

8. Mining & Organization

Mining & Organization

- Retrieving a **list of relevant documents** ([10 blue links](#)) insufficient
 - for **vague or exploratory** information needs (e.g., “find out about brazil”)
 - when there are **more documents than users can possibly inspect**
- **Organizing and visualizing** collections of documents can help users to **explore and digest** the contained information, e.g.:
 - **Clustering** groups content-wise similar documents
 - **Faceted search** provides users with means of exploration
 - **Timelines** visualize contents of timestamped document collections

Outline

8.1. Clustering

8.2. Faceted Search

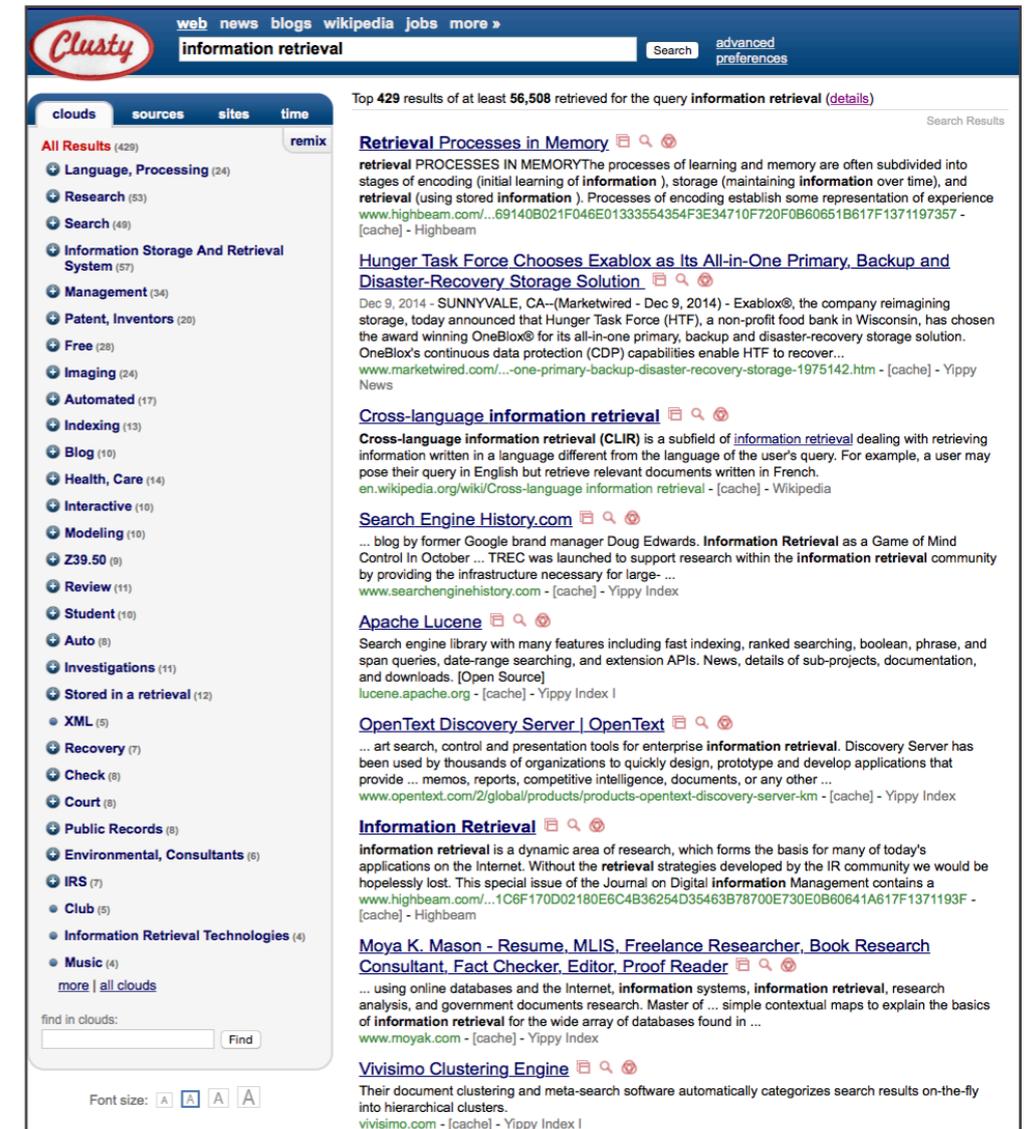
8.3. Tracking Memes

8.4. Timelines

8.5. Interesting Phrases

8.1. Clustering

- **Clustering** groups content-wise similar documents
- **Clustering** can be used to structure a document collection (e.g., entire corpus or query results)
- **Clustering methods:** DBScan, ***k*-Means**, *k*-Medoids, hierarchical agglomerative clustering
- Example of search result clustering: clusty.com



The screenshot shows the Clusty search engine interface. The search query is "information retrieval". The results are clustered into several categories, each with a count of items:

- All Results (429)
- Language, Processing (24)
- Research (53)
- Search (49)
- Information Storage And Retrieval System (57)
- Management (34)
- Patent, Inventors (20)
- Free (28)
- Imaging (24)
- Automated (17)
- Indexing (13)
- Blog (10)
- Health, Care (14)
- Interactive (10)
- Modeling (10)
- Z39.50 (9)
- Review (11)
- Student (10)
- Auto (8)
- Investigations (11)
- Stored in a retrieval (12)
- XML (5)
- Recovery (7)
- Check (8)
- Court (8)
- Public Records (8)
- Environmental, Consultants (6)
- IRS (7)
- Club (5)
- Information Retrieval Technologies (4)
- Music (4)

Below the clusters, there is a search bar and a "Find" button. The main content area displays the top 429 results, with the first few results visible:

- Retrieval Processes in Memory** - retrieval PROCESSES IN MEMORY The processes of learning and memory are often subdivided into stages of encoding (initial learning of information), storage (maintaining information over time), and retrieval (using stored information). Processes of encoding establish some representation of experience... [www.highbeam.com/...69140B021F046E01333554354F3E34710F720F0B60651B617F1371197357](#) - [cache] - Highbeam
- Hunger Task Force Chooses Exablox as Its All-in-One Primary, Backup and Disaster-Recovery Storage Solution** - Dec 9, 2014 - SUNNYVALE, CA--(Marketwired - Dec 9, 2014) - Exablox®, the company reimagining storage, today announced that Hunger Task Force (HTF), a non-profit food bank in Wisconsin, has chosen the award winning OneBlox® for its all-in-one primary, backup and disaster-recovery storage solution. OneBlox's continuous data protection (CDP) capabilities enable HTF to recover... [www.marketwired.com/...one-primary-backup-disaster-recovery-storage-1975142.htm](#) - [cache] - Yippy News
- Cross-language information retrieval (CLIR)** - Cross-language information retrieval (CLIR) is a subfield of information retrieval dealing with retrieving information written in a language different from the language of the user's query. For example, a user may pose their query in English but retrieve relevant documents written in French. [en.wikipedia.org/wiki/Cross-language_information_retrieval](#) - [cache] - Wikipedia
- Search Engine History.com** - ... blog by former Google brand manager Doug Edwards. Information Retrieval as a Game of Mind Control In October ... TREC was launched to support research within the information retrieval community by providing the infrastructure necessary for large- ... [www.searchenginehistory.com](#) - [cache] - Yippy Index
- Apache Lucene** - Search engine library with many features including fast indexing, ranked searching, boolean, phrase, and span queries, date-range searching, and extension APIs. News, details of sub-projects, documentation, and downloads. [Open Source] [lucene.apache.org](#) - [cache] - Yippy Index
- OpenText Discovery Server | OpenText** - ... art search, control and presentation tools for enterprise information retrieval. Discovery Server has been used by thousands of organizations to quickly design, prototype and develop applications that provide ... memos, reports, competitive intelligence, documents, or any other ... [www.opentext.com/2/global/products/products-opentext-discovery-server-km](#) - [cache] - Yippy Index
- Information Retrieval** - information retrieval is a dynamic area of research, which forms the basis for many of today's applications on the Internet. Without the retrieval strategies developed by the IR community we would be hopelessly lost. This special issue of the Journal on Digital Information Management contains a [www.highbeam.com/...1C6F170D02180E6C4B36254D35463B78700E730E0B60641A617F1371193F](#) - [cache] - Highbeam
- Moya K. Mason - Resume, MLIS, Freelance Researcher, Book Research Consultant, Fact Checker, Editor, Proof Reader** - ... using online databases and the Internet, information systems, information retrieval, research analysis, and government documents research. Master of ... simple contextual maps to explain the basics of information retrieval for the wide array of databases found in ... [www.moyak.com](#) - [cache] - Yippy Index
- Vivismo Clustering Engine** - Their document clustering and meta-search software automatically categorizes search results on-the-fly into hierarchical clusters. [vivismo.com](#) - [cache] - Yippy Index

k-Means

- **Cosine similarity** $\text{sim}(c,d)$ between document vectors c and d
- **Clusters** C_i represented by a cluster centroid document vector c_i
- **k-Means** groups documents into k clusters, maximizing the average similarity between documents and their cluster centroid

$$\frac{1}{|D|} \sum_{d \in D} \max_{c \in C} \text{sim}(c, d)$$

- Document d is assigned to cluster C having most similar centroid

Documents-to-Centroids

- **k-Means** is typically **implemented iteratively** with every iteration reading all documents and assigning them to most similar cluster
 - initialize cluster centroids c_1, \dots, c_k (e.g., as random documents)
 - **while** not converged (i.e., cluster assignments unchanged)
 - **for** every document d , determine **most similar** c_i , and **assign** it to C_i
 - recompute c_i as **mean** of documents assigned to cluster C_i
- Problem: Iterations need to **read the entire document collection**, which has cost in $O(nkd)$ with n as number of documents, k as number of clusters and, and d as number of dimensions

Centroids-to-Documents

- Broder et al. [1] devise an alternative method to implement k-Means, which makes use of **established IR methods**
- Key Ideas:
 - build an **inverted index** of the document collection
 - treat **centroids as queries** and identify the top-/ most similar documents in every iteration using **WAND**
 - documents showing up in **multiple top-/ results** are assigned to the most similar centroid
 - **recompute centroids** based on assigned documents
 - finally, assign **outliers** to cluster with most similar centroid

Sparsification

- While **documents are typically sparse** (i.e., contain only relatively few features with non-zero weight), **cluster centroids are dense**
- Identification of top- k most similar documents to a cluster centroid can further be speeded up by sparsifying, i.e., **considering only the p features having highest weight**

Experiments

- Datasets: Two datasets each with about 1M documents but different numbers of dimensions: ~26M for (1), ~7M for (2)

System	ℓ	Dataset 1 Similarity	Dataset 1 Time	Dataset 2 Similarity	Dataset 2 Time
k-means	—	0.7804	445.05	0.2856	705.21
wand-k-means	100	0.7810	83.54	0.2858	324.78
wand-k-means	10	0.7811	75.88	0.2856	243.9
wand-k-means	1	0.7813	61.17	0.2709	100.84

System	p	ℓ	Dataset 1 Similarity	Dataset 1 Time	ℓ	Dataset 2 Similarity	Dataset 2 Time
k-means	—	—	0.7804	445.05	—	0.2858	705.21
wand-k-means	—	1	0.7813	61.17	10	0.2856	243.91
wand-k-means	500	1	0.7817	8.83	10	0.2704	4.00
wand-k-means	200	1	0.7814	6.18	10	0.2855	2.97
wand-k-means	100	1	0.7814	4.72	10	0.2853	1.94
wand-k-means	50	1	0.7803	3.90	10	0.2844	1.39

- **Time per iteration** reduced from 445 minutes to 3.9 minutes on Dataset 1; 705 minutes to 1.39 minutes on Dataset 2

8.2. Faceted Search

8.2. Faceted Search

dblp.uni-trier.de

CompleteSearch DBLP

a DBLP mirror with extended search capabilities maintained by [Hannah Bast, University of Freiburg](#) (formerly [MPII Saarbrücken](#))

[Feedback](#) [Help](#)

zoomed in on 276 documents ... NEW: get these search results as [XML](#), [JSON](#), [JSONP](#)

efficient query processing

NOTE: The DBLP search has moved

CompleteSearch DBLP has moved to the new domain www.dblp.org. Please update any links or bookmarks you might have set accordingly. The pages under www.informatik.uni-trier.de will eventually be moved there, too. There will be a separate notification about that.

2015	
276	EE Yu Li, Man Lung Yiu: Route-Saver: Leveraging Route APIs for Accurate and Efficient Query Processing at Location-Based Services. <i>IEEE Trans. Knowl. Data Eng. (TKDE)</i> 27(1):235-249 (2015)
2014	
275	EE Haozhou Wang, Kai Zheng, Han Su, Jiping Wang, Shazia Wasim Sadiq, Xiaofang Zhou: Efficient Aggregate Farthest Neighbour Query Processing on Road Networks. <i>ADC</i> 2014:13-25
274	EE He Li, Jaesoo Yoo: An efficient scheme for continuous skyline query processing over dynamic data set. <i>BigComp</i> 2014:54-59
273	EE Heejung Yang, Chin-Wan Chung: Efficient Iceberg Query Processing in Sensor Networks. <i>Comput. J. (CJ)</i> 57(12):1834-1851 (2014)
272	EE Jiajia Li, Botao Wang, Guoren Wang, Xin Bi: Efficient Processing of Probabilistic Group Nearest Neighbor Query on Uncertain Data. <i>DASFAA</i> 2014:436-450
271	EE Nikolaos Nodarakis, Evaggelia Pitoura, Spyros Sioutas, Athanasios K. Tsakalidis, Dimitrios Tsoumakos, Giannis Tzimas: Efficient Multidimensional AkNN Query Processing in the Cloud. <i>DEXA</i> 2014:477-491
270	EE Yunjun Gao, Qing Liu, Baihua Zheng, Gang Chen: On efficient reverse skyline query processing. <i>Expert Syst. Appl. (ESWA)</i> 41(7):3237-3249 (2014)
269	EE Kisung Kim, Bongki Moon, Hyoung-Joo Kim: RG-index: An RDF graph index for efficient SPARQL query processing. <i>Expert Syst. Appl. (ESWA)</i> 41(10):4596-4607 (2014)
268	EE Alfredo Cuzzocrea, José Cecilio, Pedro Furtado: An Effective and Efficient Middleware for Supporting Distributed Query Processing in Large-Scale Cyber-Physical Systems. <i>IDCS</i> 2014:124-135
267	EE Khaled Mohammed Al-Naami, Sadi Evren Seker, Latifur Khan: GISQF: An Efficient Spatial Query Processing System. <i>IEEE CLOUD</i> 2014:681-688
266	EE Jia Liu, Bin Xiao, Kai Bu, Lijun Chen: Efficient distributed query processing in large RFID-enabled supply chains. <i>INFOCOM</i> 2014:163-171
265	EE Yuan-Ko Huang, Lien-Fa Lin: Efficient processing of continuous min-max distance bounded query with updates in road networks. <i>Inf. Sci. (ISCI)</i> 278:187-205 (2014)
264	EE Qiming Fang, Guangwen Yang: Efficient Top-k Query Processing Algorithms in Highly Distributed Environments. <i>JCP</i> 9(9):2000-2006 (2014)
263	EE Jiping Wang, Kai Zheng, Hoyoung Jeung, Haozhou Wang, Bolong Zheng, Xiaofang Zhou: Cost-Efficient Spatial Network Partitioning for Distance-Based Query Processing. <i>MDM</i> 2014:13-22
262	EE Sanjay Chatterji, G. S. Sreedhara, Maunendra Sankar Desarkar: An Efficient Tool for Syntactic Processing of English Query Text. <i>MIKE</i> 2014:278-287
261	EE Merih Seran Uysal, Christian Beecks, Thomas Seidl: On Efficient Query Processing with the Earth Mover's Distance. <i>PIKM@CIKM</i> 2014:25-32
260	EE Fabian Nagel, Gavin M. Bierman, Stratis D. Viglas: Code Generation for Efficient Query Processing in Managed Runtimes. <i>PVLDB</i> 7(12):1095-1106 (2014)
259	EE Tomas Karnagel, Matthias Hille, Mario Ludwig, Dirk Habich, Wolfgang Lehner, Max Heimel, Volker Markl: Demonstrating efficient query processing in heterogeneous environments. <i>SIGMOD</i> 2014:693-696
258	EE Junfeng Zhou, Zhifeng Bao, Wei Wang, Jinjia Zhao, Xiaofeng Meng: Efficient query processing for XML keyword queries based on the IDList index. <i>VLDB J. (VLDB)</i> 23(1):25-50 (2014)
2013	
257	EE Jianbin Qin, Wei Wang, Chuan Xiao, Yifei Lu, Xuemin Lin, Haixun Wang: Asymmetric signature schemes for efficient exact edit similarity query processing. <i>ACM Trans. Database Syst. (TODS)</i> 38(3):16 (2013)

[\[more\]](#)

Refine by AUTHOR

Hans-Peter Kriegel (11)
Guoren Wang (8)
Qing Li (7)
Yunjun Gao (7)
[\[top 4\]](#) [\[top 50\]](#) [\[top 250\]](#)

Refine by VENUE

IEEE Trans. Knowl. Data Eng. (TKDE) (15)
ICDE (12)
CIKM (10)
DASFAA (10)
[\[top 4\]](#) [\[top 50\]](#) [\[all 155\]](#)

Refine by YEAR

2015 (1)
2014 (18)
2013 (21)
2012 (19)
[\[top 4\]](#) [\[all 30\]](#)

Refine by TYPE

Conference (181)
Journal (90)
Book (3)
CoRR (2)
[\[top 4\]](#)

8.2. Faceted Search

The screenshot shows the Amazon search results page for 'digital camera'. The search bar at the top contains 'digital camera' and the results are sorted by 'Relevance'. The left sidebar contains a faceted search menu with the following sections:

- Show results for:**
 - Any Category
 - Electronics
 - Camera & Photo
 - Digital Cameras**
 - Point & Shoot Digital Cameras (7,099)
 - Point & Shoot Digital Camera Bundles (1,593)
 - DSLR Camera Bundles (14,423)
 - DSLR Cameras (2,709)
 - Compact System Camera Bundles (1,759)
 - Compact System Cameras (1,109)
 - See more
- Refine by:**
 - International Shipping**
 - Ship to Germany
 - Eligible for Free Shipping**
 - Free Shipping by Amazon
 - Camera Expert Reviews**
 - DPRReview Tested (283)
 - Digital Camera Megapixels**
 - 36 MP & Up (106)
 - 24 to 35.9 MP (1,924)
 - 22 to 23.9 MP (170)
 - 20 to 21.9 MP (1,477)
 - 18 to 19.9 MP (2,826)
 - 16 to 17.9 MP (3,555)
 - 14 to 15.9 MP (1,062)
 - 12 to 13.9 MP (1,989)
 - 10 to 11.9 MP (824)
 - 8 to 9.9 MP (429)
 - 7.9 MP & Under (1,656)
 - Optical Zoom**
 - 2.9x & Under (1,341)
 - 3x to 3.9x (2,192)
 - 4x to 5.9x (2,545)
 - 6x to 9.9x (653)
 - 10x to 12.9x (819)
 - 13x & Up (1,606)
 - Point & Shoot Digital Camera Video Resolution**
 - 1080 P
 - 720 P
 - 480 P
 - Point & Shoot Digital Camera Color**
 - Color swatches: Purple, Green, White, Yellow, Red, Blue, Black, Brown, Grey, Silver, Gold
 - Avg. Customer Review**
 - ★★★★★ & Up (5,072)
 - ★★★★☆ & Up (6,369)
 - ★★★☆☆ & Up (6,729)

The main content area shows product listings with the following details:

- Showing results in Electronics.** Show instead results in All Departments.
- Related Searches:** camera.
- Product 1:** Nikon Coolpix L330 - 20.2 MP Digital Camera with 26x zoom 35mm NIKKOR VR lens and FULL HD 720p (Black). Price: \$149.99 (was \$249.99). Prime. Get it by Monday, Jan 19. More Buying Choices: \$137.99 new (70 offers), \$145.00 used (3 offers). FREE Shipping on orders over \$35. #1 Best Seller in Compact System Cameras. 4.5 stars, 442 reviews.
- Product 2:** Sony W800/B 20.1 MP Digital Camera (Black). Price: \$89.00. More Buying Choices: \$89.00 new (3 offers), \$60.00 used (20 offers). FREE Shipping. 4.5 stars, 442 reviews.
- Product 3:** Nikon COOLPIX L830 16 MP CMOS Digital Camera with 34x Zoom NIKKOR Lens and Full 1080p HD Video (Black). Price: \$195.99 (was \$299.95). Prime. Get it by Monday, Jan 19. More Buying Choices: \$195.99 new (16 offers), \$170.91 used (19 offers). Trade-in eligible for an Amazon gift card. FREE Shipping on orders over \$35. 4.5 stars, 750 reviews.
- Product 4:** Nikon Coolpix S3600 Digital Camera (Silver) with 8GB Card + Case + Accessory Kit. Price: \$54.95 used & new (1 offer). 4.5 stars, 2 reviews.
- Product 5:** Sony W800/S 20 MP Digital Camera (Silver). Price: \$89.00. More Buying Choices: \$68.00 new (4 offers), \$63.24 used (12 offers). FREE Shipping. 4.5 stars, 442 reviews.
- Product 6:** Sony DSCW830/B 20.1 MP Digital Camera with 2.7-Inch LCD (Black). Price: \$74.99 used & new (23 offers). Trade-in eligible for an Amazon gift card. FREE Shipping on orders over \$35. #1 Best Seller in Digital Point & Shoot Cameras. 4.5 stars, 227 reviews.
- Product 7:** Canon PowerShot SX520 16 Digital Camera with 42x Optical Image Stabilized Zoom with 3-Inch LCD (Black). Price: \$199.00 (was \$329.00). Prime.
- Product 8:** Nikon COOLPIX L830 16 MP CMOS Digital Camera with 34x Zoom NIKKOR Lens and Full 1080p HD Video (Red). Price: \$195.00 (was \$299.96). Prime. Get it by Monday, Jan 19.
- Product 9:** Canon EOS Rebel T5 18MP EF-S Digital SLR Camera USA warranty with canon EF-S 18-55mm f/3.5-5.6 IS [Image Stabilizer...]. Price: \$599.95 (was \$899.95). Prime.

8.2. Faceted Search

The screenshot shows a search interface for 'Flamenco'. The search results are grouped by location, with 'Afghanistan' and 'China' visible. The left sidebar contains various facets such as Media, Location, Date, Themes, Objects, Nature, and Places and Spaces. The main content area displays a grid of items, including a 'Girl's Ceremonia...' and several 'Embroidery' pieces.

Flamenco

Refine your search further within these categories: These terms define your current search. Click the to remove a term.

Media (group results)
costume (3), drawing (2), lithograph (1), woodcut (6), woven object (2)

Location: all > Asia
Afghanistan (1), China (4), China or Tibet? (3), India (2), Japan (13), Russia (1), Turkey (3), Turkmenistan (1)

Date (group results)
17th century (3), 18th century (3), 19th century (10), 20th century (3), date ranges spanning multiple centuries (7), date unknown (2)

Themes (group results)
music, writing, and sport (5), nautical (1), religion (2)

Objects (group results)
clothing (5), food (1), furnishings (4), timepieces (1)

Nature (group results)
bodies of water (3), fish (1), flowers (2), geological formations (1), heavens (3), invertebrates and arthropods (1), mammals (2), plant material (3), trees (1)

Places and Spaces (group results)
bridges (1), buildings (1), dwellings (1)

Location: Asia

Shapes, Colors, and Materials: fabrics

Search all items within current results

28 items (grouped by location) [view ungrouped items](#)

Afghanistan 1

Girl's Ceremonia...
no artist
20th century

China 4

4 boats on lake...
Anonymous
post World War II

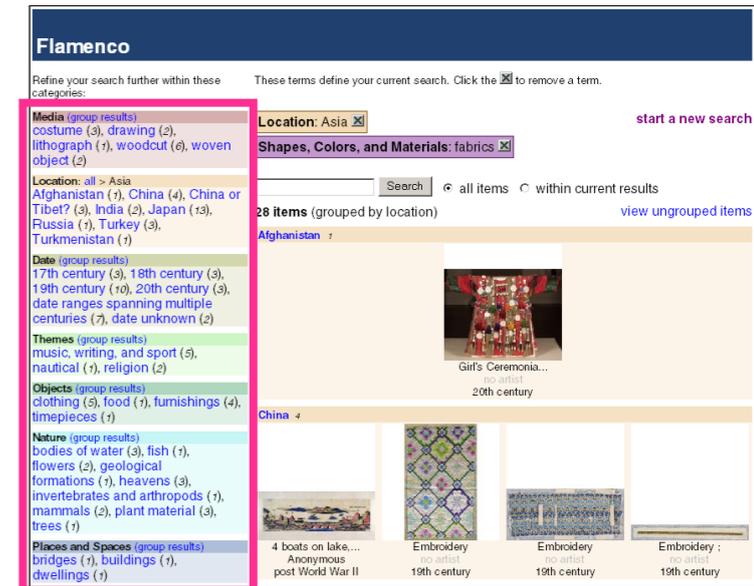
Embroidery
no artist
19th century

Embroidery
no artist
19th century

Embroidery ;
no artist
19th century

Faceted Search

- **Faceted search** [3,7] supports the user in exploring/navigating a collection of documents (e.g., query results)
- **Facets** are orthogonal sets of categories that can be **flat or hierarchical**, e.g.:
 - topic: arts & photography, biographies & memoirs, etc.
 - origin: Europe > France > Provence, Asia > China > Beijing, etc.
 - price: 1–10\$, 11–50\$, 51–100\$, etc.
- Facets are **manually curated** or **automatically derived** from meta-data



Automatic Facet Generation

- Need to manually curate facets **prevents their application** for **large-scale** document collections with **sparse meta-data**
- Dou et al. [3] investigate how facets can be **automatically mined** in a **query-dependent manner** from pseudo-relevant documents
- Observation: **Categories** (e.g., brands, price ranges, colors, sizes, etc.) are typically **represented as lists** in web pages
- Idea: Extract lists from web pages, rank and cluster them, and use the **consolidated lists as facets**

List Extraction

- Lists are **extracted from web pages** using several patterns
 - **enumerations** of items in text (e.g., *we serve beef, lamb, and chicken*)
via: `item{, item} * (and|or) {other} item`
 - **HTML form elements** (<SELECT>) and **lists** ()
ignoring instructions such as “select” or “chose”
 - as rows and columns of **HTML tables** (<TABLE>)
ignoring header and footer rows
- Items in extracted lists are **post-processed**, removing non-alphanumeric characters (e.g., brackets), converting them to lower case, and removing items longer than 20 terms

List Weighting

- Some of the extracted lists are **spurious** (e.g., from HTML tables)
- Intuition: Good lists consist of items that are **informative** to the query, i.e., are **mentioned in many** pseudo-relevant documents
- **Lists weighted** taking into account a document matching weight S_{DOC} and their average inverse document frequency S_{IDF}

$$S_l = S_{DOC} \cdot S_{IDF}$$

- **Document matching weight S_{DOC}**

$$S_{DOC} = \sum_{d \in R} (s_d^m \cdot s_d^r)$$

with s_d^m as fraction of list items mention in document d
and s_d^r as importance of document d (estimated as $\text{rank}(d)-1/2$)

List Weighting

- Average inverse document S_{IDF} is defined as

$$S_{IDF} = \frac{1}{|l|} \sum_{i \in l} idf(i)$$

- Problem: Individual lists (extracted from a single document) may still contain **noise**, be **incomplete**, or **overlap** with other lists
- Idea: Cluster lists containing similar items to consolidate them and form dimensions that can be used as facets

List Clustering

- Distance between two lists is defined as

$$d(l_1, l_2) = 1 - \frac{|l_1 \cap l_2|}{\min\{|l_1|, |l_2|\}}$$

- Complete-linkage distance between two clusters

$$d(c_1, c_2) = \max_{l_1 \in c_1, l_2 \in c_2} d(l_1, l_2)$$

- Greedy clustering algorithm

- pick most important not-yet-clustered list
- add nearest lists while cluster diameter is smaller than Dia_{max}
- save cluster if total weight is larger than W_{min}

Dimension and Item Ranking

- Problem: In which order to present dimensions and items therein?

- **Importance of a dimension** (cluster) is defined as

$$S_c = \sum_{s \in Sites(c)} \max_{l \in c, l \in s} S_l$$

favoring dimensions grouping lists with high weight

- **Importance of an item** within a dimension defined as

$$S_{i|c} = \sum_{s \in Sites(c)} \frac{1}{\sqrt{AvgRank(c, i, s)}}$$

favoring items which are often ranked high within containing lists

Anecdotal Results

◉ Dimensions mined from top-100 of commercial search engine

query: **watches**

1. cartier, breitling, omega, citizen, tag heuer, bulova, casio, rolex, audemars piguet, seiko, accutron, movado, fossil, gucci, ...
 2. men's, women's, kids, unisex
 3. analog, digital, chronograph, analog digital, quartz, mechanical, manual, automatic, electric, dive, ...
 4. dress, casual, sport, fashion, luxury, bling, pocket, ...
 5. black, blue, white, green, red, brown, pink, orange, yellow, ...
-

query: **lost**

1. season 1, season 6, season 2, season 3, season 4, season 5
 2. matthew fox, naveen andrews, evangeline lilly, josh holloway, jorge garcia, daniel dae kim, michael emerson, terry o'quinn, ...
 3. jack, kate, locke, sawyer, claire, sayid, hurley, desmond, boone, charlie, ben, juliet, sun, jin, ana, lucia ...
 4. what they died for, across the sea, what kate does, the candidate, the last recruit, everybody loves hugo, the end, ...
-

query: **lost season 5**

1. because you left, the lie, follow the leader, jughead, 316, dead is dead, some like it hoth, whatever happened happened, the little prince, this place is death, the variable, ...
 2. jack, kate, hurley, sawyer, sayid, ben, juliet, locke, miles, desmond, charlotte, various, sun, none, richard, daniel
 3. matthew fox, naveen andrews, evangeline lilly, jorge garcia, henry ian cusick, josh holloway, michael emerson, ...
 4. season 1, season 3, season 2, season 6, season 4
-

query: **flowers**

1. birthday, anniversary, thanksgiving, get well, congratulations, christmas, thank you, new baby, sympathy, fall
2. roses, best sellers, plants, carnations, lilies, sunflowers, tulips, gerberas, orchids, iris
3. blue, orange, pink, red, purple, white, green, yellow

query: **what is the fastest animals in the world**

1. cheetah, pronghorn antelope, lion, thomson's gazelle, wildebeest, cape hunting dog, elk, coyote, quarter horse
 2. birds, fish, mammals, animals, reptiles
 3. science, technology, entertainment, nature, sports, lifestyle, travel, gaming, world business
-

query: **the presidents of the united states**

1. john adams, thomas jefferson, george washington, john tyler, james madison, abraham lincoln, john quincy adams, william henry harrison, martin van buren, james monroe, ...
 2. the presidents of the united states of america, the presidents of the united states ii, love everybody, pure frosting, these are the good times people, freaked out and small, ...
 3. kitty, lump, peaches, dune buggy, feather pluckn, back porch, kick out the jams, stranger, boll weevil, ca plane pour moi, ...
 4. federalist, democratic-republican, whig, democratic, republican, no party, national union, ...
-

query: **visit beijing**

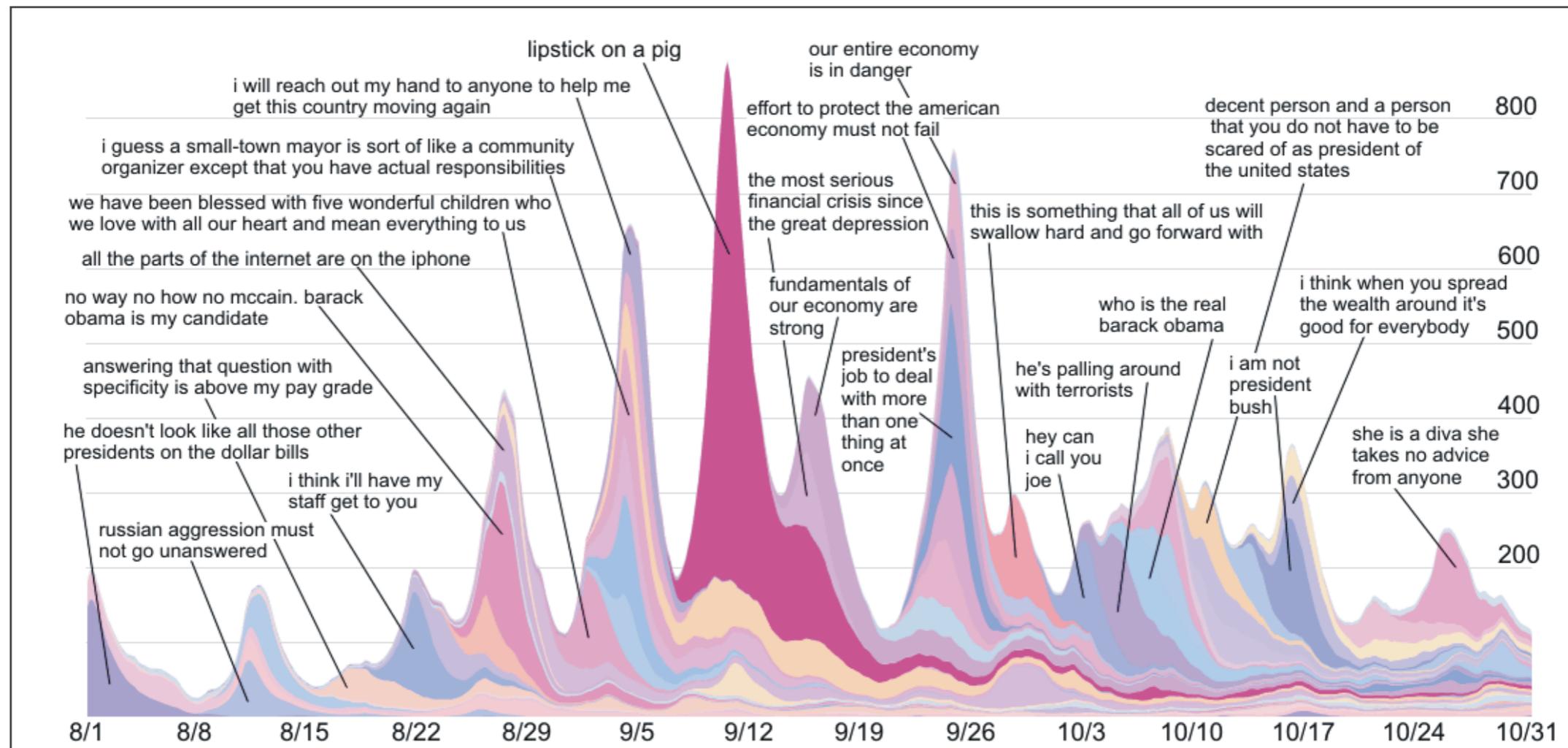
1. tiananmen square, forbidden city, summer palace, temple of heaven, great wall, beihai park, hutong
 2. attractions, shopping, dining, nightlife, tours, travel tip, transportation, facts
-

query: **cikm**

1. databases, information retrieval, knowledge management, industry research track
2. submission, important dates, topics, overview, scope, committee, organization, programme, registration, cfp, publication, programme committee, organisers, ...
3. acl, kdd, chi, sigir, www, icml, focs, ijcai, osdi, sigmod, sosp, stoc, uist, vldb, wsdm, ...

8.3. Tracking Memes

- Leskovec et al. [5] track **memes** (e.g., “lipstick on a pig”) and visualize their volume in traditional news and blogs

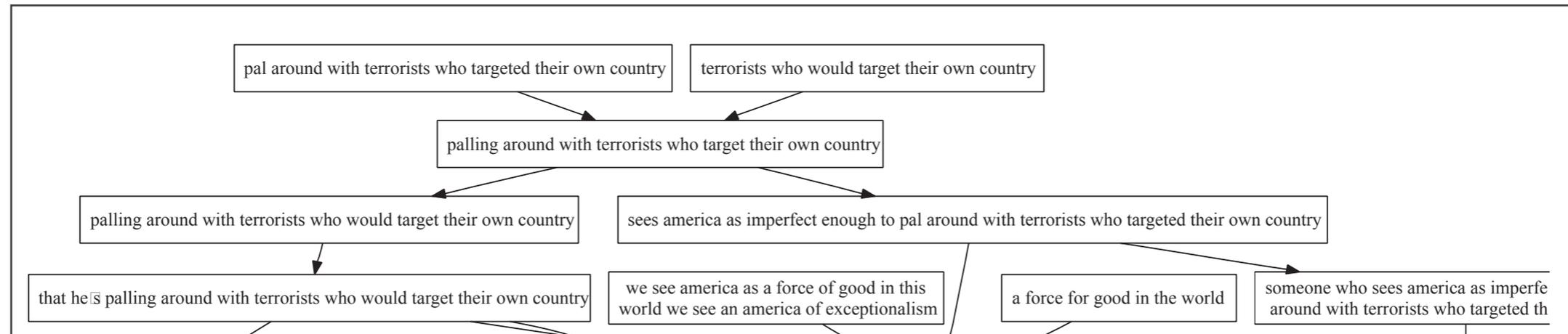


- Demo: <http://www.memetracker.org>

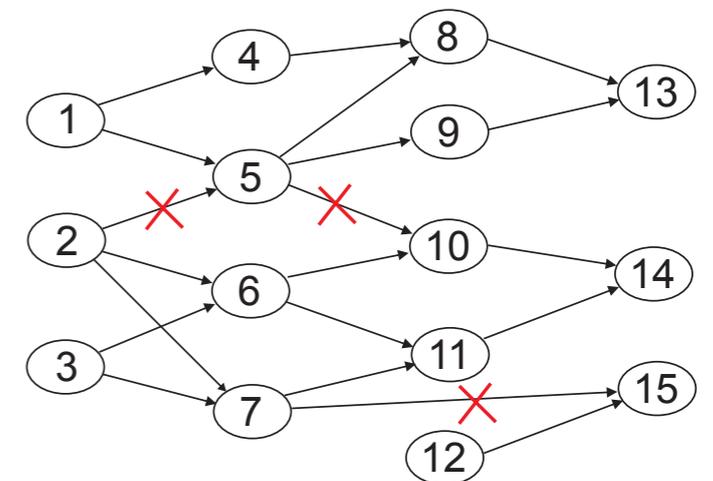
Phrase Graph Construction

- Problem: Memes are **often modified** as they spread, so that first **all mentions of the same meme** need to be identified
- Construction of a **phrase graph** $G(V, E)$:
 - **vertices** V correspond to **mentions of a meme** that are reasonably long and occur often enough
 - **edge** (u, v) exists if meme mentions u and v
 - u is **strictly shorter** than v
 - either: have **small directed token-level edit distance** (i.e., u can be transformed into v by adding at most ϵ tokens)
 - or: have a **common word sequence** of length at least k
 - **edge weights** based on **edit distance** between u and v and how often v occurs in the document collection

Phrase Graph Partitioning

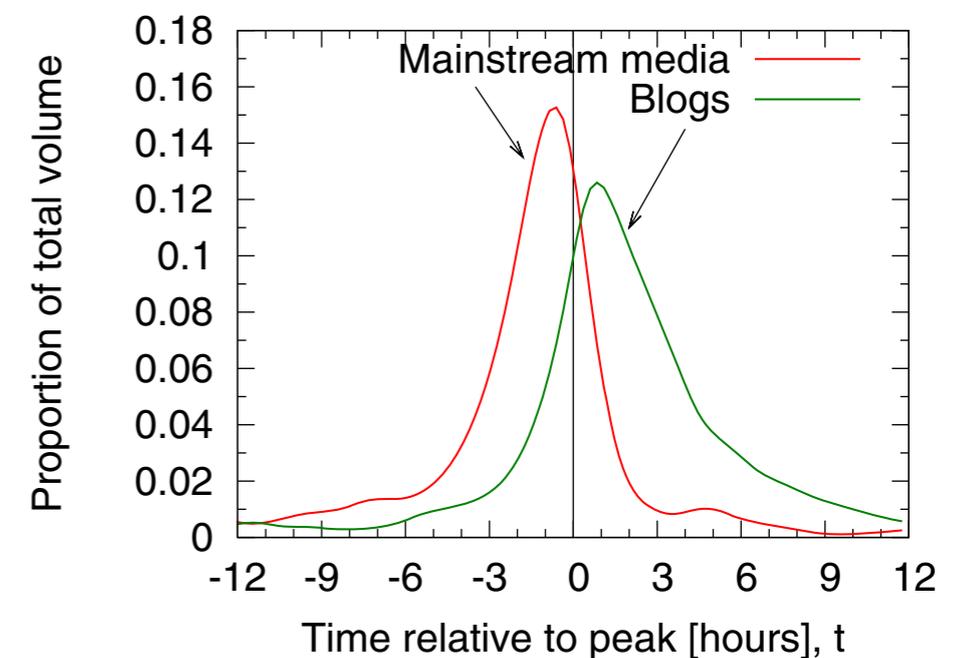
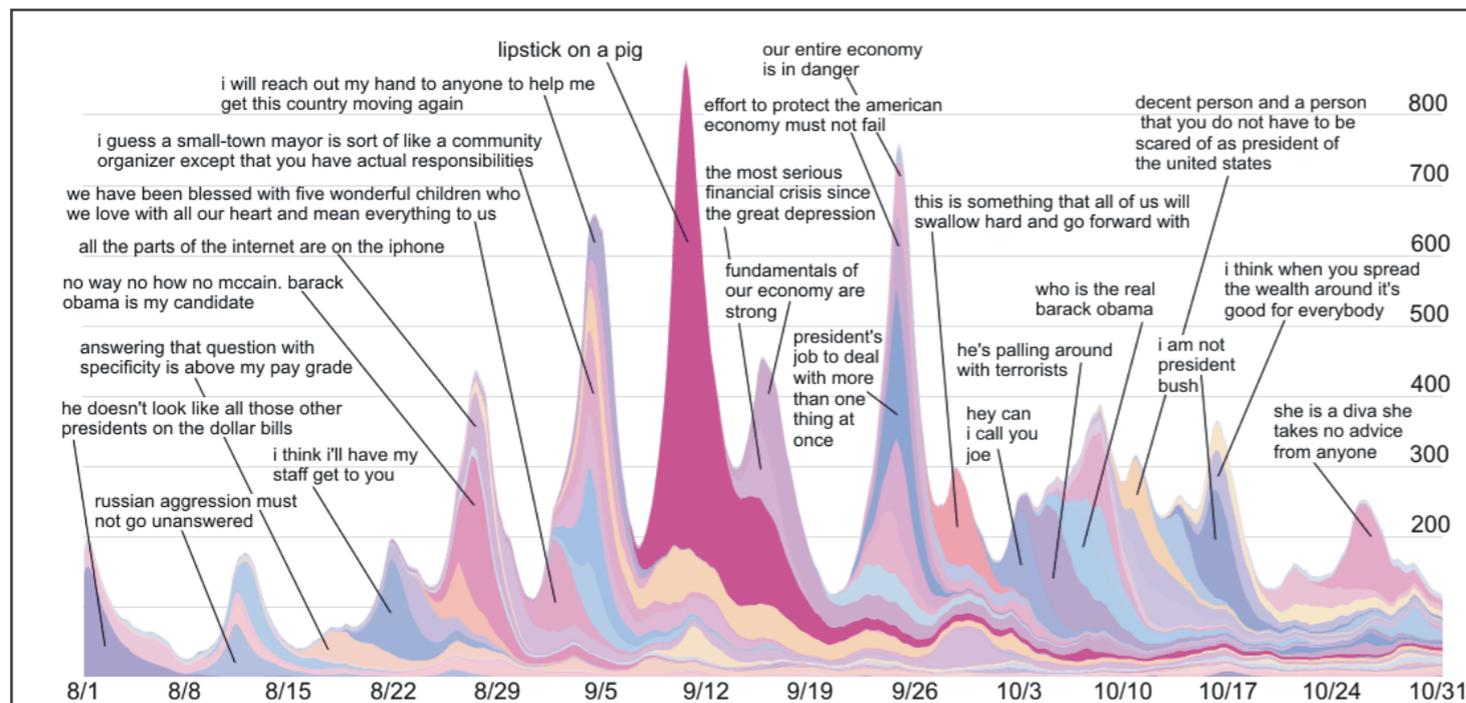


- Phrase graph is an **directed acyclic graph (DAG)** by construction
- Partition $G(V, E)$ by **deleting a set of edges having minimum total weight**, so that each resulting **component is single-rooted**
- Phrase graph partitioning is *NP*-hard, hence addressed by **greedy heuristic algorithm**



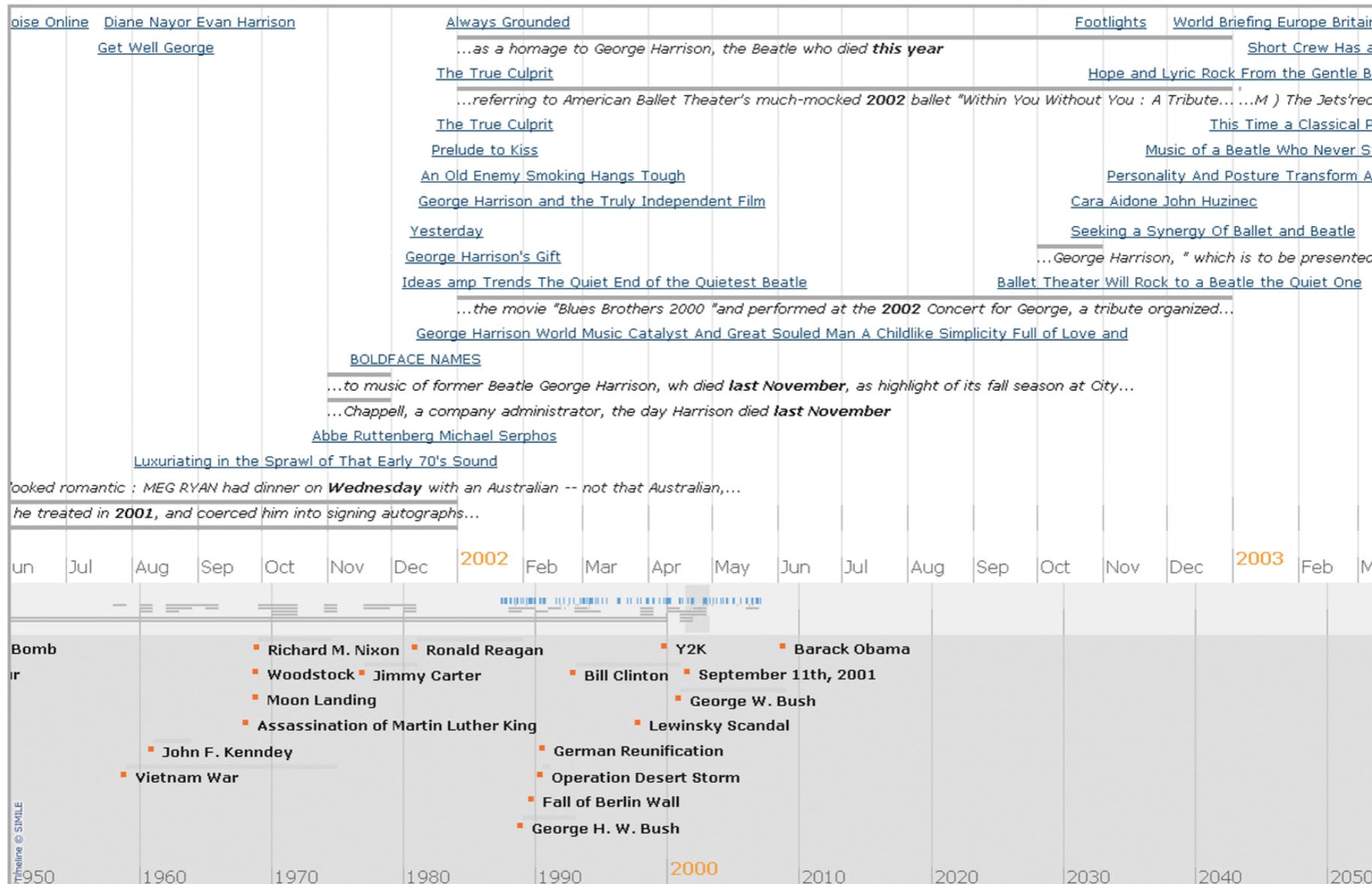
Applications

- **Clustering of meme mentions** allows for insightful analyses, e.g.:
 - **volume of meme** per time interval
 - **peek time** of meme in traditional news and social media
 - **time lag** between peek times in traditional news and social media



8.4. Timelines

- **Timelines** visualize, e.g., **major events and topics** and their **occurrence/importance** as they occur in a **collection of timestamped documents**



Timelines

- Swan and Allan [6] devise an approach based on **statistical tests** to **automatically generate a timeline** from a **collection of timestamped documents** (e.g., entire corpus or query result)
- consider only **named entities** (e.g., persons, organizations, locations) and **noun phrases** (e.g., nuclear power plant, debt crisis, car insurance)
- **partition document collection at day granularity**

Timelines

- Problem: How to identify significantly time-varying features?
- Assume that the following **statistics** have been computed
 - N_d as the number of documents in the partition for day d
 - N as the number of documents in the document collection
 - fd as the number of documents with feature f in the partition for day d
 - F as the number of documents with feature f in the document collection
- Derive a **contingency table** from these statistics

	f	¬f
d	fd	$N_d - fd$
¬d	$F - fd$	$N - N_d - F + fd$

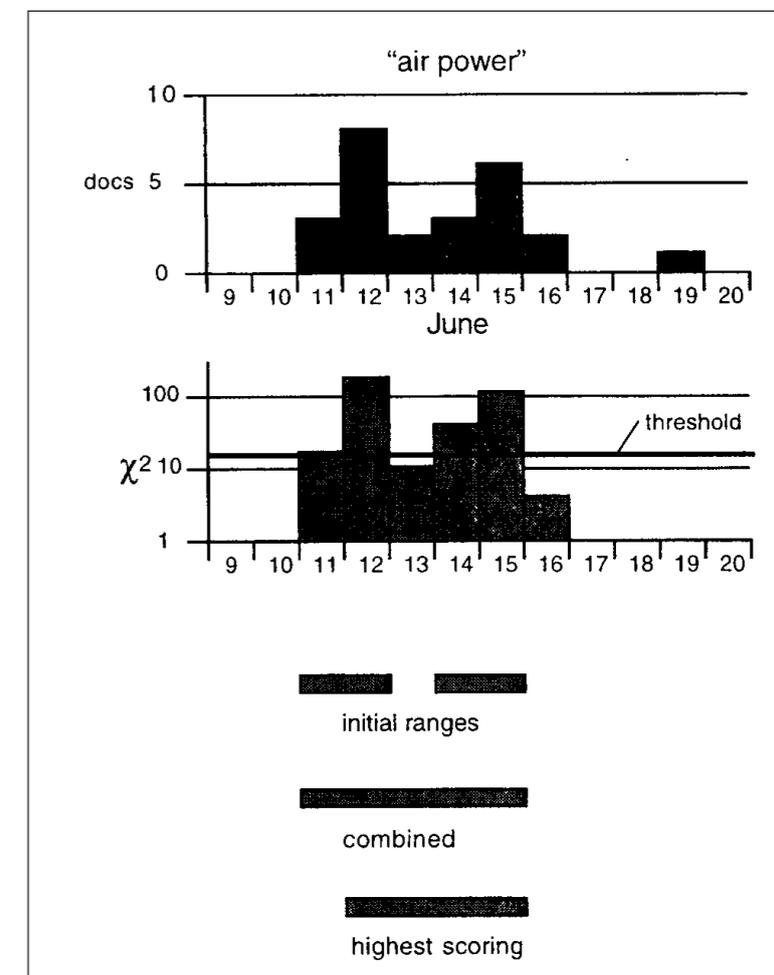
	f	¬f
d	a	b
¬d	c	d

χ^2 Statistic

- **χ^2 statistic** identifies features **which occur significantly more often on day d** than at other times covered by the collection

$$\chi^2 = \frac{N(ad - bc)^2}{(a + b)(a + c)(b + c)(b + d)}$$

- Keep days with **χ^2 score above threshold** and **coalesce ranges** of days allowing for a gap of at most one days in between
- Determine subrange with **highest χ^2 score**



8.5. Interesting Phrases

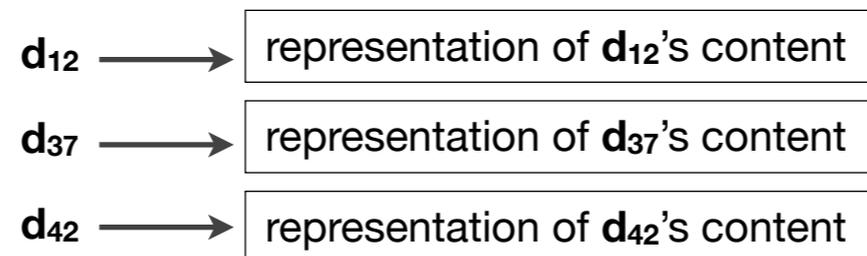
- Bedathur et al. [2] consider the problem of **identifying interesting phrases** that are descriptive for a given query result D'
- Phrase p is considered **interesting** if it occurs **more often in documents from D'** than in the general document collection D

$$I(p, D') = \frac{df(p, D')}{df(p, D)}$$

- Phrase p is **only considered** if it
 - **occurs at least σ times** in the document collection (e.g., set as 10)
 - **has length of at most λ** (e.g., set as 5)

How to Identify Interesting Phrases Efficiently?

- **Forward index** maintains a representation of every document



- **Phrase dictionary** keeps frequency $df(p, D)$ for every phrase p
- **High-level algorithm** for identifying top- k interesting phrases
 - **access** the forward index for each $d \in D'$
 - **merge** the $|D'|$ document representations
 - **output** the k most interesting phrases
- **Different document representations** differ in terms of efficiency

Document Content

- Idea: Represent document content explicitly as a **sequence of terms** (or compressed term identifiers)



- Benefit:
 - **space efficient**
- Drawbacks:
 - requires **enumeration of all phrases** in document including globally infrequent ones that occur less than σ times in D
 - requires **phrase dictionary**

Phrases

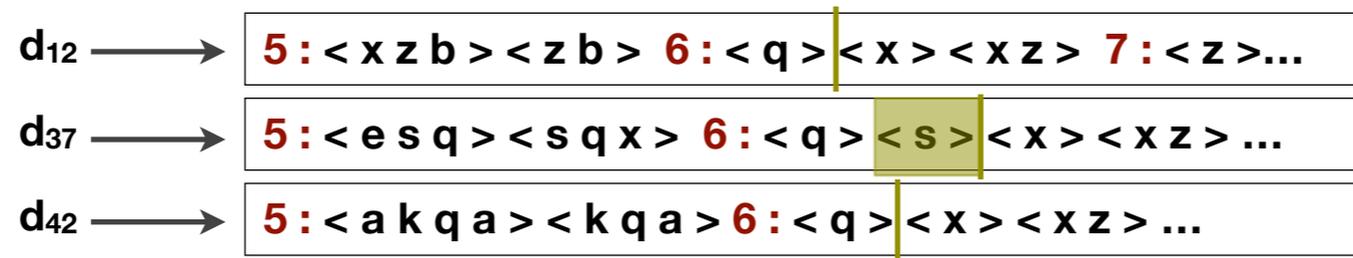
- Idea: Keep **all globally frequent phrases** contained in document **d** in a consistent (e.g., lexicographic) order

d_{12}	→	<code>< a > < a x > < a x z > < b > < b l > ...</code>
d_{37}	→	<code>< d > < d l > < d l e > < e > < e s > ...</code>
d_{42}	→	<code>< a > < a k > < a y > < d > < d a > ...</code>

- Benefits:
 - considers **only globally frequent phrases**
 - **consistent order** allows for **efficient merging**
- Drawbacks:
 - **space inefficient**
 - requires **phrase dictionary**

Frequency-Ordered Phrases

- Idea: Keep **all globally frequent phrases** contained in document d in **ascending order of their embedded global frequency**



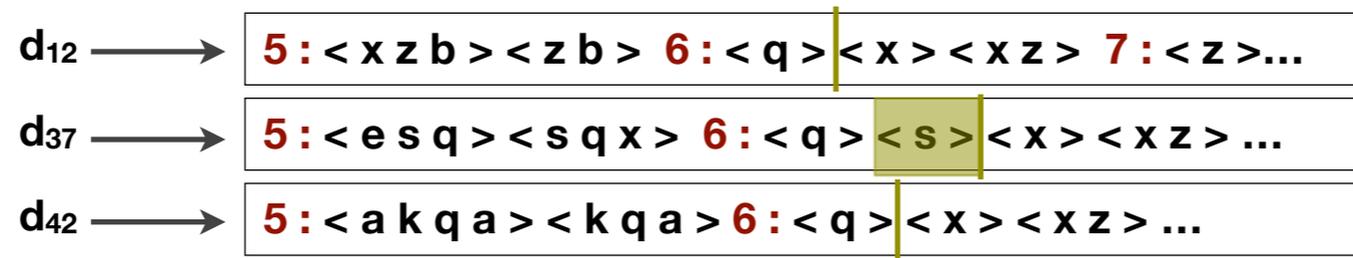
- Interestingness of any unseen phrase is **upper-bounded** by

$$\min\left(1, \frac{|D'|}{df(p, D)}\right)$$

where p is the **last phrase encountered**

Frequency-Ordered Phrases

- Idea: Keep **all globally frequent phrases** contained in document d in **ascending order of their embedded global frequency**



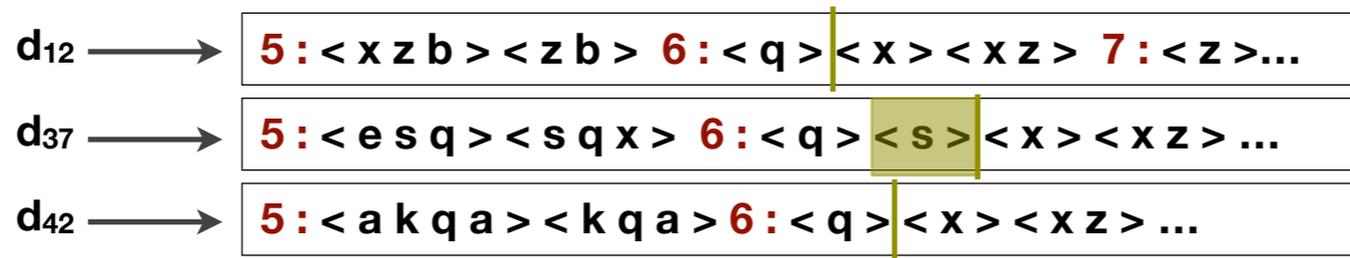
- Interestingness of any unseen phrase is **upper-bounded** by

$$\min\left(1, \frac{|D'|}{df(p, D)}\right) \quad \frac{3}{6}$$

where p is the **last phrase encountered**

Frequency-Ordered Phrases

- Idea: Keep **all globally frequent phrases** contained in document d in **ascending order of their embedded global frequency**



- Benefits:
 - **early termination** possible when no unseen phrase can make it into the top- k most interesting phrases
 - **self-contained** (i.e., no phrase dictionary needed)
- Drawbacks:
 - **space inefficient**

Prefix-Maximal Phrases

- Observation: Globally frequent phrases are often **redundant** and we do not have to keep all of them

d_{12} →

< a > < a x > < a x z > < b > < b l > ...

d_{37} →

< d > < d l > < d l e > < e > < e s > ...

d_{42} →

< a > < a k > < a y > < d > < d a > ...

- Definition: A phrase p is **prefix-maximal** in document d if
 - p is **globally frequent**
 - d does not contain another globally frequent phrase p' of which p is a **prefix**
- Prefix-maximal phrase p (e.g., <a x z> in d_{12}) **represents all its prefixes** (i.e., <a> and <a x>); they're guaranteed to be globally frequent and contained in d

Prefix-Maximal Phrases

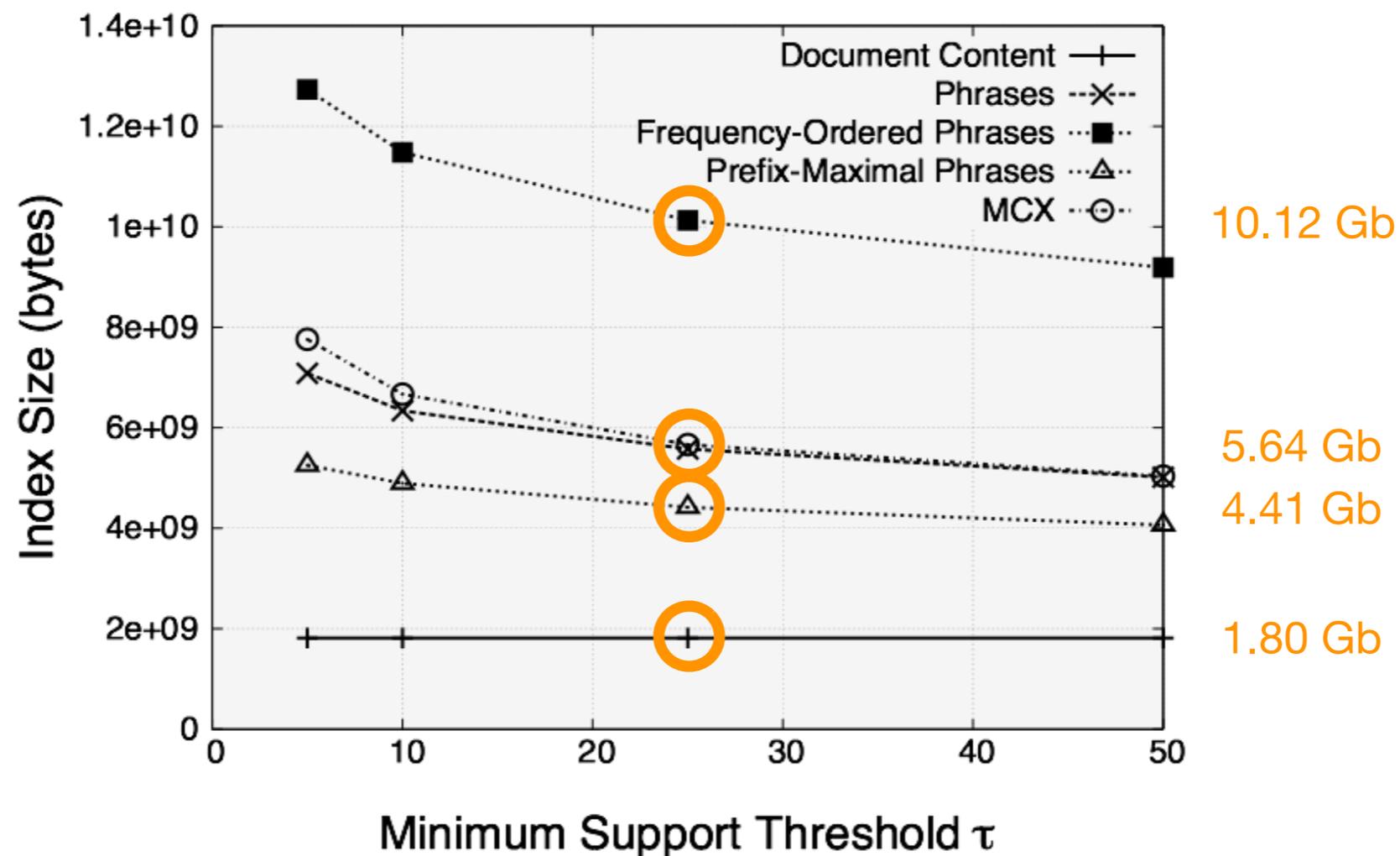
- Idea: Keep **only prefix-maximal phrases** contained in d in lexicographic order and extract prefixes on-the-fly



- Benefits:
 - **space efficient**
- Drawbacks:
 - extraction of prefixes entails **additional bookkeeping**
 - requires **phrase dictionary**

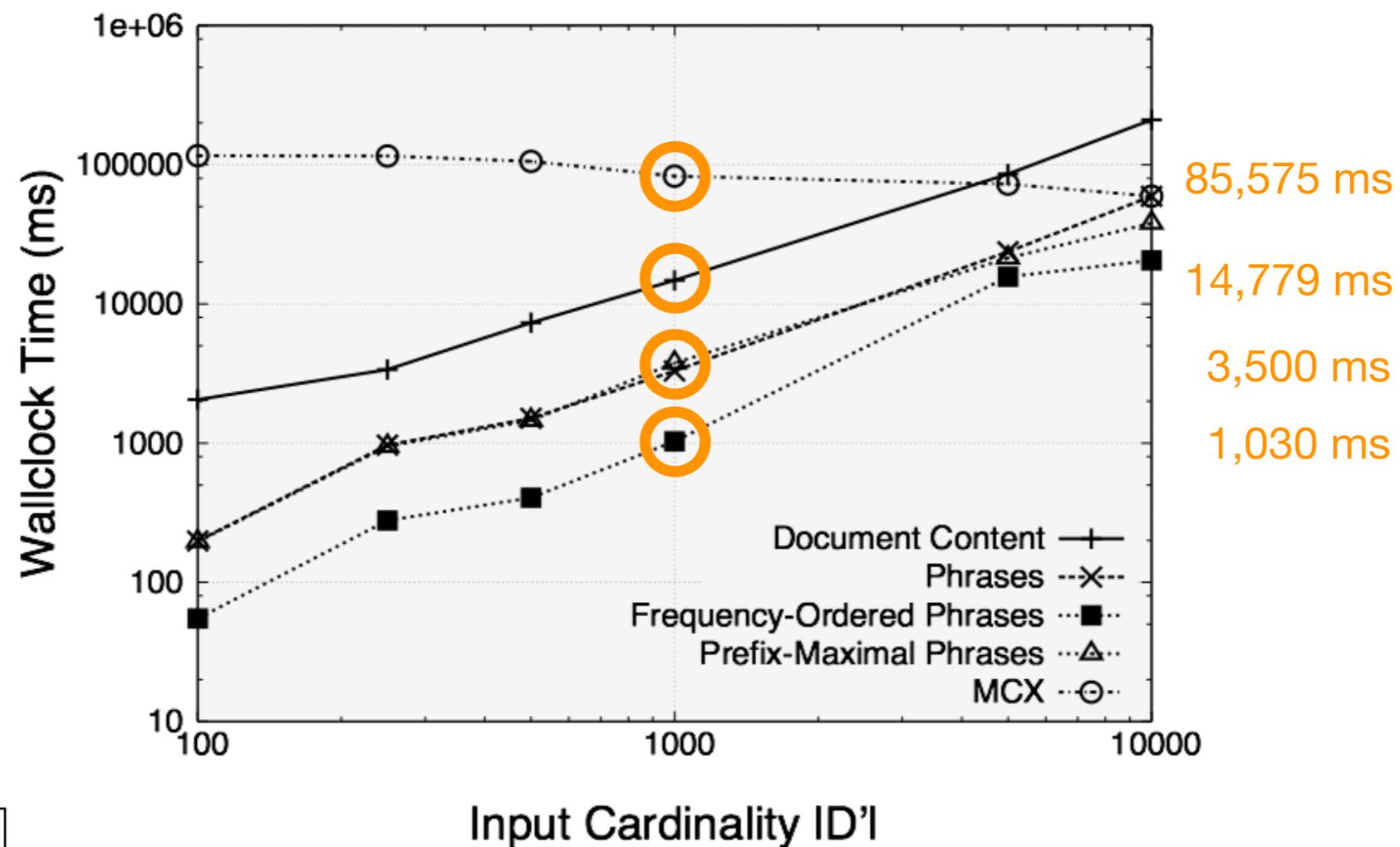
Experiments

- Dataset: The New York Times Annotated Corpus consisting of **1.8 million newspaper articles** published in 1987–2007



Experiments

- Dataset: The New York Times Annotated Corpus consisting of **1.8 million newspaper articles** published in 1987–2007



$k = 100$
 $\tau = 10$

Anecdotal Results

- Query: **john lennon**

- 1) ...since john lennon was assassinated...
- 2) ...lennon's childhood...
- 3) ...post beatles work...

- Query: **bob marley**

- 1) ...music of bob marley...
- 2) ...marley the jamaican musician...
- 3) ...i shot the sheriff...

- Query: **john mccain**

- 1) ...to beat al gore like...
- 2) ...2000 campaign in arizona...
- 3) ...the senior senator from virginia...

Summary

- **Clustering** groups similar documents; *k*-Means can be implemented efficiently by leveraging established IR methods
- **Faceted search** uses orthogonal sets of categories to allow users to explore/navigate a set of documents (e.g., query results)
- **Mememes can be tracked** and allow for insightful analyses of media attention and time lag between traditional media and blogs
- **Timelines** identify significant time-varying features in a set of documents (e.g., query results) and visualize them
- **Interesting phrases** provide insights into query results; they can be determined efficiently by using a suitable index organization

References

- [1] **A. Broder, L. Garcia-Pueyo, V. Josifovski, S. Vassilvitskii, S. Venkatesan:** *Scalable k-Means by Ranked Retrieval*, WSDM 2014
- [2] **S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, G.:** *Interesting-Phrase Mining for Ad-Hoc Text Analytics*, PVLDB 2010
- [3] **Z. Dou, S. Hu, Y. Luo, R. Song, J.-R. Wen:** *Finding Dimensions for Queries*, CIKM 2011
- [4] **M. Hearst:** *Clustering Versus Faceted Categories for Information Exploration*, CACM 49(4), 2006
- [5] **J. Leskovec, L. Backstrom, J. Kleinberg:** *Meme-tracking and the Dynamics of the News Cycle*, KDD 2009
- [6] **R. Swan and J. Allan:** *Automatic Generation of Timelines*, SIGIR 2000
- [7] **K.-P. Yee, K. Swearingen, K. Li, M. Hearst:** *Faceted Metadata for Image Search and Browsing*, CHI 2003