RuSSIR
Russian Summer School
in Information Retrieval
2008

Data structures in Information Retrieval

Language Models and Graphs

RuSSIR
Russian Summer School
in Information Retrieval
2008

Speller example

Britany Spears

Britany → Britain (LD: 2 Mod. Pr.: 5e-5)

Britany → Britney (LD: 2 Mod. Pr.: 3e-7)

Britain Spears (Pr: 7e-8)

Britney Spears (Pr: 3e-4)

Language models

Language model – probabilistic presentation of language:

- Characters
- Words
- Phrases

Language Models Applications

- Index/Search pruning
- Distribution Data in cluster/ balancing

- Spelling/ Query Suggestion
- Ranking Algorithm
- Classification/extraction/ other IR tasks

N-gram word model

$(Word_1, Word_2, \ldots Word_n) \rightarrow P$

Smoothing:

- Unseen N-grams

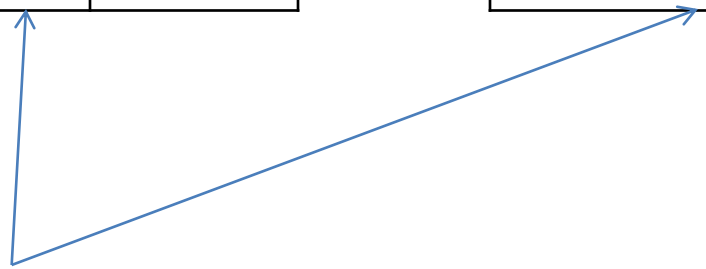
  Laplace: P = 1/(n+2)
- Outliers, lack of data

  Prior: Poisson, Dirichlet

Naïve Presentation of LM

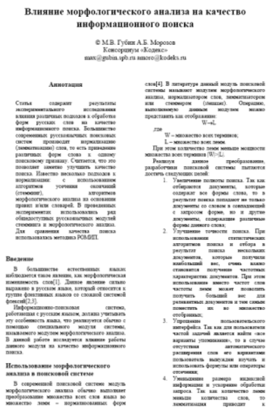
Key	Prob
Word1	0.002
Word2	0.0004
WordN	0.00043

Key	Prob
Word1, Word'1	0.000001
Word2, Word'2	0.000001
WordN, Word'N	0.00003

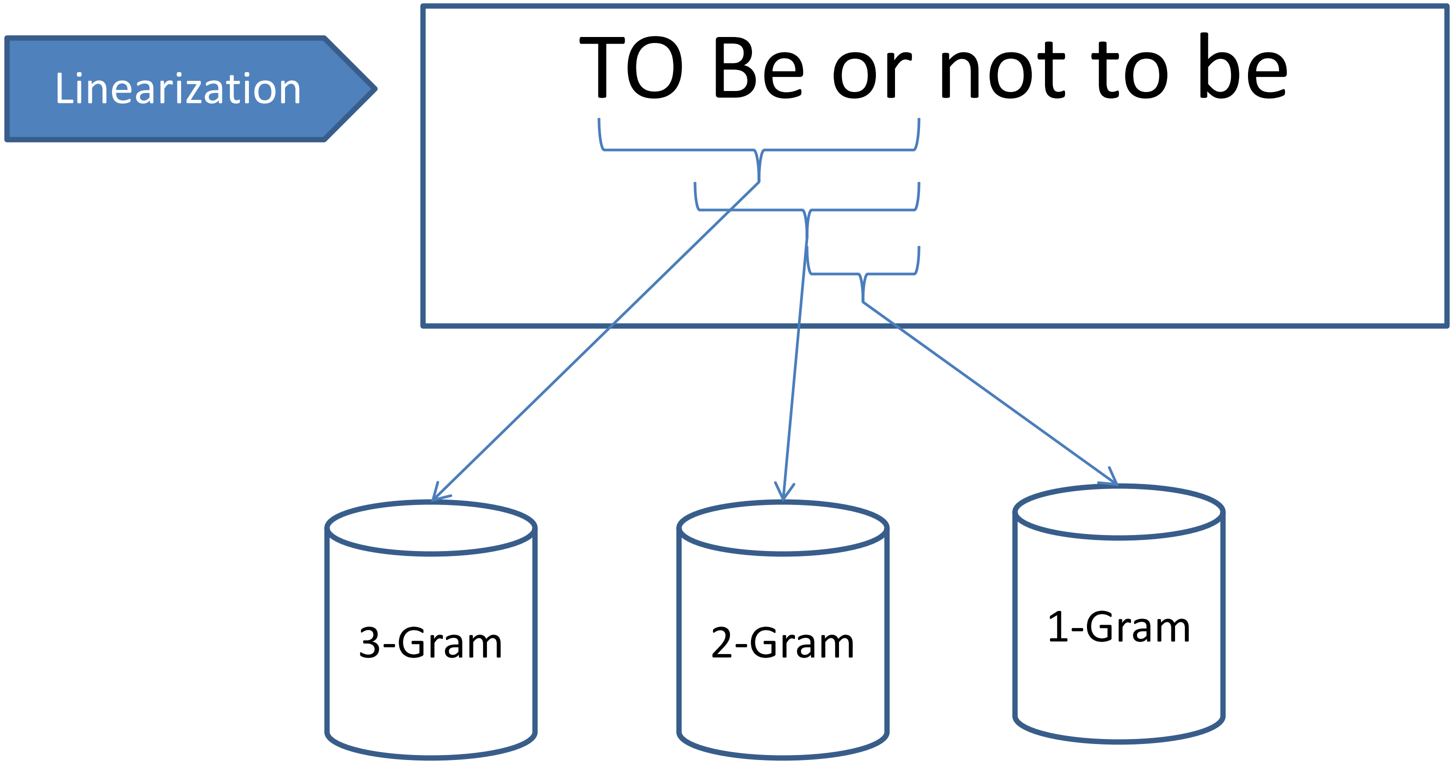


Any search structure

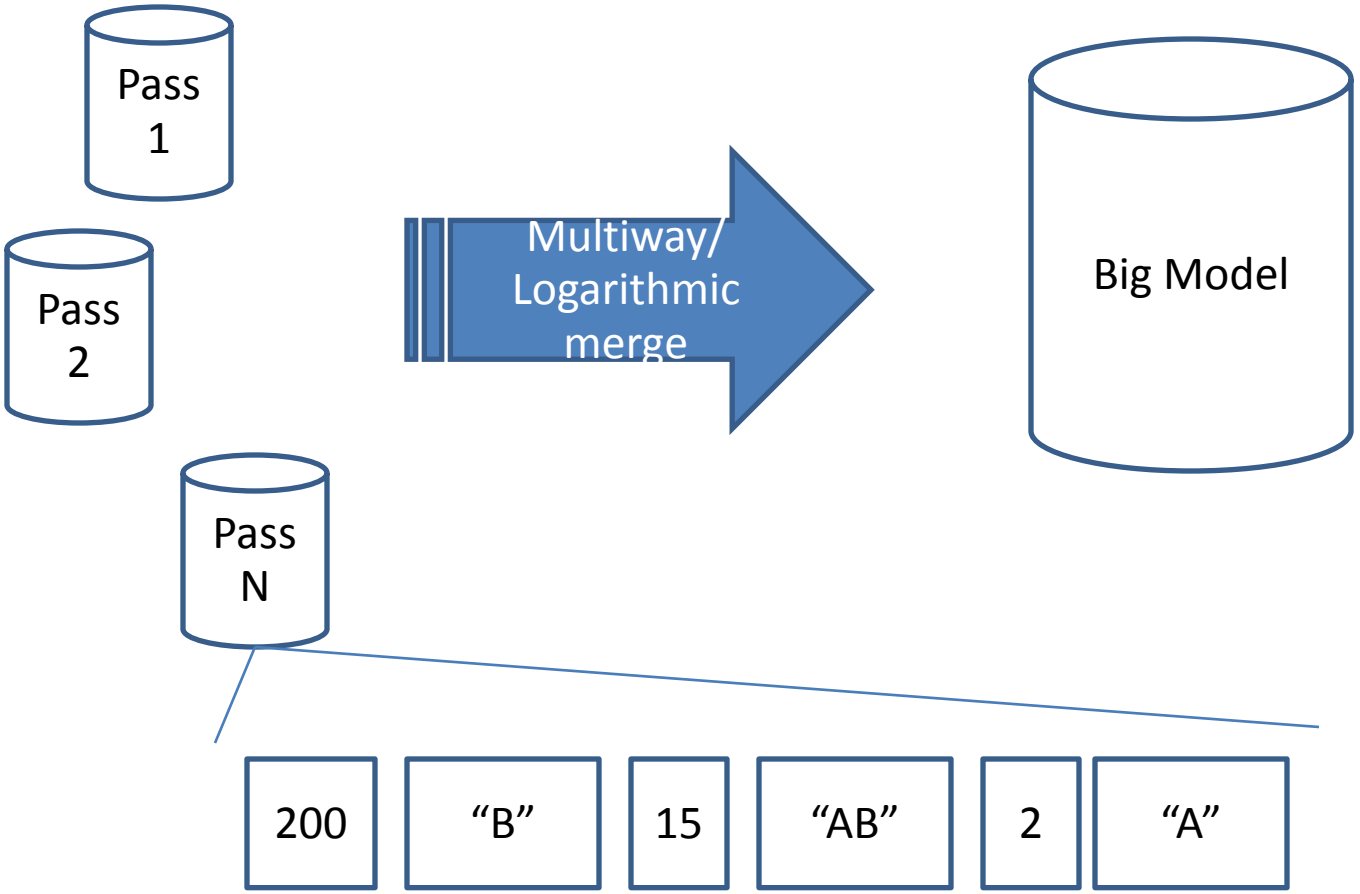
Building LM


Влияние морфологического анализа на качество информационного поиска

© М.В. Губин А.Б. Морозов
Консорциум «Кодекс»



Big LM


Pass 1
Pass 2
Pass N
Multiway/ Logarithmic merge
Big Model
200
“B”
15
“AB”
2
“A”

Presentation of Probabilities

During build: counters

During use: log(p)

$P_1 * P_2 \ldots * P_N \rightarrow \log(P_1)+\log(P_2)\ldots+\log(P_N)$

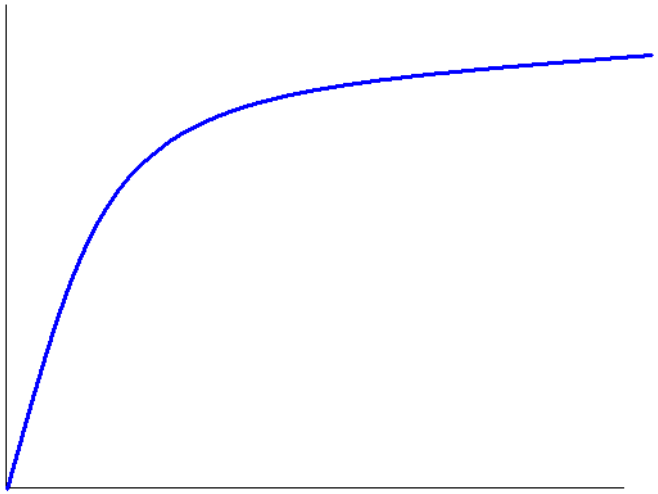
$P_1+P_2\ldots+P_N \rightarrow \log(e^{\log(P_1)}+e^{\log(P_2)}\ldots+e^{\log(P_n)})=$

$\log(P_1)+ \log(1+e^{\log(P_2)-\log(P_1)}\ldots+e^{\log(P_n)-\log(P_1)})$

Example of “super-pruned LM”

“the”=0.062 “of” = 0.03 others = 0.000028
H=0.28

“the” = 0.062 others = 0.000029 H=0.17



Pruning LMs

Task: Put into memory N-gram model

Model: N words + probability

1. Create N-1,N-2… word model
2. Use low order model to predict high order [backoffs]
3. Pruning: $|P_{est}-P_{real}|<C$ => remove [iterative]

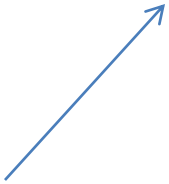
4. Perform lossy compression

Use low order model for prediction

$P(W1) = X$

$P(W2) = Y$

$P(W1,W2) = P(W1)*P(W2|W1) \approx X*Y$



Independent assumption
$P(W2|W1) \approx P(W2)$

Lossy compression of LM (1)

1. Take a good hash function

   F(w1,…wN) -> [0…BIG_INT)

   BIG_INT reduces noise

2. Encode Model:

   <to,be: 0.002> → <1213444: 0.002>

   <be,or: 0.0004> → <33755687: 0.0004>

   <or,not: 0.00043> → <6453: 0.00043>

Lossy compression of LMs (2)

hash	Prob
1213444	0.002
33755687	0.0004
6453	0.00043

Sort

hash	Prob
6453	0.00043
1213444	0.002
33755687	0.0004

Δ

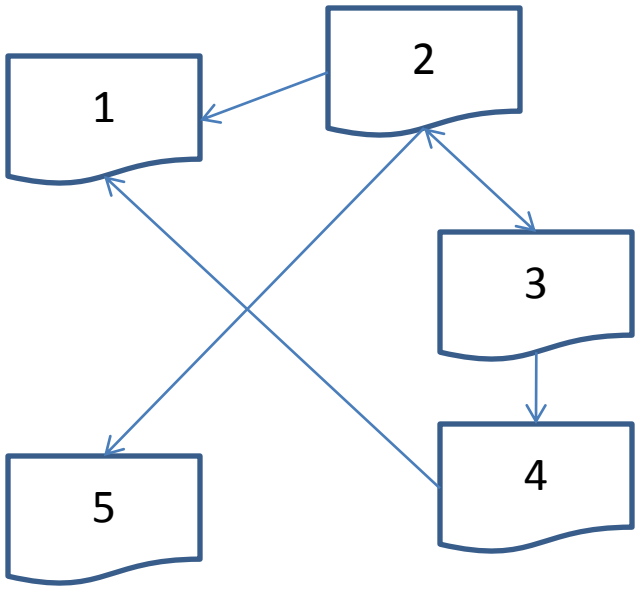
hash	Prob
0	0.00043
1206991	0.002
32542243	0.0004

Offsets
6453
…
…

Using compressed LM

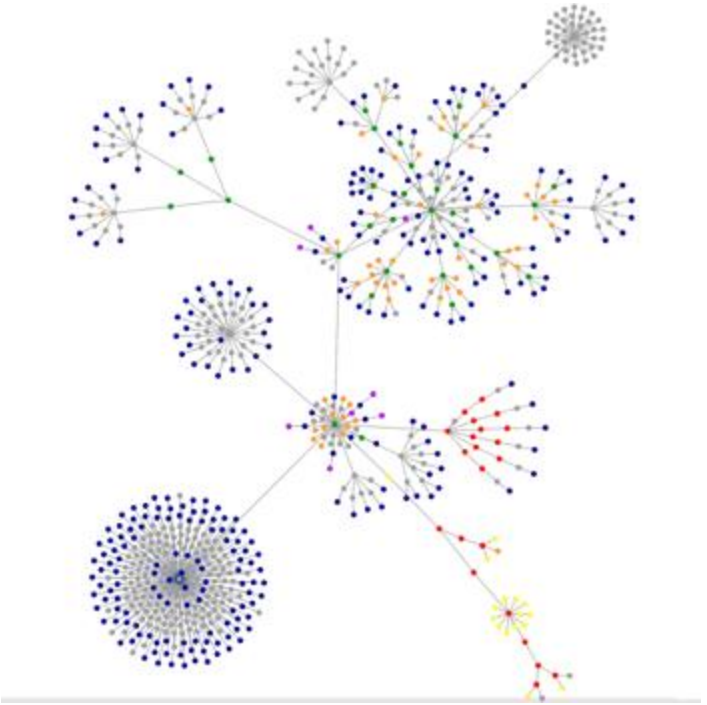
```
hash = hash_function(phrase)
(offset,n_offset) = search_in_offsets(hash)
for decom in decompression(offset,n_offset):
    if hash = decom.cod:
        return decom.prob
backup(phrase)
```

Links Analysis



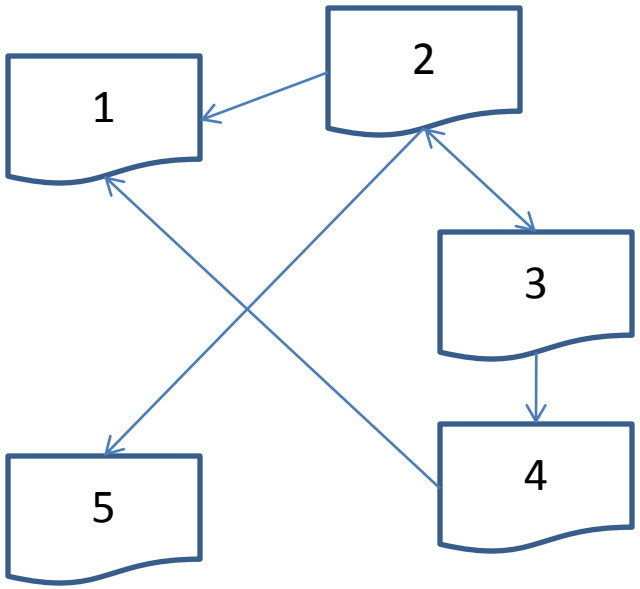
ID	1	2	3	4	5
1					
2	1		1		1
3		1		1	
4	1				
5					

Sparseness of Adjacency Matrix



- Small-world phenomena
- Power-law distribution

Adjacency Matrix Presentation



1→<>

2→<1,3,5>

3→<2,4>

4→<1>

5→<>

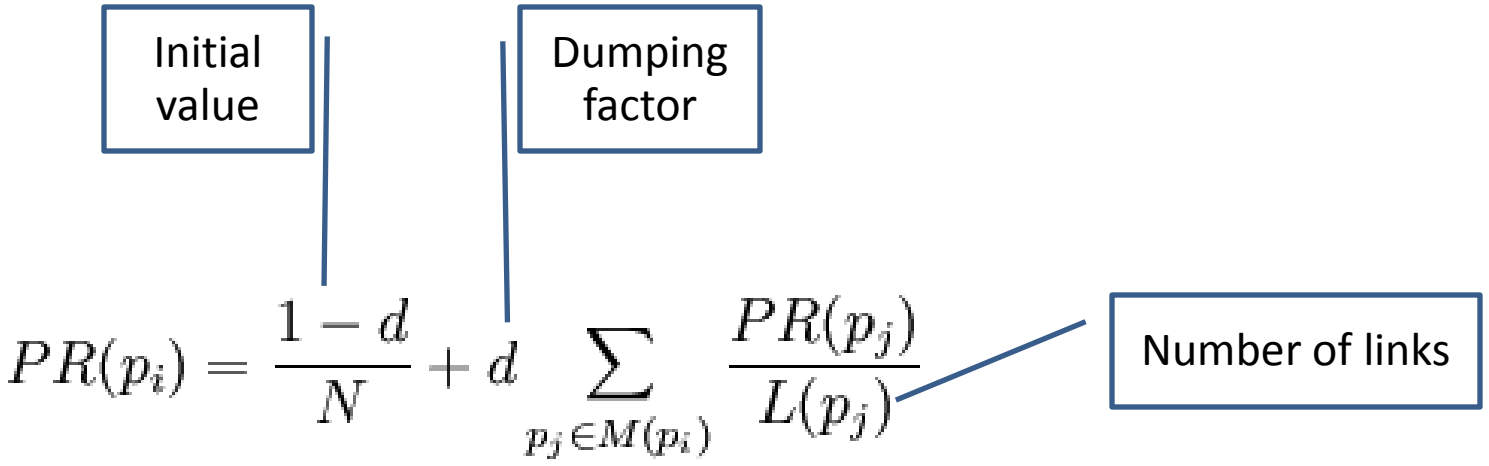
Compress with all algorithms from Inverted Files!
Webgraph http://law.dsi.unimi.it/ ζ codes 3.08 bits/link

PageRank Algorithm

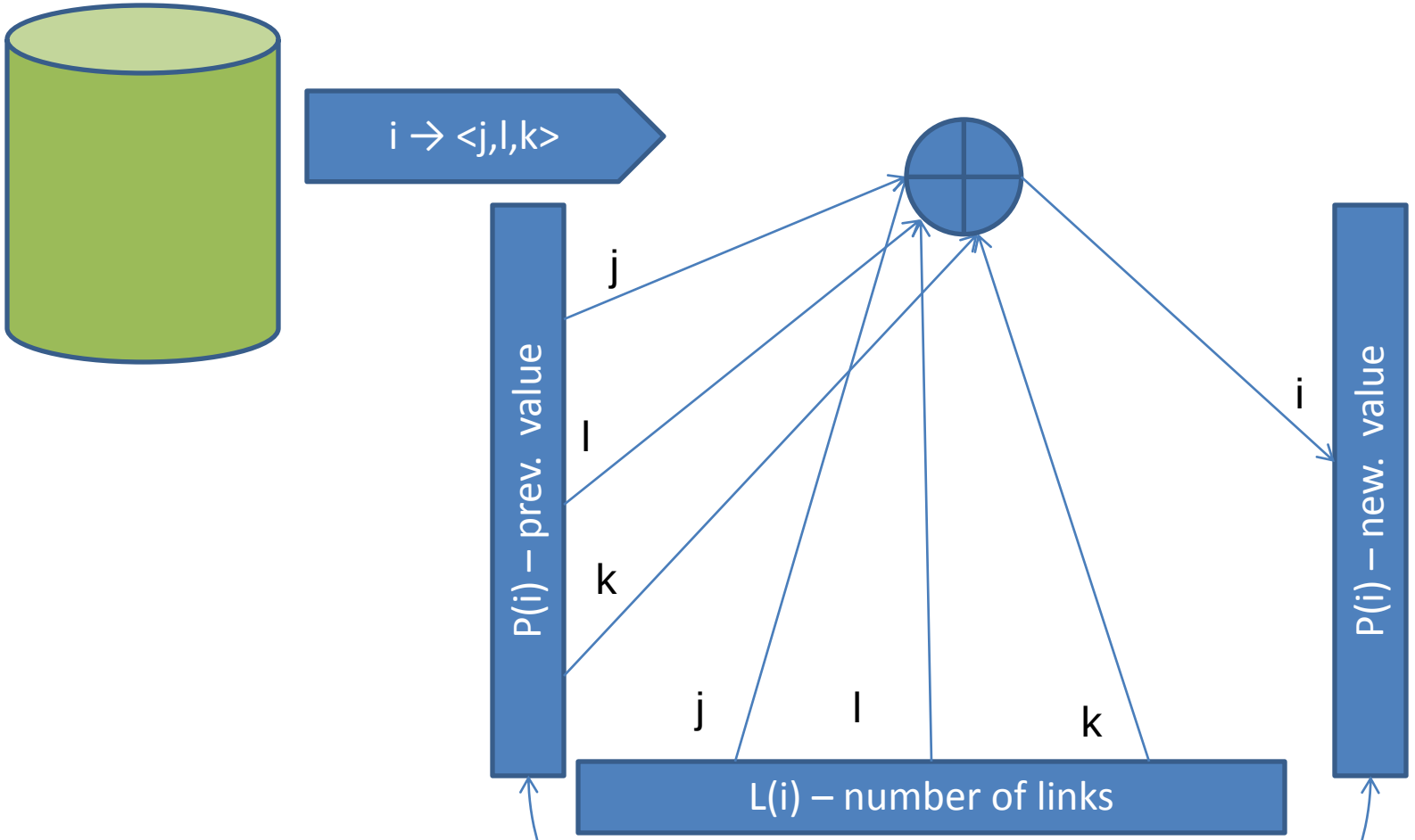
"Probability of random walk"

"Popularity metrics"

Eigenvector Ax = λx



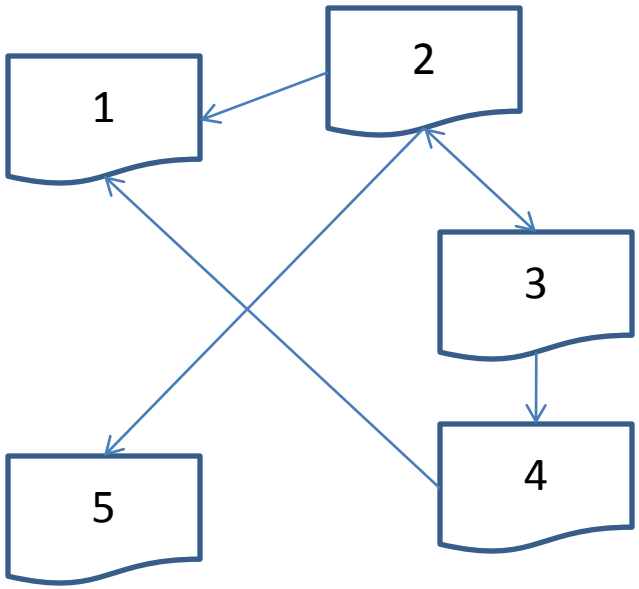
Pager Rank Implementation



$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

PageRank

optimization 1 (1)



Inverted Adjacency Matrix Presentation

1 →<2,4>

2 →<3>

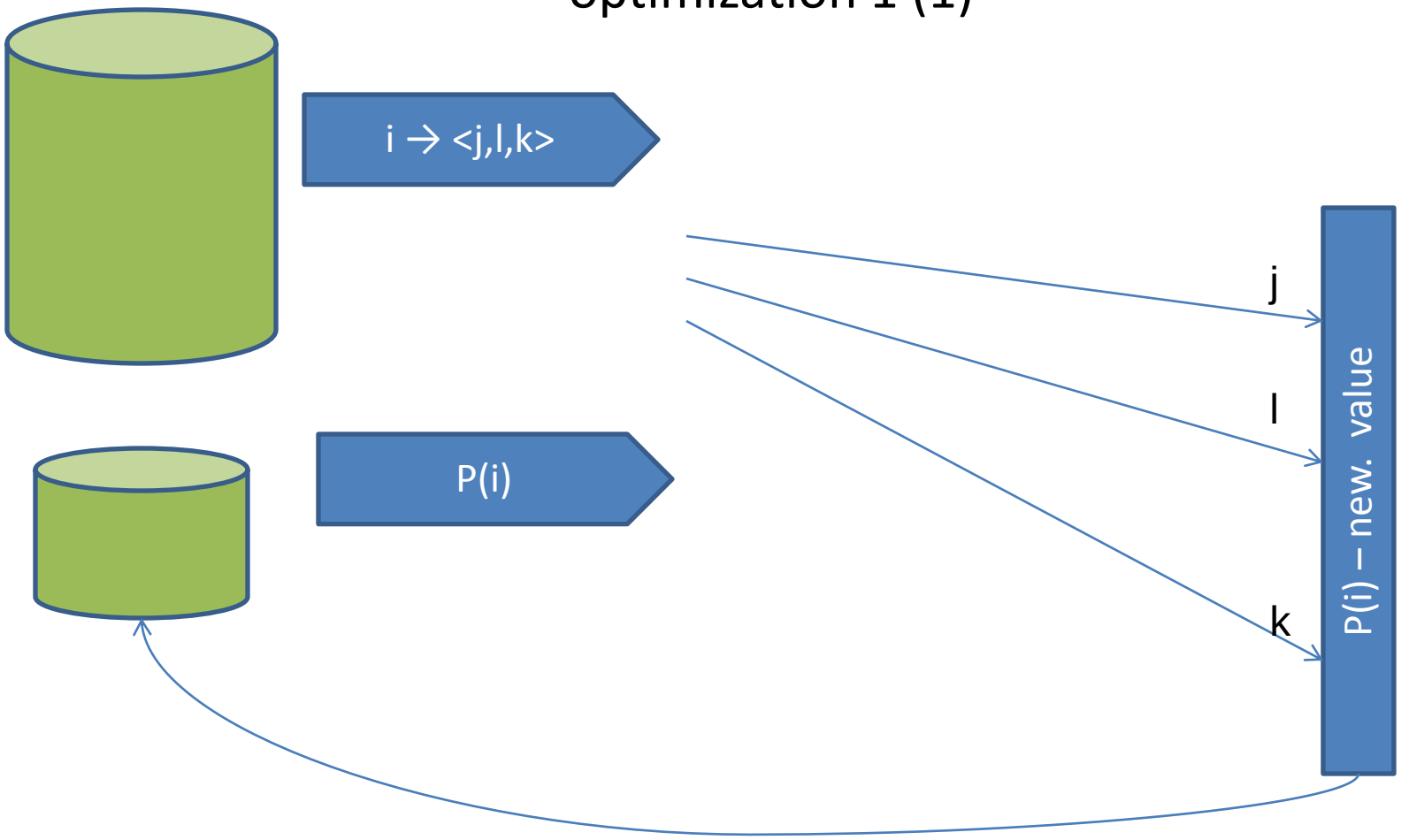
3 →<2>

4 →<3>

5 →<2>

PageRank

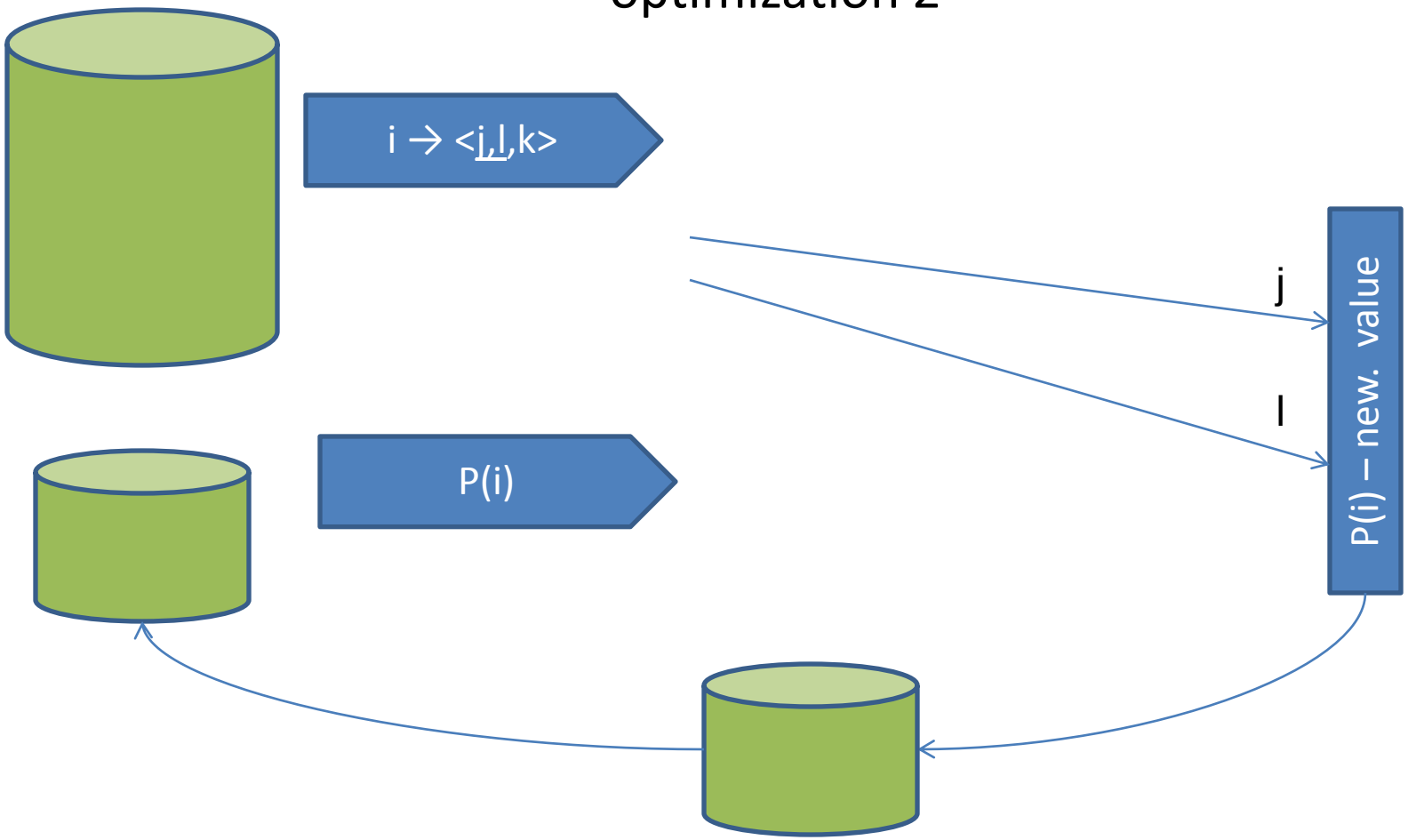
optimization 1 (1)



$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

PageRank

optimization 2

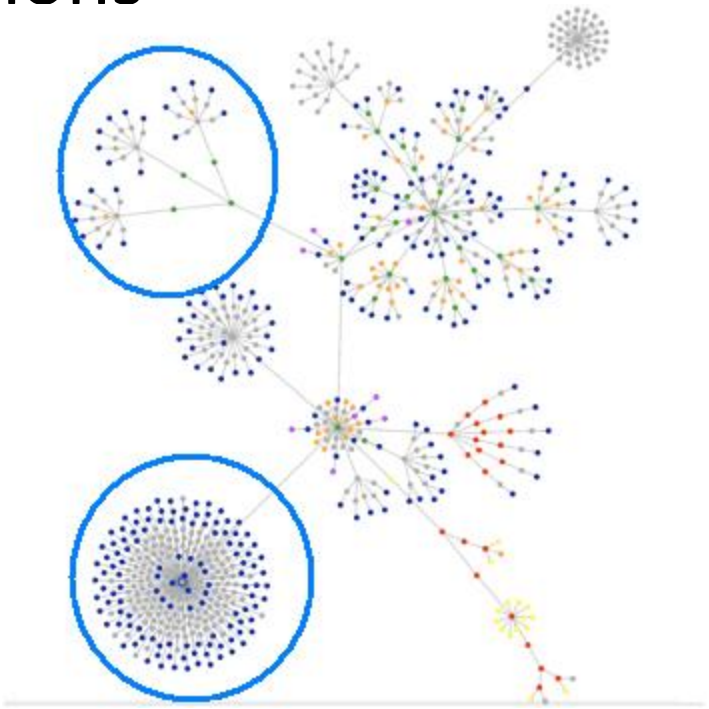


Reduce size of array

PageRank

optimization …

1. Reduce number of iterations: initial values from previous calculations
2. Parallelization
3. Graph reduce



Converge conditions

1. Fixed number of iterations (<100)
2. $\sum (P_{new}(i)-P(i)) < \Delta$
3. No re-ranking:

   $P(i) > P(j) \rightarrow P_{new}(i) > P_{new}(j)$

   Variants: N-top, sample

Finished

1. Index big corpus
2. Search in index
3. LM for spell-corrector
4. Link analysis

Q&A

References

1. **Emmanuel Eckard and Jean-C´edric Chappelier** Free Software for research in Information Retrieval and Textual Clustering **Technical report 2007**
2. **Ricardo Baeza-Yates et al.** Modern Information Retrieval ACM Press New York 1999
3. **Manning et al.** An Introduction to Information Retrieval Cambridge University Press (July 31, 2008)
4. **Donald E. Knuth** The Art of Computer Programming Vol. 3 Sorting and Searching , Second Edition Addison-Wesley, 1998
5. **Ulrich Drepper** What Every Programmer Should Know About Memory Red Hat, Inc 2007 http://people.redhat.com/drepper/cpumemory.pdf
6. **Witten et al.,** Managing Gigabytes: Compressing and Indexing Documents and Images, 2nd edition published by Morgan Kaufmann Publishing, San Francisco, ISBN 1-55860-570-3

References

7. **Cheng, C., Shann, J. J., and Chung, C.** Unique-order interpolative coding for fast querying and space-efficient indexing in information retrieval systems. Inf. Process. Manage. 42, 2 (Mar. 2006), 407-428.
8. **Sergey Melnik, Sriram Raghavan, Beverly Yang, Hector Garcia-Molina** Building a Distributed Full-Text Index for the Web Proceedings *WWW10, 2001*
9. **S.Buttcher, C.Clarke** A Document-Centric Approach to Static Index Pruning in Text Retrieval Systems, CIKM'06
10. **Diego Puppin, Fabrizio Silvestri, Domenico Laforenza.** "Query-Driven Document Partitioning and Collection Selection". Invited Paper. Proceedings of INFOSCALE '06
11. **Ken Church et al.** Compressing Trigram Language Models With Golomb Coding Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational

References

12. **Gianna M. Del Corso, Antonio Gullí, and Francesco Romani** Fast PageRank Computation via a Sparse Linear System Internet Mathematics Vol. 2, No. 3: 251-273
13. **Frank McSherry** A Uniform Approach to Accelerated PageRank Computation Proceedings WWW 2005 conference
14. **Taher Haveliwala**. "Efficient Computation of PageRank," Stanford University Technical Report, September 1999