

CarnegieMellon

15-826: Multimedia Databases and Data Mining

Lecture #13: Text - part II

C. Faloutsos

CarnegieMellon

Must-read Material

- MM Textbook, Chapter 6

Optional (but terrific to read)

- ★ • McIlroy, M. D. (Jan. 1982). "Development of a Spelling List." IEEE Trans. on Communications COM-30(1): 91-99.
 - <http://ieeexplore.ieee.org/document/1095395/>
- ★ • Severance, D. G. and G. M. Lohman (Sept. 1976). "Differential Files: Their Application to the Maintenance of Large Databases." ACM TODS 1(3): 256-267.
 - <http://dl.acm.org/citation.cfm?id=320484>

Outline

Goal: 'Find **similar / interesting** things'

- Intro to DB
- Indexing - similarity search
 - Primary key
 - ...
 - fractals
 - – Text
 - ...
- Data Mining

Text - Detailed outline

- text
 - problem
 - full text scanning
 - ➔ – inversion
 - signature files
 - clustering
 - information filtering and LSI

Problem




- How to find doc's with “data mining”?




CarnegieMellon

Conclusion



- How to find doc's with “data mining”?
- A1: full text scanning
 - A1.1: string editing distance
- A2: inversion
 - Elias Codes
- (A3: signature files – ‘Bloom filters’)

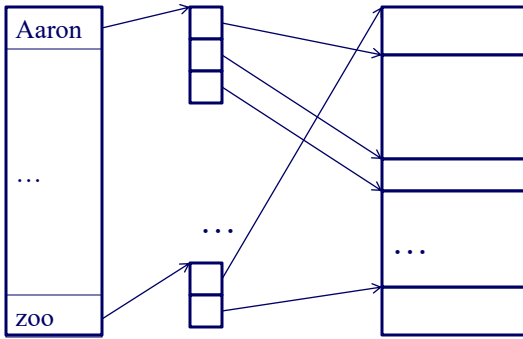


15-826 Copyright (c) 2019 C. Faloutsos 7

CarnegieMellon

Text - Inversion

Dictionary Postings lists Text file



15-826 Copyright (c) 2019 C. Faloutsos 8

CarnegieMellon

Text - Inversion

Dictionary Postings lists Text file

The diagram illustrates the text inversion process. On the left is a **Dictionary** containing words like "Aaron" and "ZOO". In the middle are **Postings lists**, which are lists of document IDs for each word. On the right is a **Text file** containing the original text. Arrows show that "Aaron" points to a postings list, which then points to the corresponding document in the text file. Similarly, "ZOO" points to another postings list, which points to the corresponding document in the text file.

Q: space overhead?

15-826 Copyright (c) 2019 C. Faloutsos 9

CarnegieMellon

Text - Inversion

- how to organize dictionary?
- stemming – Y/N?
- insertions?

15-826 Copyright (c) 2019 C. Faloutsos 10

Text - Inversion

- how to organize dictionary?
 - B-tree, hashing, TRIEs, PATRICIA trees, ...
- stemming – Y/N?
- insertions?


Text - Inversion

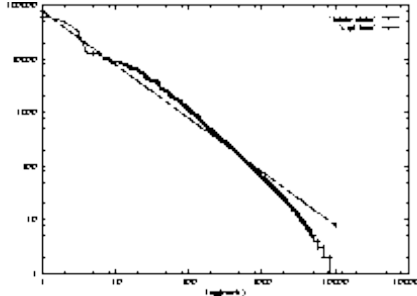
- variations:
- Parallelism [Tomasich+,93]
- Insertions [Tomasich+94], [Brown+]
 - ‘zipf’ distributions
- Approximate searching (‘glimpse’ [Wu+])

CarnegieMellon

Text - Inversion

- postings list – more Zipf distr.: eg., rank-frequency plot of ‘Bible’





$$freq \approx \frac{1}{rank \ln(1.78V)}$$

15-826 Copyright (c) 2019 C. Faloutsos 13

CarnegieMellon

Text - Inversion

- postings lists
 - Cutting+Pedersen
 - (keep first 4 in B-tree leaves)
 - how to allocate space: [Faloutsos+92]
 - geometric progression
 - compression (Elias codes) [Zobel+] – down to 2% overhead!
 - Compression and doc reordering [Blandford+2002]

15-826 Copyright (c) 2019 C. Faloutsos 14

CarnegieMellon

Text - Inversion

Dictionary Postings lists Text file

The diagram illustrates the text inversion process. On the left is a vertical 'Dictionary' box containing the words 'Aaron', '...', and 'ZOO'. In the middle are two vertical 'Postings lists' boxes, each containing three slots. Arrows point from 'Aaron' to the top slot of the first postings list, and from 'ZOO' to the bottom slot of the second postings list. On the right is a vertical 'Text file' box with four lines. Arrows point from the top slot of the first postings list to the first and second lines of the text file, and from the bottom slot of the second postings list to the third and fourth lines of the text file. Ellipses '...' are used to indicate that there are more words in the dictionary and more postings lists.

A: mainly, the postings lists

15-826 Copyright (c) 2019 C. Faloutsos 15

CarnegieMellon

Text - Inversion

Dictionary Postings lists Text file

The diagram is identical to the one on slide 15, showing the mapping from a dictionary to postings lists and then to a text file.

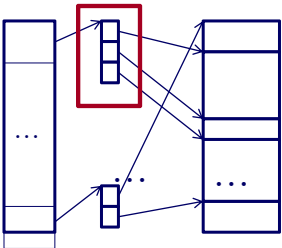
A: mainly, the postings lists **How to compress them?**

15-826 Copyright (c) 2019 C. Faloutsos 16

CarnegieMellon

How to compress them?

- A1: record the differences [Zobel+]



2,3,4 \rightarrow 3*4 bytes

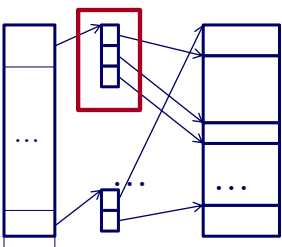
Q: Something better?

15-826 Copyright (c) 2019 C. Faloutsos 17

CarnegieMellon

How to compress them?

- A1: record the differences [Zobel+]



2,3,4 \rightarrow 3*4 bytes

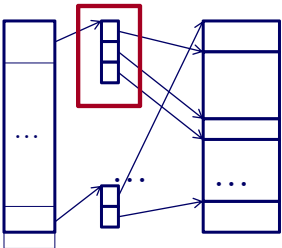
Q: Something better?
A: encode *deltas*
Q': how?

15-826 Copyright (c) 2019 C. Faloutsos 18

CarnegieMellon

How to compress them?

- A1: record the differences [Zobel+]



2,3,4 \rightarrow 3*4 bytes

Q: Something better?
A: encode *deltas*
Q': how?
A': Elias codes

15-826 Copyright (c) 2019 C. Faloutsos 19

CarnegieMellon

'short integers' \rightarrow few bits'

- How exactly to achieve that
- So that they are self-delimiting?

15-826 Copyright (c) 2019 C. Faloutsos 20

CarnegieMellon

Integer coding: small integers - > few bits

number	binary	Self-delimiting
2	10	00 1 10
3	11	00 1 11
15	1111	0000 1 1111

- $O(\log(i))$ bits for integer i
- can drop middle '1'
- can drop one of the zeros (!)
- (can apply recursively, to length)

15-826

21

CarnegieMellon

Integer coding: small integers - > few bits

number	binary	Self-delimiting
2	10	00 1 10
3	11	00 1 11
15	1111	0000 1 1111

- $O(\log(i))$ bits for integer i
- **can drop middle '1'**
- can drop one of the zeros (!)
- (can apply recursively, to length)

15-826

22

CarnegieMellon

Integer coding: small integers - > few bits

number	binary	Self-delimiting
2	10	00 1 10
3	11	00 1 11
15	1111	0000 1 1111

- $O(\log(i))$ bits for integer i
- can drop middle '1'
- **can drop one of the zeros (!)**
- (can apply recursively, to length)

15-826

23

CarnegieMellon

Integer coding: small integers - > few bits

number	binary	Self-delimiting
2	10	0 10
3	11	0 11
15	1111	000 1111

Elias gamma (γ) codes

15-826

Copyright (c) 2019 C. Faloutsos

24

CarnegieMellon

Text - Inversion

Dictionary Postings lists Text file

The diagram illustrates the text inversion process. On the left is a vertical 'Dictionary' box containing the words 'Aaron', '...', and 'ZOO'. In the middle are two vertical 'Postings lists' boxes, each containing three slots. Arrows point from 'Aaron' to the top three slots of the first postings list, and from 'ZOO' to the top three slots of the second postings list. On the right is a vertical 'Text file' box divided into several rows. Arrows point from the top three slots of the first postings list to the top three rows of the text file, and from the top three slots of the second postings list to the top three rows of the text file. Ellipses (...) are used to indicate that the dictionary, postings lists, and text file can contain more items than shown.

A: mainly, the postings lists

15-826 Copyright (c) 2019

**How to compress them
EVEN MORE?**

CarnegieMellon

How to compress them?

- A1: record the differences [Zobel+]
- A2: REORDER/cluster the doc's

-> smaller deltas

The diagram shows a similar structure to the previous slide, but with a red box highlighting a cluster of postings lists in the middle. This cluster is connected to a specific row in the text file on the right. The text file also has ellipses (...) to indicate other rows. This illustrates how reordering or clustering documents can lead to smaller deltas in the postings lists.

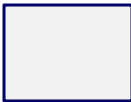
15-826 Copyright (c) 2019 C. Faloutsos 26

CarnegieMellon

Document Reordering

Doc1 Doc2 Doc3 ...

Aaron	1	0	1	0	1	0 ...
...						
ZOO	0	0	0	1	0	1 ...



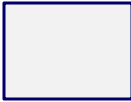
15-826 Copyright (c) 2019 C. Faloutsos 27

CarnegieMellon

Document Reordering

Doc1 Doc2 Doc3 ...

Aaron	1	0	1	0	1	0 ...
...						
ZOO	0	0	0	1	0	1 ...




15-826 Copyright (c) 2019 C. Faloutsos 28

CarnegieMellon

Document Reordering

	Doc1	Doc3	...			Doc2	
Aaron	1	1	1	0	0	0	...
...							
ZOO	0	0	0	1	1	0	...

Shorter runs; easier to compress



15-826 Copyright (c) 2019 C. Faloutsos 29


CarnegieMellon

Conclusions

- Conclusions: needs space overhead (2%-300%), but it is the fastest


15-826 Copyright (c) 2019 C. Faloutsos 30

CarnegieMellon



Conclusion

- How to find doc's with "data mining"?
- ✓ • A1: full text scanning
 - A1.1: string editing distance
- ✓ • A2: inversion
 - Elias Codes
- (A3: signature files – 'Bloom filters')



15-826 Copyright (c) 2019 C. Faloutsos 31

CarnegieMellon

Text - Detailed outline

- text
 - problem
 - full text scanning
 - inversion
 - ➔ – signature files
 - clustering
 - information filtering and LSI

15-826 Copyright (c) 2019 C. Faloutsos 32

CarnegieMellon

Signature files

- idea: 'quick & dirty' filter

Signature file Text file

The diagram shows two rectangular boxes representing files. The left box is labeled 'Signature file' and contains the text '..JoSm..'. The right box is labeled 'Text file' and contains the text '... John Smith ...'. Both boxes have a horizontal line near the top, suggesting a header or separator line.

15-826 Copyright (c) 2019 C. Faloutsos 33

CarnegieMellon

Signature files

- idea: 'quick & dirty' filter
- then, do seq. scan on sign. file and discard 'false alarms'
- Adv.: easy insertions; faster than seq. scan
- Disadv.: $O(N)$ search (with small constant)
- Q: how to extract signatures?

15-826 Copyright (c) 2019 C. Faloutsos 34

CarnegieMellon

Signature files

- A: superimposed coding!! [Mooers49], ...

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
doc.signature	001 010 111 011

m (=4 bits/word)
 F (=12 bits sign. size)

15-826 Copyright (c) 2019 C. Faloutsos 35

CarnegieMellon

Signature files

- A: superimposed coding!! [Mooers49], ...

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
doc.signature	001 010 111 011

data ↑ ↑↑ ↑

actual match

15-826 Copyright (c) 2019 C. Faloutsos 36

CarnegieMellon

Signature files

- A: superimposed coding!! [Mooers49], ...

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
doc.signature	001 010 111 011

retrieval ↑ ↑ ↑ ↑

actual dismissal

15-826 Copyright (c) 2019 C. Faloutsos 37

CarnegieMellon

Signature files

- A: superimposed coding!! [Mooers49], ...

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
doc.signature	001 010 111 011

nucleotic ↑ ↑ ↑ ↑

false alarm ('false drop')

15-826 Copyright (c) 2019 C. Faloutsos 38

CarnegieMellon

Signature files

- A: superimposed coding!! [Mooers49], ...

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
doc.signature	001 010 111 011

‘YES’ is ‘MAYBE’
‘NO’ is ‘NO’

15-826 Copyright (c) 2019 C. Faloutsos 39

CarnegieMellon

Signature files

- Q1: How to choose F and m ?
- Q2: Why is it called ‘false drop’ ?
- Q3: other apps of signature files?

15-826 Copyright (c) 2019 C. Faloutsos 40

Signature files

- Q1: How to choose F and m ?

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
<u>doc.signature</u>	<u>001 010 111 011</u>

m (=4 bits/word)

F (=12 bits sign. size)

Signature files

- Q1: How to choose F and m ?
- A: so that doc. signature is 50% full

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
<u>doc.signature</u>	<u>001 010 111 011</u>

m (=4 bits/word)

F (=12 bits sign. size)

CarnegieMellon

Signature files

- Q1: How to choose F and m ?
- ➔ • Q2: Why is it called ‘false drop’ ?
- Q3: other apps of signature files?

15-826

Copyright (c) 2019 C. Faloutsos

43

CarnegieMellon

Signature files

- Q2: Why is it called ‘false drop’ ?
- Old, but fascinating story [Mooers, 1949]
 - how to find qualifying books (by title word, and/or author, and/or keyword)
 - in $O(1)$ time?
 - **without computers** (1949...)



Calvin Mooers

15-826

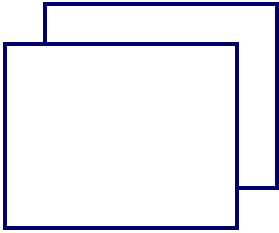
Copyright (c) 2019 C. Faloutsos

44

CarnegieMellon

Signature files

- ‘State of the art’ : cards – but $> O(1)$



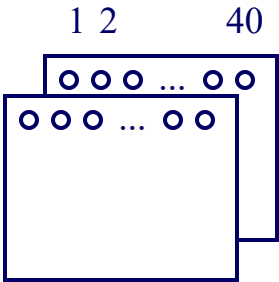
- one copy alpha by author
- one by title
- ...

15-826 Copyright (c) 2019 C. Faloutsos 45

CarnegieMellon

Signature files

- Solution: edge-notched cards



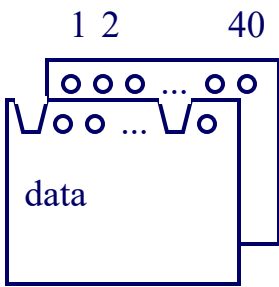
- each title word is mapped to m numbers (how?)
- and the corresponding holes are cut out:

15-826 Copyright (c) 2019 C. Faloutsos 46

CarnegieMellon

Signature files

- Solution: edge-notched cards



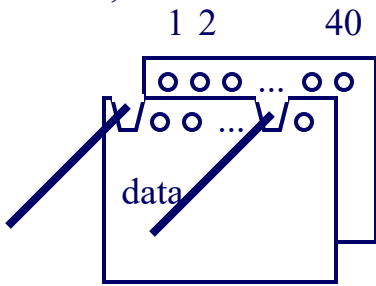
'data' -> #1, #39

15-826 Copyright (c) 2019 C. Faloutsos 47

CarnegieMellon

Signature files

- Search, e.g., for 'data': activate needle #1, #39, and shake the stack of cards!




'data' -> #1, #39

15-826 Copyright (c) 2019 C. Faloutsos 48

Signature files

- Also known as ‘zatocoding’, from ‘Zator’ company.

Signature files

- Q1: How to choose F and m ?
- Q2: Why is it called ‘false drop’ ?
-  • Q3: other apps of signature files?

CarnegieMellon

Signature files

- Q3: other apps of signature files?
- A: anything that has to do with **‘membership testing’**: does *‘data’* belong to the set of words of the document?

<u>Word</u>	<u>Signature</u>
data	001 000 110 010
base	000 010 101 001
<u>doc.signature</u>	<u>001 010 111 011</u>

15-826

Copyright (c) 2019 C. Faloutsos

51

CarnegieMellon

Signature files

- UNIX’s early ‘spell’ system [McIlroy]
- Bloom-joins in System R* [Mackert+] and ‘active disks’ [Riedel99]
- differential files [Severance + Lohman]



Doug
McIlroy



Eric
Riedel



Guy Lohman

15-826

Copyright (c) 2019 C. Faloutsos

52

CarnegieMellon

App#1: Unix' s spell

Dictionary

~30,000 words

aaron
apple
...
zoo

?
← electroencephalogram

What to do if the dictionary does not fit in memory (~1980)?

15-826 Copyright (c) 2019 C. Faloutsos 53

CarnegieMellon

App#1: Unix' s spell

A: allow for a few typos! And use

- huge bit string (2^{27})
- Hash each dictionary word to a bit
- Compress the string

aaron
apple
...
zoo

2^{27} bits

Copyright (c) 2019 C. Faloutsos 54

CarnegieMellon

App#1: Unix' s spell

Sub-questions:

- Q1: How often do we allow typos?
- Q2: Will the (compressed) bit string fit in memory?
- Q3: How to compress the bit string?

15-826

Copyright (c) 2019 C. Faloutsos

55

CarnegieMellon

App#2: Bloom-joins

R @Chicago

A	B	C
1		
1		
3		
...		
1		
12		

S @NY

A	E	F
1		
18		
1		
...		
23		
2		

R join S (@PIT)

15-826

Copyright (c) 2019 C. Faloutsos

56

CarnegieMellon

App#2: Bloom-joins

R @Chicago

A	B	C
1		
1		
3		
...		
1		
12		

S @NY

A	E	F
1		
18		
1		
...		
23		
2		

Idea: reduce transmission cost: 'R semijoin S'

57

CarnegieMellon

App#2: Bloom-joins

Idea: reduce transmission cost: 'R semijoin S'

That is,

- 'S' ships its unique values of 'A'
- 'R' deletes non-matching tuples
- (and they both send their tuples to PIT)

Q: what if we want to send at most, say 100 bytes NY -> Chicago?

15-826 Copyright (c) 2019 C. Faloutsos 58

CarnegieMellon

App#2: Bloom-joins

Q: what if we want to send at most,
say 100 bytes NY -> Chicago?

A: Bloom-join! Send a bloom filter
of the S.A values

15-826

Copyright (c) 2019 C. Faloutsos

59

CarnegieMellon

App#3: Differential files

Problem definition:

- A large file (eg., with EMPLOYEE records), nicely packed and organized (eg., B-tree)
- A few insertions/deletions, that we would like to keep separate, and merge, at night
- How to search, eg., for employee #123?

15-826

Copyright (c) 2019 C. Faloutsos

60

CarnegieMellon

App#3: Differential files

ssn	name	...
1		
5		
12		
...		
503		
509		

flag	ssn	name, ..
i	123	
i	55	
d	17	
d	33	

Differential file

15-826 Copyright (c) 2019 C. Faloutsos 61

CarnegieMellon

App#3: Differential files

- Q: How to search, eg., for employee #123?
- A: bloom-filter, for keys of diff. file

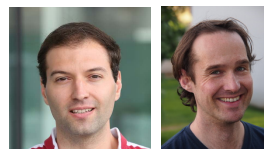
- Q: What are the advantages of differential files?
- A: <see paper, for 10(!) of them>

15-826 Copyright (c) 2019 C. Faloutsos 62

CarnegieMellon

App#4: Virus-scan

- Q: How to search, for thousands of patterns?
- A: Bloom filter:
- *Exact Pattern Matching with Feed-Forward Bloom Filters*, J. Moraru, and D. Andersen, (ALENEX11) , Jan 2011



15-826

Copyright (c) 2019 C. Faloutsos

CarnegieMellon

Signature files - conclusions


- easy insertions; slower than inversion
- brilliant idea of ‘quick and dirty’ filter: quickly discard the vast majority of non_qualifying elements, and focus on the rest.

15-826

Copyright (c) 2019 C. Faloutsos


64

CarnegieMellon



Conclusion

- How to find doc's with "data mining"?
- ✓ • A1: full text scanning
 - A1.1: string editing distance
- ✓ • A2: inversion
 - Elias Codes
- (A3: signature files – 'Bloom filters')



15-826 Copyright (c) 2019 C. Faloutsos 65

CarnegieMellon

References

- Blandford, D. and Blelloch, G. 2002. *Index Compression through Document Reordering*. Data Compression Conference (DCC '02) (April 02 - 04, 2002).
- Brown, E. W., J. P. Callan, et al. (March 1994). *Supporting Full-Text Information Retrieval with a Persistent Object Store*. EDBT conference, Cambridge, U.K., Springer Verlag.

15-826 Copyright (c) 2019 C. Faloutsos 66

References - cont' d

- Faloutsos, C. and H. V. Jagadish (Aug. 23-27, 1992). On B-tree Indices for Skewed Distributions. 18th VLDB Conference, Vancouver, British Columbia.
- Karp, R. M. and M. O. Rabin (March 1987). "Efficient Randomized Pattern-Matching Algorithms." IBM Journal of Research and Development 31(2): 249-260.
- Knuth, D. E., J. H. Morris, et al. (June 1977). "Fast Pattern Matching in Strings." SIAM J. Comput 6(2): 323-350.

References - cont' d

- Mackert, L. M. and G. M. Lohman (August 1986). R* Optimizer Validation and Performance Evaluation for Distributed Queries. Proc. of 12th Int. Conf. on Very Large Data Bases (VLDB), Kyoto, Japan.
- Manber, U. and S. Wu (1994). GLIMPSE: A Tool to Search Through Entire File Systems. Proc. of USENIX Techn. Conf.
- ★ • McIlroy, M. D. (Jan. 1982). "Development of a Spelling List." IEEE Trans. on Communications COM-30(1): 91-99.

References - cont' d

- Mooers, C. (1949). Application of Random Codes to the Gathering of Statistical Information Bulletin 31. Cambridge, Mass, Zator Co.
- Pedersen, D. C. a. J. (1990). Optimizations for dynamic inverted index maintenance. ACM SIGIR.
- Riedel, E. (1999). Active Disks: Remote Execution for Network Attached Storage. ECE, CMU. Pittsburgh, PA.

References - cont' d

- ★ • Severance, D. G. and G. M. Lohman (Sept. 1976). "Differential Files: Their Application to the Maintenance of Large Databases." ACM TODS 1(3): 256-267.
- Tomasic, A. and H. Garcia-Molina (1993). Performance of Inverted Indices in Distributed Text Document Retrieval Systems. PDIS.
- Tomasic, A., H. Garcia-Molina, et al. (May 24-27, 1994). Incremental Updates of Inverted Lists for Text Document Retrieval. ACM SIGMOD, Minneapolis, MN.

CarnegieMellon

References - cont' d

- Wu, S. and U. Manber (1992). "AGREP- A Fast Approximate Pattern-Matching Tool." .
- Zobel, J., A. Moffat, et al. (Aug. 23-27, 1992). An Efficient Indexing Technique for Full-Text Database Systems. VLDB, Vancouver, B.C., Canada.

15-826

Copyright (c) 2019 C. Faloutsos

71