 권위 점수는 모두 1로 초기화
모든 페이지를 들러가면서 반복하여 허브 점수와 권위 점수 업데이트

Iterative Update

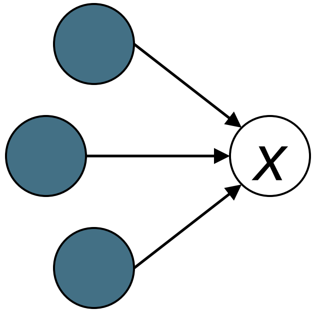
- Repeat the following updates, for all *x*:

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$



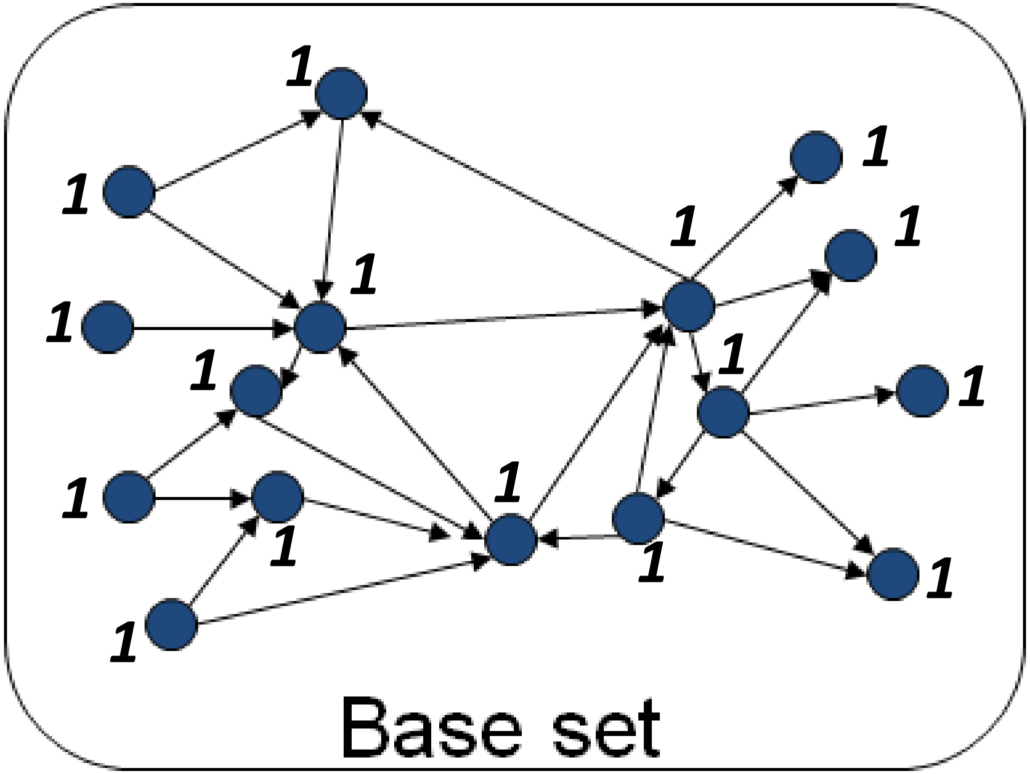
허브 점수 : 링크한 권위 페이지의 점수의 합

$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$



권위 점수 : 링크받은 허브 페이지의 점수의 합

Iterative Update



Scaling Down of the Scores

- To prevent the *h()* and *a()* values from getting too big, scale down after each iteration.
- Scaling factor doesn't matter:
  - we only care about the *relative* values of the scores.

점수가 너무 커지면 곤란하니까 매번 반복이 끝날 때마다 스케일 다운
상대적인 점수만 알면 됨

How many iterations?

- Claim
  - Relative values of scores will converge after a few iterations. 점수의 상대적인 값은 몇 번의 반복 후에 수렴됩니다.
  - We only require the relative orders of the *h()* and *a()* scores, not their absolute values.
    우리는 h()와 a()의 절대값이 아니라 상대적인 순서만 필요합니다.
- Then, how many iterations?
  - In practice, at most 5 iterations will get you fairly good results. 실제로는 최대 5회 반복하면 상당히 좋은 결과를 얻을 수 있습니다.

Some interesting insights about HITS

- Frequently, the documents that emerge as top hubs and authorities include language other than the language of the query.
- This cross-language retrieval effect resulted from <u>link analysis</u>, with no linguistic translation.

종종 상위 허브 및 권한으로 나타나는 문서에는 질의 언어 이외의 언어가 포함됩니다.
이러한 언어 간 검색 효과는 언어 번역 없이 링크 분석에서 비롯되었다.

Query: “Japan Elementary Schools”

- schools
- LINK Page-13
- “ú–{,ÌŠwZ
- a‰„¬ŠwZƒz[ƒƒy[ƒW
- 100 Schools Home Pages (English)
- K-12 from Japan 10/...rnet and Education)
- http://www...iglobe.ne.jp/~IKESAN
- ,l,f,j¬ŠwZ,U”N,P‘g•¨Œê
- ÒŠ—’¬—§ÒŠ—“Œ¬ŠwZ
- Koulutus ja oppilaitokset
- TOYODA HOMEPAGE
- Education
- Cay's Homepage(Japanese)
- –y“ì¬ŠwZ,Ìƒz[ƒƒy[ƒW
- UNIVERSITY
- ‰J—³¬ŠwZ DRAGON97-TOP
- Â‰ª¬ŠwZ,T”N,P‘gƒz[ƒƒy[ƒW
- ¶µ°é¼ÂÁ© ¥á¥Ë¥å¡¼ ¥á¥Ë¥å¡¼

Hub Pages

- The American School in Japan
- The Link Page
- ‰ªèŽ—§ˆä“c¬ŠwZƒz[ƒƒy[ƒW
- Kids' Space
- ˆÀéŽ—§ˆÀé¼•”¬ŠwZ
- ‹{é‹³ˆç‘åŠw•‘®¬ŠwZ
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- _“ÞìŒ§E‰¡•ls—§’†ì¼¬ŠwZ,Ìƒy
- http://www...p/~m_maru/index.html
- fukui haruyama-es HomePage
- Torisu primary school
- goo
- Yakumo Elementary,Hokkaido,Japan
- FUZOKU Home Page
- Kamishibun Elementary School...

Authority Pages