

Information Access and Retrieval (GBX9MO23)

Web Information Retrieval

<http://gbx9mo23.imag.fr/>

M2R – MOSIG

2021-2022

Philippe Mulhem 

Philippe.Mulhem@imag.fr

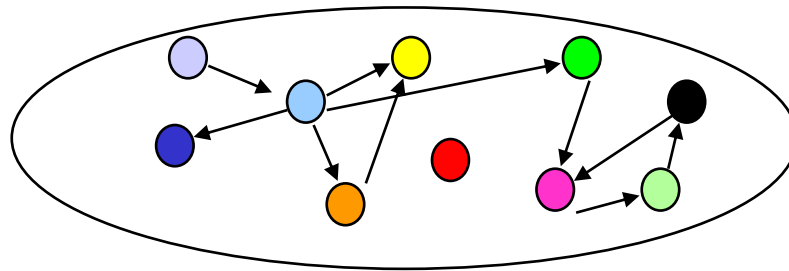
<http://lig-membres.imag.fr/mulhem/>

Outline

- The Web as a graph
- Web document access
 - Crawler/Indexing/Content Retrieval
 - Graph Usage
 - Integration of elements for Web retrieval
- Conclusion

The Web as a graph

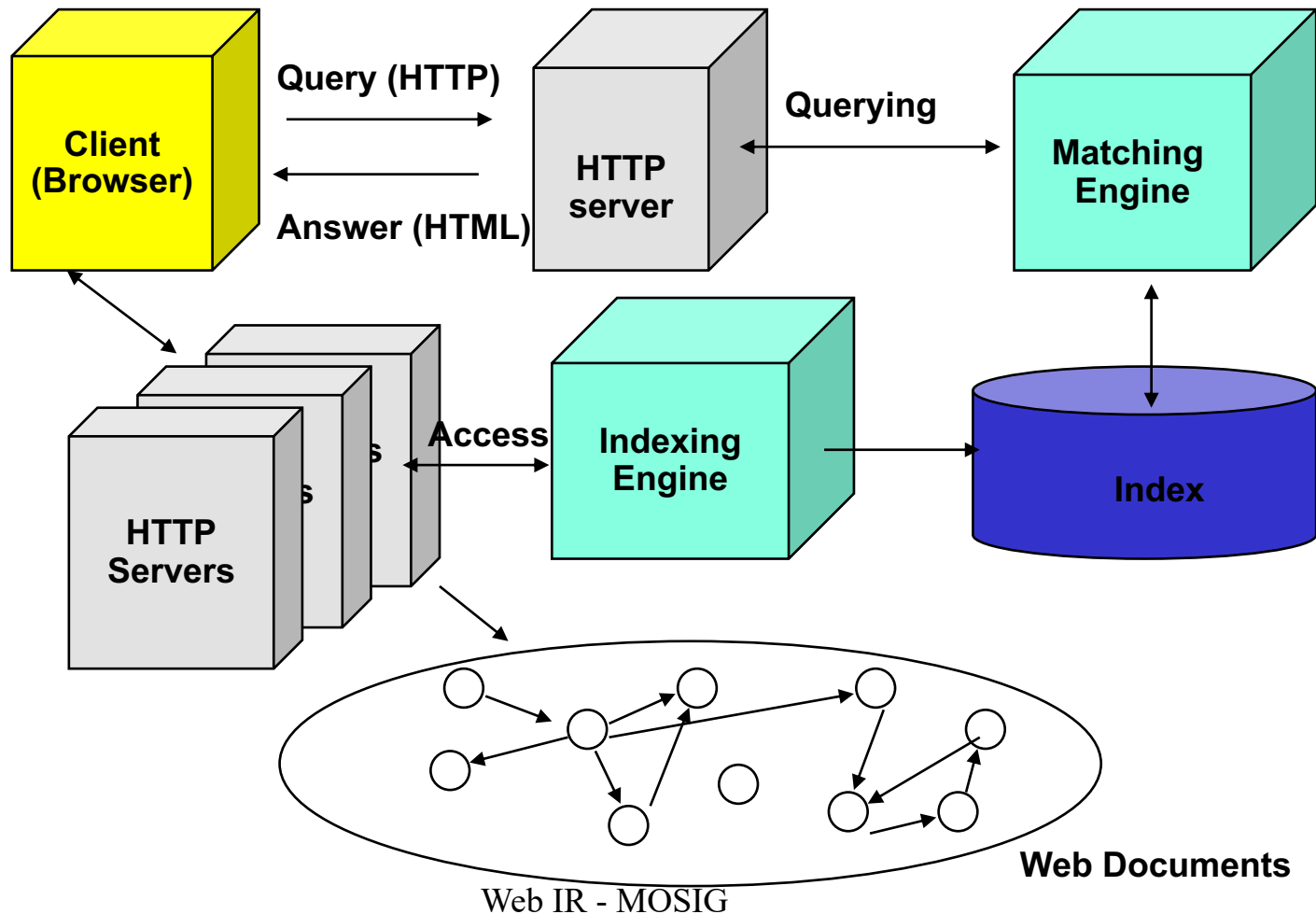
- Web: largest source of information
- Web pages: nodes
- Hypertext links: edges (meanings: structure, recommendation, similarity, reading paths ?)



- This graph structure may be used by IR

Web documents access

- Information available but not accessible... need for retrieval



Web Retrieval by queries

- General features
 - Automatic indexing
 - Robust: heterogeneity (languages, documents structures/formats)
 - Coverage (Google: 45 Billons pages)
 - Huge storage (hundreds of terabytes)
 - Freshness consideration
 - Response time (for interaction, 4 billions queries/day)
 - Multi-level indexing (content/structure)
 - Models that mix content/structure/links (boolean models with word positions, query logs, social data)

Indexing engine - Crawler

- Dynamic exploration of the web (bots)
 - Googlebot, Bingbot, Slurp (Yahoo), DuckDuckBot
- Simple crawler using a set of seeds URL named E

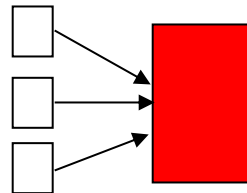
```
while E not empty do
  e ← get_one_element_of(E)
  p ← get_page_from_URL(e)
  index(p)
  E = E ∪ outlinks(p)
```
- Refinements on
 - Number of pages, Topics
 - Efficiency (wrt. overload of web servers)
- Classical inverted indexes with term position, adapted to high dimensions (terms x documents x queries)
 - Distributed indexes

Content Retrieval

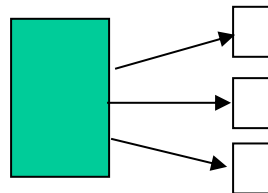
- Queries on the web are generally short (2-4 words)
 - « The best place to hide a dead body is page 2 of the Google search results. » (Brian Clark)
 - people are interested in the first result page
 - All elements that may enhance the top-10 results are used (web server features, social tags, HTML validations, ...)
 - Classical Boolean models with term position
 - AND/OR/NOT
 - Proximity search
- + 100's of other parameters (not only content)
- ⇒ like the Web graph: links between pages

Graph Usage – HITS [Kleinberg99]

- Goal: to characterize the pages retrieved after processing a query q
 - Good sources of content (Authorities, AUTH)



- Good sources of links (Hubs, HUB)

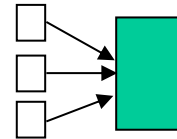


Graph Usage – HITS [Kleinberg99]

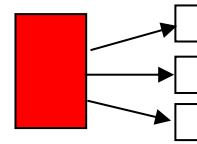
- Pages characterized by $\langle \text{HUB}; \text{AUTH} \rangle$ value

→ Intuition

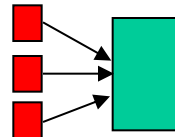
- Authority value from in-links



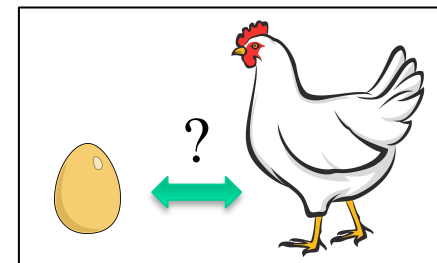
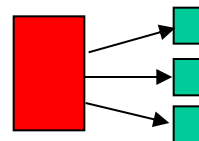
- Hub value from out-links



- High Authority if in-links from good Hubs



- High Hub if out-links to good Authorities



Graph Usage – HITS [Kleinberg99]

– Initialisation :

$$\sum_{V \text{ in result}(q)} AUTH[V]^2 = \sum_{V \text{ in result}(q)} HUB[V]^2 = 1 \quad (\text{if } N \text{ pages: } AUTH[V] = HUB[V] = \frac{1}{\sqrt{N}})$$

– Iteration (2 steps) until convergence

1. For each page V compute intermediate values (from the preceding normalized values):

$$AUTH'[V] = \sum_{Edge(W_i, V)} HUB[W_i]$$

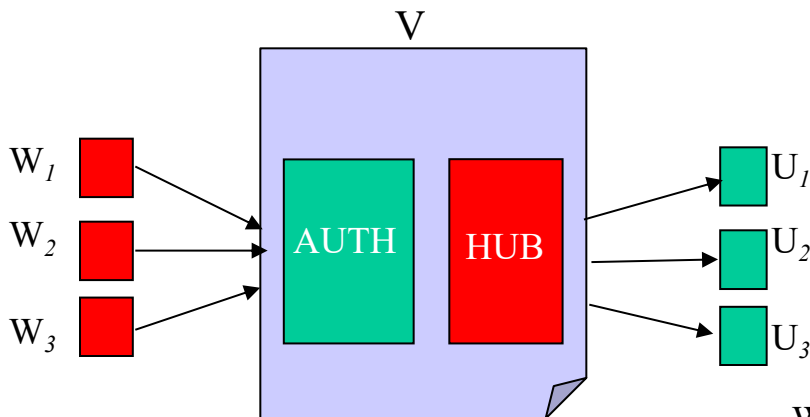
$$HUB'[V] = \sum_{Edge(V, U_i)} AUTH[U_i]$$

2. Final values after normalization:

$$AUTH[V] = \frac{AUTH'[V]}{\sqrt{\sum_U AUTH'[U]^2}}$$

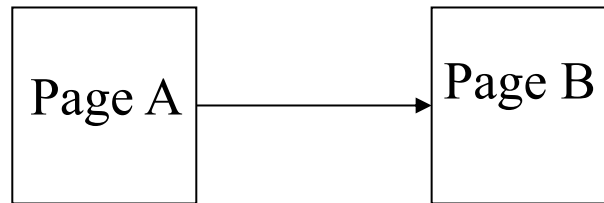
$$HUB[V] = \frac{HUB'[V]}{\sqrt{\sum_U HUB'[U]^2}}$$

so that $\sum AUTH[U]^2 = \sum HUB[U]^2 = 1$



Graph Usage – HITS [Kleinberg99]

- 2 pages:

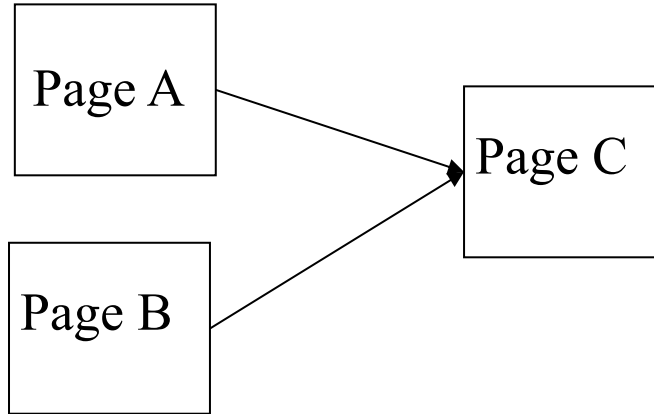


- Init : $AUTH[A] = AUTH[B] = 1 / \sqrt{2} = 0.71$ (same for HUB)
- $AUTH'[A] = 0$; $AUTH'[B] = HUB[A]$; $HUB'[A] = AUTH[B]$; $HUB'[B] = 0$
- Convergence at iteration 2: same AUTH and HUB values as in step 1.

	AUTH[A]	HUB[A]	AUTH[B]	HUB[B]	$\sqrt{\sum AUTH'[U]^2}$	$\sqrt{\sum HUB'[U]^2}$
0	0.71	0.71	0.71	0.71		
1.1	0	0.71	0.71	0	0.71	0.71
1.2	0	1	1	0		
2.1	0	1	1	0	1	1
2.2	0	1	1	0		

Graph Usage – HITS [Kleinberg99]

- 3 pages
- Init : $1/\sqrt{3}$

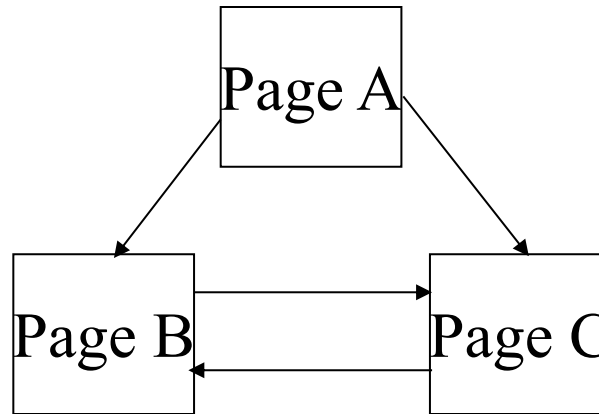


	AUTH[A]	HUB[A]	AUTH[B]	HUB[B]	AUTH[C]	HUB[C]
0						
1.1	0.58	0.58	0.58	0.58	0.58	0.58
1.2	0	0.58	0	0.58	1.15	0
2.1	0	0.71	0	0.71	1	0
2.2	0	1	0	1	1.41	0
	0	0.71	0	0.71	1	0

$\sqrt{\sum A[U]^2}$	$\sqrt{\sum H[U]^2}$
1.15	0.82
1.41	1.41

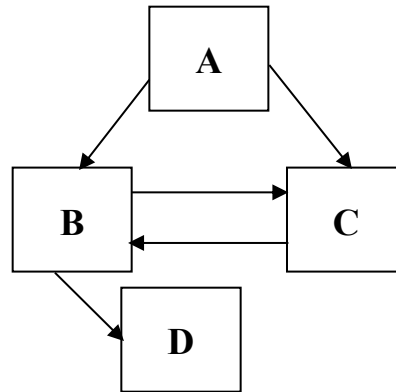
Graph Usage – HITS [Kleinberg99]

- 3 pages



	AUTH[A]	HUB[A]	AUTH[B]	HUB[B]	AUTH[C]	HUB[C]
0	0.58	0.58	0.58	0.58	0.58	0.58
1.1	0	1.15	1.15	0.58	1.15	0.58
1.2	0	0.82	0.71	0.41	0.71	0.41
2.1	0	1.41	1.22	0.71	1.22	0.71
2.2	0	0.82	0.71	0.41	0.71	0.41

- 4 pages



Note : stop when mean of differences ≤ 0.02

	AUTH[A]	HUB[A]	AUTH[B]	HUB[B]	AUTH[C]	HUB[C]	AUTH[D]	HUB[D]
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1.1	0.00	1.00	1.00	1.00	1.00	0.50	0.50	0.00
1.2	0,00	0,67	0,67	0,67	0,67	0,33	0,33	0,00
2.1	0,00	1,33	1,00	1,00	1,33	0,67	0,67	0,00
2.2	0,00	0,74	0,56	0,56	0,74	0,37	0,37	0,00
3.1	0,00	1,30	1,11	1,11	1,30	0,56	0,56	0,00
3.2	0,00	0,72	0,62	0,62	0,72	0,31	0,31	0,00
4.1	0,00	1,34	1,03	1,03	1,34	0,62	0,62	0,00
4.2	0,00	0,74	0,57	0,57	0,74	0,34	0,34	0,00
5.1	0,00	1,32	1,09	1,09	1,32	0,57	0,57	0,00
5.2	0,00	0,73	0,60	0,60	0,73	0,32	0,32	0,00

Graph Usage – HITS [Kleinberg99]

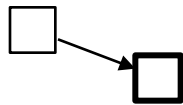
- Conclusion
 - Query post processing: Overload for each query processed
 - « Heavy » computation (top-200 pages)
 - Easy to spam with automatically generated links
- How to avoid these drawbacks ?
 - Pagerank

Graph Usage – Pagerank

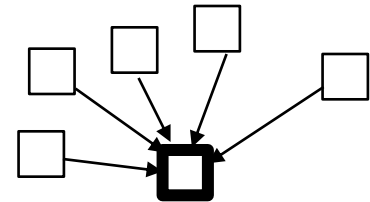
- Google [Brin & Page 1998]
 - Pagerank computes the popularity of web pages
 - +++ independently of any query
 - A Pagerank value is one positive floating point value
 - +++ simple
 - Graphically:



Low popularity



Medium popularity



High popularity

Graph Usage – Pagerank

- All web pages vote for the popularity of one Web page.
- An in-link on a page is voting « for » the page.
- No link is an « abstention » about the page
- Pagerank computes the « chances » to access one page A from any other page on the web, according A's in-links

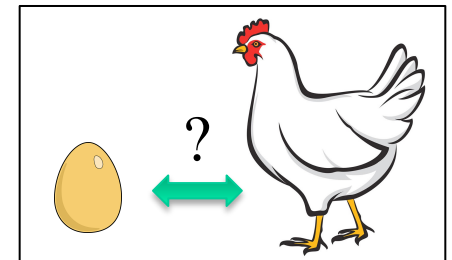
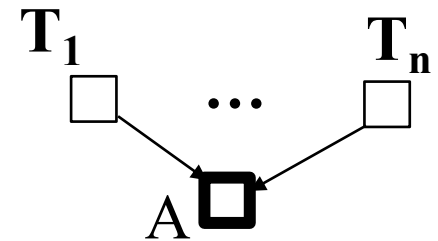
Graph Usage – Pagerank

- The Pagerank, PR, of a page A is defined as:

$$PR(A) = (1 - d) + d * \left(\frac{PR(T_1)}{C(T_1)} + \dots + \frac{PR(T_n)}{C(T_n)} \right)$$

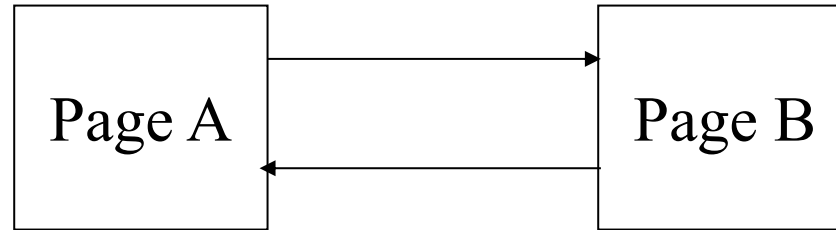
avec

- d : dumping factor in [0,1]
 - » 1-d chances to jump directly to A
 - » d chances to get to A through links
 - T_i : Web page that links to A
 - $C(T_i)$: number of out-links from T_i
 - PR in $[1-d, +\infty[$
- How to compute $PR(A)$???
- Iteration until convergence, with initial values of 1.0



Graph Usage – Pagerank

- 2 pages



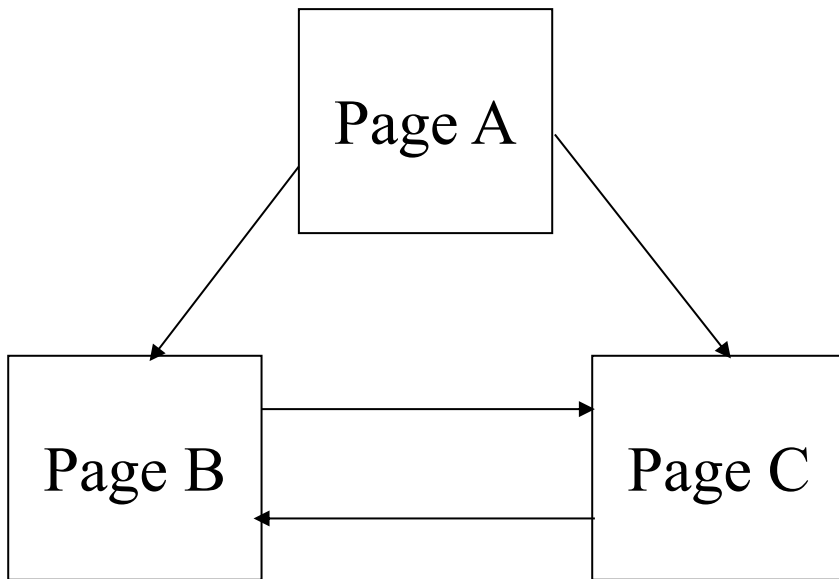
Warning: PR from the preceding iteration

- $d=0.85$, init $PR(A) = PR(B) = 1$
- $PR(A)=0.15+0.85*(PR(B)/1) = 0.15+0.85*1 = 1$
- $PR(B)=0.15+0.85*(PR(A)/1) = 0.15+0.85*1 = 1$
- Initial values do not change in this case: convergence.

Graph Usage – Pagerank

- 3 pages

- $d=0.85$, init $PR(A) = PR(B) = PR(C)=1$
- $PR(A)=0.15$
- $PR(B)=0.15+0.85*(PR(A)/2 + PR(C)/1)$
- $PR(C)=0.15+0.85*(PR(A)/2 + PR(B)/1)$

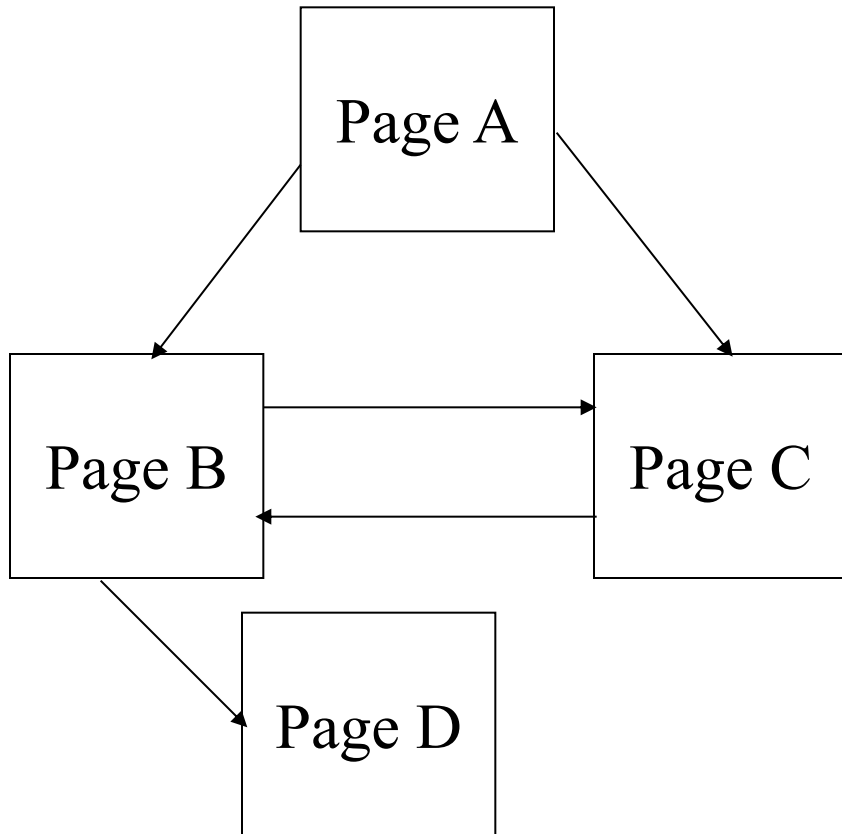


	PR(A)	PR(B)	PR(C)
0	1	1	1
1	0,15	1,425	1,425
2	0,15	1,425	1,425

Green arrows indicate the flow of values from the row 0 to the rows 1 and 2. From PR(A)=1, arrows point to PR(A)=0,15 and PR(B)=1,425. From PR(B)=1, arrows point to PR(B)=1,425 and PR(C)=1,425. From PR(C)=1, arrows point to PR(C)=1,425 and PR(B)=1,425.

Graph Usage – Pagerank

- Example

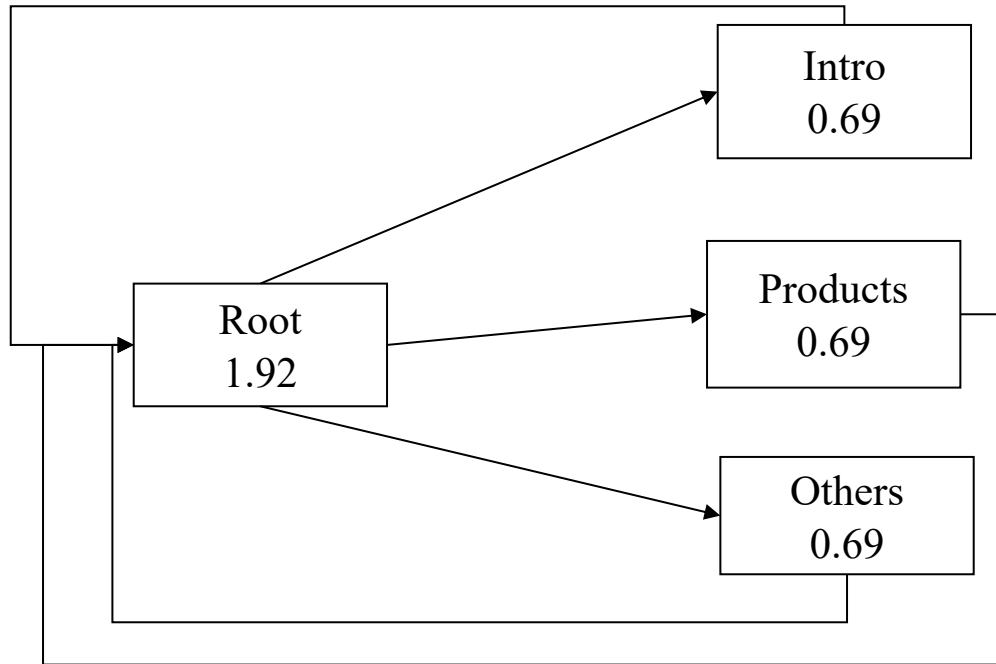


	PR(A)	PR(B)	PR(C)	PR(D)
0	1	1	1	1
1	0.15	1.425	1	0.575
2	0,15	1,06	0,82	0,76
3	0,15	0,91	0,67	0,60
4	0,15	0,78	0,60	0,54
5	0,15	0,72	0,55	0,48
6	0,15	0,68	0,52	0,46
7	0,15	0,66	0,50	0,44

convergence: stop when average of differences lower than threshold 0.02.

Graph Usage – Pagerank

- Simple hierarchy
 - Root page has larger PR value



Graph Usage – Pagerank

- Conclusion
 - + A priori computation
 - + Hard to spam
 - Complex to compute on billions of pages
 - Disadvantages novelty
 - Choice of one interpretation of links

Integration of elements for Web retrieval

- How to mix the elements related to the content (ex. BM25 matching scores) and elements related to the graph (ex. Pagerank values) ?
- We can use learning approaches:
 - We need
 - Queries
 - Features (BM25, Pagerank, ...)
 - Relevance indications
 - We learn a function that integrates these values
 - Learning To Rank (LTR)

Integration of elements for Web retrieval

LTR using Logistic Regression

- Learn the best combination of coefficients

- For an item i (a document), if we have 2 parameters $\langle \text{PageRank}, \text{LM} \rangle$, written $\langle \text{item}_{i,1}, \text{item}_{i,2} \rangle$, \Rightarrow 3 coeffs b_0, b_1 and b_2 learned for the function:

$$\text{output}(\text{item}_i) \leftarrow b_0 + b_1 \cdot \text{item}_{i,1} + b_2 \cdot \text{item}_{i,2} \quad (1)$$

- Logistic regression: $p(\text{class}=1 | \text{item}_i) = 1 / (1 + e^{-\text{output}(\text{item}_i)}) \quad (2)$
 - $\text{class}=1 \Leftrightarrow$ relevant item; $\text{class}=0 \Leftrightarrow$ non-relevant item

- Iteration (e.g. Gradient Descent) until convergence

1. output computation using current coefficients b_j
2. Re-estimate b_j 's based on the class prediction error (on i items with resp. class Y_i , j coeffs, m : learning rate parameter)

$$b_j' = b_j - m \cdot \sum_i \left((p(\text{class} = 1 | \text{item}_i) - Y_i) \cdot \text{item}_{i,j} \right) \quad (3)$$

- Decision for the class i prediction one item

- if $P(\text{class}=1 | \text{item}) > 0.5$ then decision: $\text{class}=1$; else decision: $\text{class}=0$

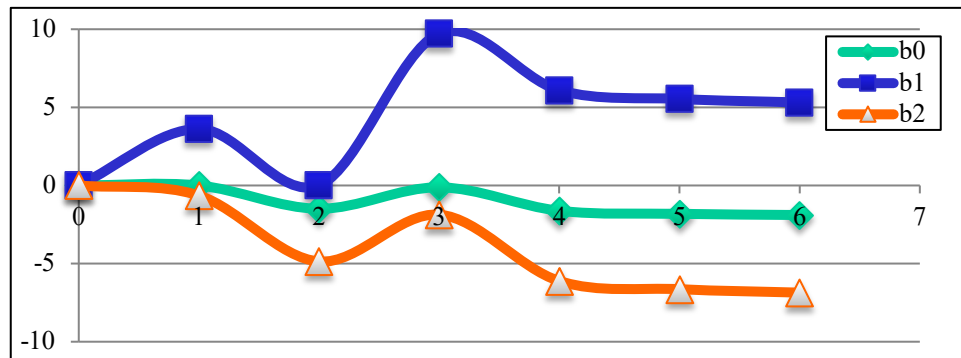
Integration of elements for Web retrieval

10 items

(Y=1) \Leftrightarrow relevant

(Y=0) \Leftrightarrow non-relevant

With $m=0.3$



Item id	Item _{,1}	Item _{,2}	Y
0	2,781	2,551	0
1	1,465	2,362	0
2	3,397	4,400	0
3	1,388	1,850	0
4	3,064	3,005	0
5	7,628	2,759	1
6	5,332	2,089	1
7	6,923	1,771	1
8	8,675	-0,242	1
9	7,674	3,509	1

Item Id	Y	P(class = 1)	decision
0	0	0,034	0
1	0	0,001	0
2	0	0,001	0
3	0	0,011	0
4	0	0,207	0
5	1	1	1
6	1	1	1
7	1	1	1
8	1	1	1
9	1	1	1

init : $b_0 = b_1 = b_2 = 0.0$

At the end of epoch 6:

epoch	b0	b1	b2
6	-1,891	5,314	-6,859

No decision error

Integration of elements for Web retrieval

- Large scale dataset
- Letor 3.0 GOV corpus [Qin, Liu, Xu, Li 2010]

⇒ 64 features

- 575 queries
- 1000 doc per query
- Relevance per query
 - 1: relevant
 - 0: not relevant

49	HITS authority	Q-D
50	HITS hub	Q-D
51	PageRank	D

21	BM25 of body	Q-D
22	BM25 of anchor	Q-D
23	BM25 of title	Q-D
24	BM25 of URL	Q-D
25	BM25 of whole document	Q-D
26	LMIR.ABS of body	Q-D
27	LMIR.ABS of anchor	Q-D
28	LMIR.ABS of title	Q-D
29	LMIR.ABS of URL	Q-D
30	LMIR.ABS of whole document	Q-D
31	LMIR.DIR of body	Q-D
32	LMIR.DIR of anchor	Q-D
33	LMIR.DIR of title	Q-D
34	LMIR.DIR of URL	Q-D
35	LMIR.DIR of whole document	Q-D
36	LMIR.JM of body	Q-D
37	LMIR.JM of anchor	Q-D
38	LMIR.JM of title	Q-D
39	LMIR.JM of URL	Q-D
40	LMIR.JM of whole document	Q-D

Integration of elements for Web Retrieval

- Many others factors
 - User data (clicks, ...)
 - Site speed, extension
 - Pages : number of backlinks, size
 - Social : refs on Twitter, Google+, ...

 - Correlation study http://www.searchmetrics.com/wp-content/uploads/Ranking_Correlations_EN_print.pdf
 - Guesses on Google <http://backlinko.com/google-ranking-factors>

Conclusion

- Short view of Web search specificities
- Learning to Rank for use of multiple features
- Research problems
 - Integration of users input (logs) to enhance results
 - Dimensionality (users, web pages, queries, languages, dynamicity)
 - Integration of content an semantics (RDF Resource Description Framwork)

Open Work

- Understand the computation of HUB/AUTH + Pagerank on the slides examples
- Understand the ideas behind Learning To Rank

References

- L. Page and S. Brin and R. Motwani and T. Winograd, **The PageRank Citation Ranking: Bringing Order to the Web**, Technical Report. Stanford InfoLab, 1999, <http://infolab.stanford.edu/~backrub/google.html>
- Jon M. Kleinberg, **Authoritative sources in a hyperlinked environment** J. ACM 46, 5, September 1999, <http://www.cs.cornell.edu/home/kleinber/auth.pdf>
- T.-Y. Liu, **Learning to Rank for Information Retrieval**, Foundations and Trends in Information Retrieval Vol. 3, No. 3 (2009): http://didawiki.di.unipi.it/lib/exe/fetch.php/magistraleinformatica/ir/ir13/1_-_learning_to_rank.pdf (part 1)
- T.-Y. Liu, J. Xu, T. Qin, W.-Y. Xiong, and H. Li, **LETOR: Benchmark dataset for research on learning to rank for information retrieval**, in SIGIR'07 Workshop on Learning to Rank for Information Retrieval, 2007, <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/08/letor3.pdf>