Web Dynamics

Part 6 – Searching the Past

6.1 Time-travel problems

6.2 Efficient Time-Travel Search

6.3 Temporal measures of page importance

Time Travel Problems on the Web

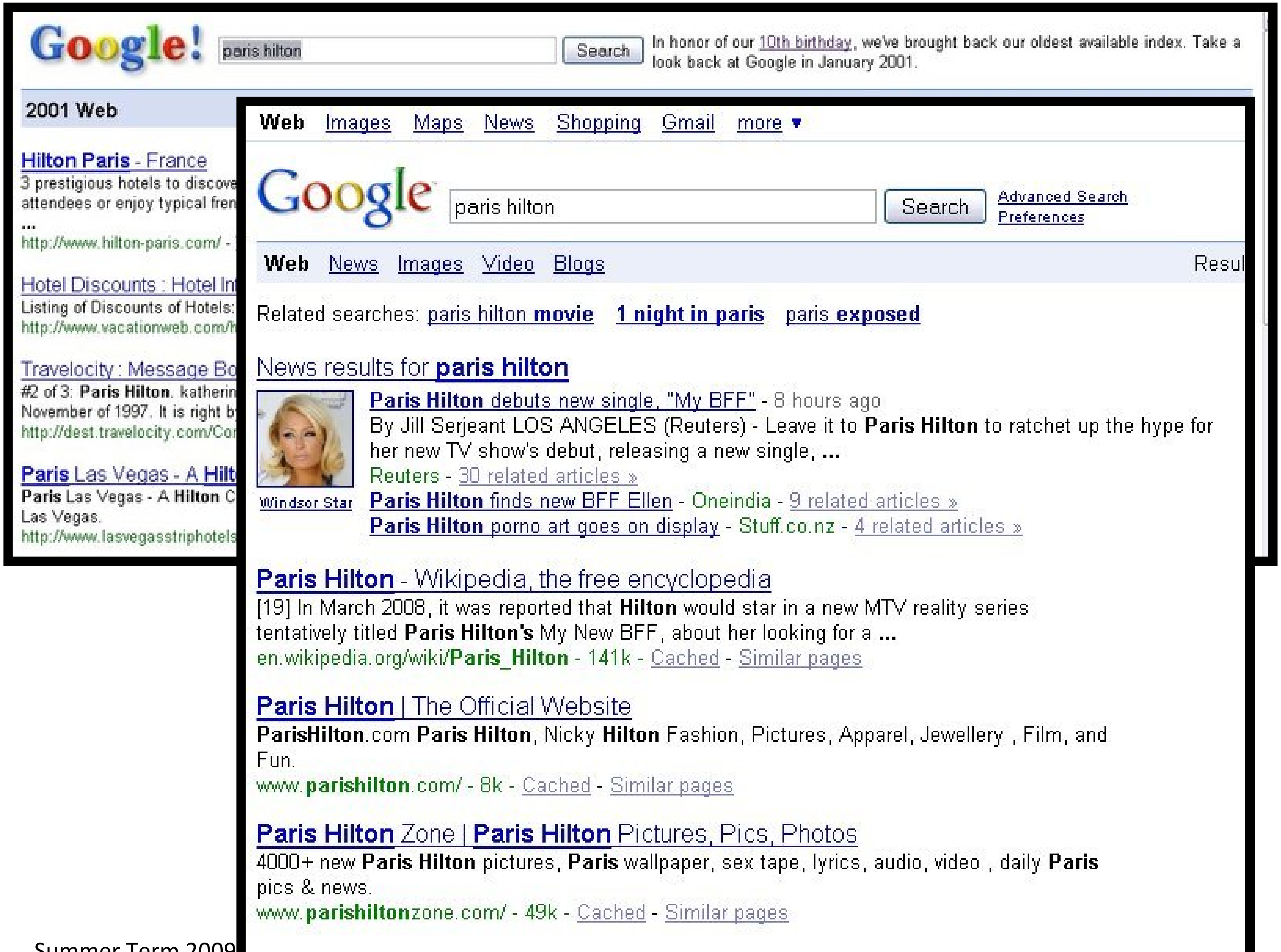
Search engines index only the ***current*** Web

But: Many interesting aspects on the ***historical*** Web:

- 5.2 Search the Web as of a specific time in the past („opinions of major US politicians on the Iraq War in 2002")
- 5.3 Analyze the Web as of a specific time in the past („most authoritative news page in 2002")
- Analyze temporal development of the Web („since when have political blogs been around?")

Web Archives don't provide these functionalities
(at least not publicly)

Rare example: Google@2001



http://www.techtalkz.com/blog/google/time-travel-search-google-in-2001.html

Web Dynamics

Part 6 – Searching the Past

6.1 Time-travel problems

6.2 Efficient Time-Travel Search

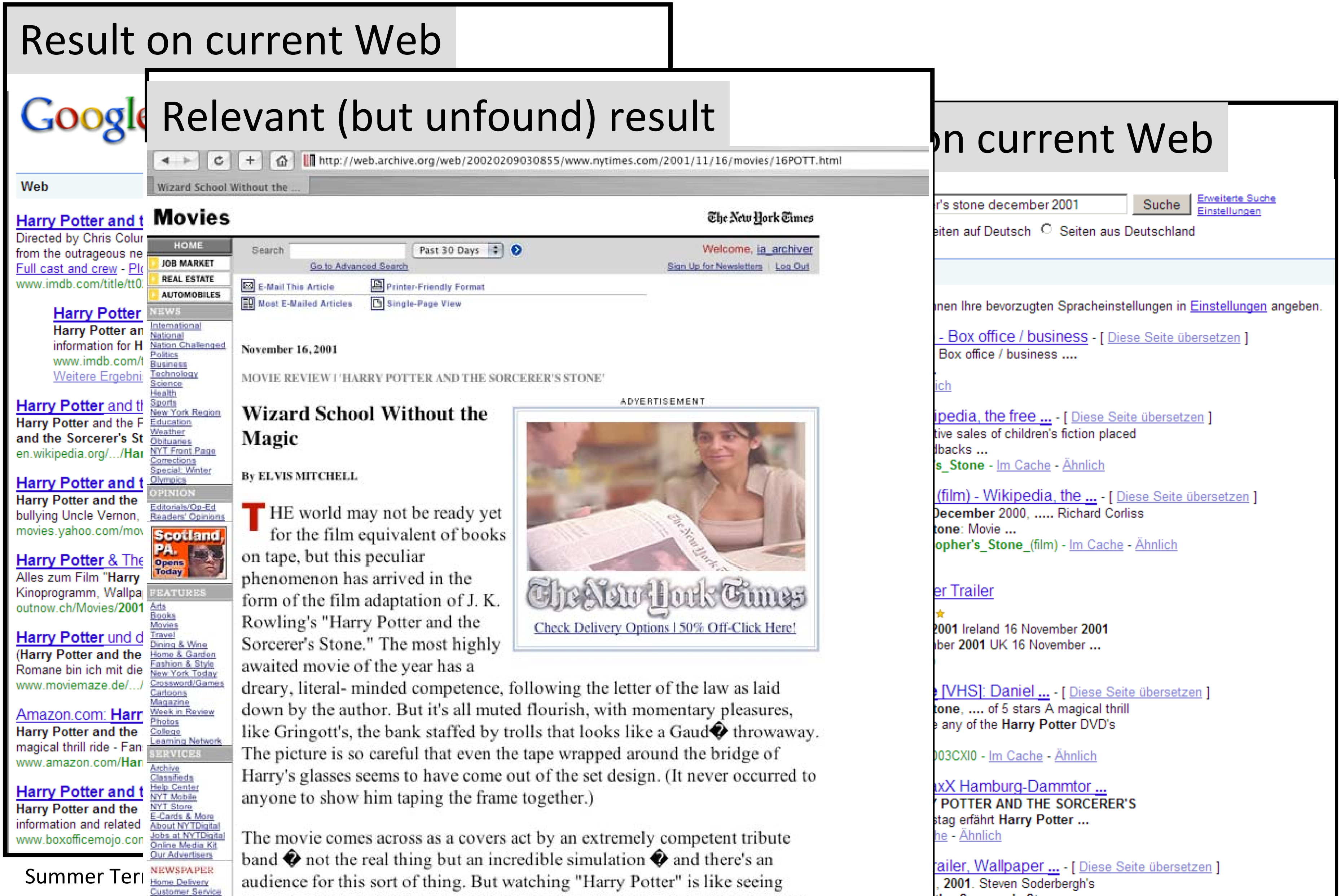
6.3 Temporal measures of page importance

(Some of the slides were contributed by Klaus Berberich)

The Need for Time-Travel Search

- Historical information needs, e.g.,
  - Contemporary (~2001) articles about the movie "Harry Potter and the Sorcerer's Stone"
  - Search for prior art for a patent submitted 2005
  - Links to some illegal content before Feb 2009
- Relevant pages disappeared in the current Web, but preserved by Web archives (e.g., archive.org)
- Search in existing Web archives limited and ignores the time-axis

The Need for Time-Travel Search



Summer Ter

Time-Travel Search Beyond the Web

More versioned document collections:

- Wikis (like Wikipedia)
- Repositories (e.g., controlled by CVS, Subversion)
- Your Desktop

Formal Model: Document Versions

Assume continuous time dimension $T=[0\ldots\infty($.

For each document (=url) d, maintain set of different ***versions*** $V(d)$, where each $v \in V(d)$ is a tuple $v=(c_v, [s_v,e_v])$, with $e_v=\infty$ for current versions.

content of v

lifetime of v

Different versions of the same document have ***disjoint lifetimes*** $\Rightarrow (d,s_v)$ identifies version

Archive can only estimate versions of a document

Time-Travel Keyword Queries

Time-travel keyword query $q=(k,I)$ combination of

- standard keyword query $k=(k_1, \ldots k_n)$
- time-of-interest interval $I=(s_I,e_I)$

Two important subclasses:

- ***Point-in-time*** queries: $s_I=e_I$ ← our focus
- ***Interval*** queries: $e_I>s_I$

Example:

"harry potter" @ 2001/11/14

This is a point-in-time query if the granularity of time is 1 day!

Scoring Point-in-Time Time-Travel Queries

Reminder: score in standard text retrieval:

$$s(d,q) \propto \sum_{k \in q} tf(d,k) \cdot idf(k)$$

frequency of k in d

importance of k → $idf(k) \propto \frac{N}{df(k)}$

score of version $v=(c_v,[s_v,e_v])$ for $q=(\{k_1 \ldots k_n\},t)$

$$s_T(v,q) \propto \begin{cases} 0 & \text{if } t \notin [sv,ev] \\ \sum_{k_i} tf(c_v,k_i) \cdot idf(k_i,t) & \text{if } t \in [sv,ev] \end{cases}$$

frequency of k_i in c_v

importance of k_i at query time t → $idf(k,t) \propto \frac{N(t)}{df(k,t)}$

N: # docs; N(t): #docs at time t
df(k): # docs with term k
df(k,t): # docs with term k at time t

Inverted Lists in Text IR

Reminder: Inverted Lists in text retrieval

For each term *t*, keep list *(d,score(d,t))* of documents containing term t and their score, in some order

List for term t in score order	List for term t in document order
d1,0.9	**d1,0.9**
d7,0.85	**d2,0.84763**
d2,0.84763	**d4, 0.27**
d119,0.79	**d7,0.85**
…	**…**

Query processing using merge joins of these lists (plus optional top-k for efficiency)

Extension for time-travel: SOPT

1. Split score in tf and idf component (idf is query-dependent!) ← store this somewhere else
2. For each term k, keep list $(v, tf(v,k), (s_v, e_v))$ of document versions containing term k, their tf value, and their lifetime, in some order

List for term k in score order

~~d1,90,(2001/jan/01,2001/jan/15)~~
~~d1,90,(2001/jan/16,2001/feb/28)~~
d7,85,(2004/aug/14,2004/aug/16) ✓
d1,84,(2001/mar/01,∞) ✓
...

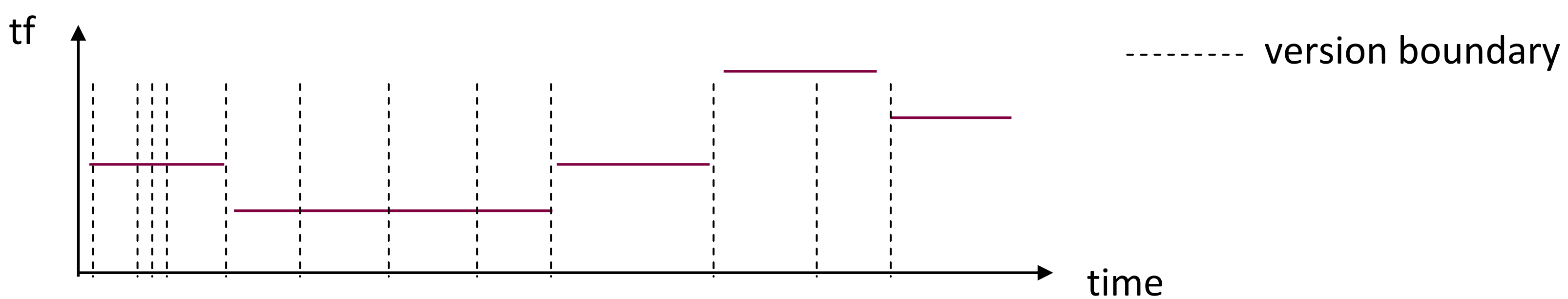
Example:
k@2004/aug/15

Query processing using merge joins of these lists plus ignoring versions where lifetime does not match query

This is not good enough

Major problems of this simple approach:

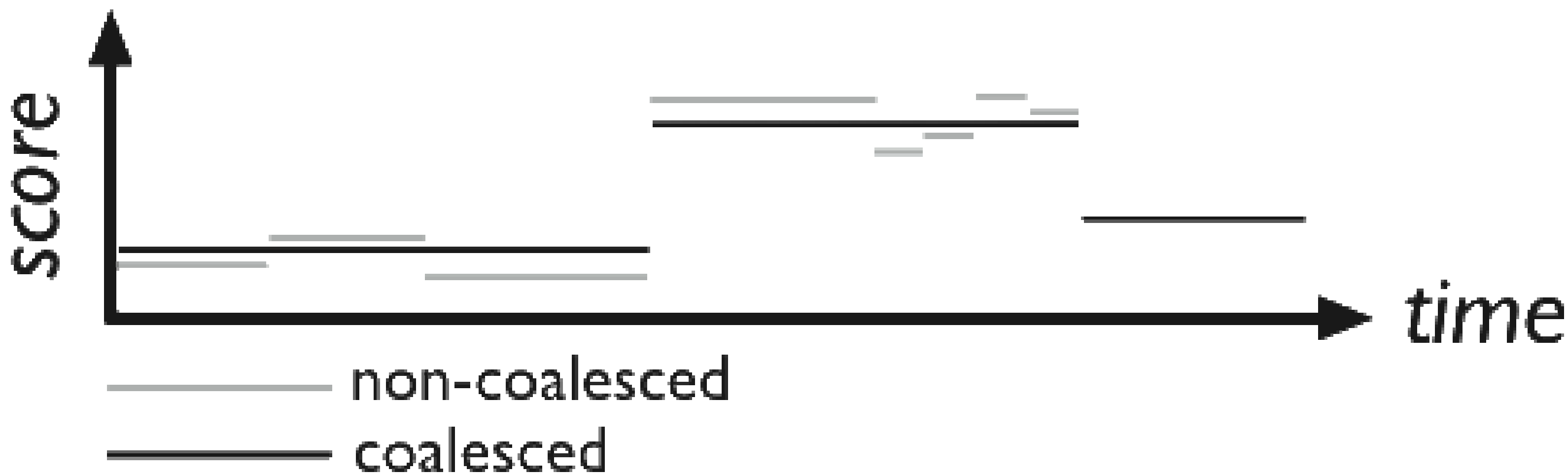
- index size explodes (***one index entry per version per term***)
  $\Rightarrow$ for Wikipedia alone: $9 \cdot 10^9$ entries!
- Many entries
  - differ only in their lifetimes
  - have almost identical tf values (hardly matters for ranking)



Reducing Index Size: Coalescing

Idea:

Coalesce sequences of temporally adjacent postings having similar scores

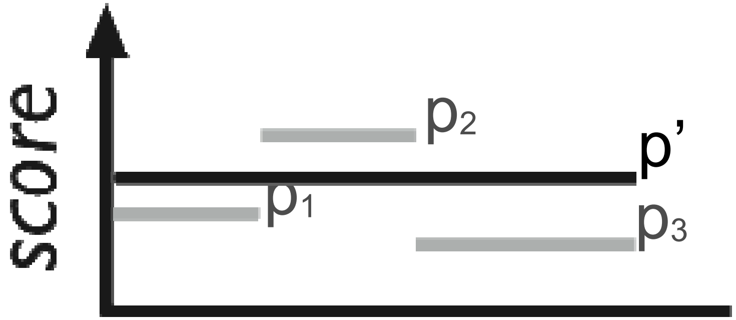


Can drastically reduce index size
But: what happens to result quality?

Formal Optimization Problem

Problem Statement:

Given input sequence I find a ***minimal length output sequence*** O with approximation errors bounded by a threshold ε



Guarantee:

$|p' - p_i| / |p_i| \leq \varepsilon$

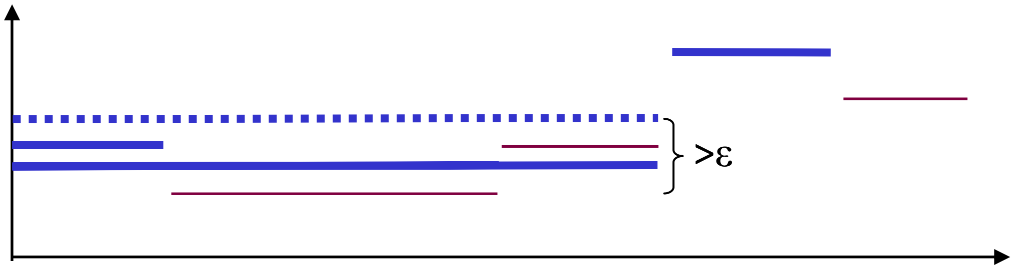
Approximate Temporal Coalescing (ATC):

finds an optimal output sequence using a greedy linear time algorithm

Approximate Temporal Coalescing (ATC)

General approach:

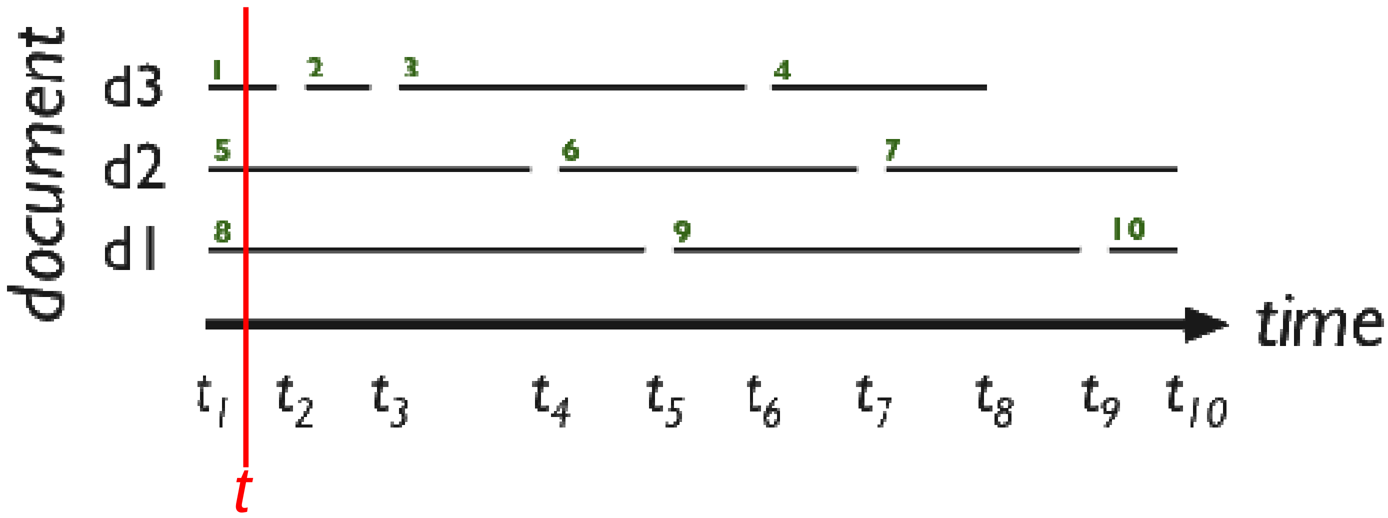
- Scan from left to right
- Maintain current estimate for representative p‘
- When next value is encountered, check if it can be represented within the error margin
  - If not, close current subsequence



Tuning query performance

Problem:

Many postings are ***ignored*** during query processing

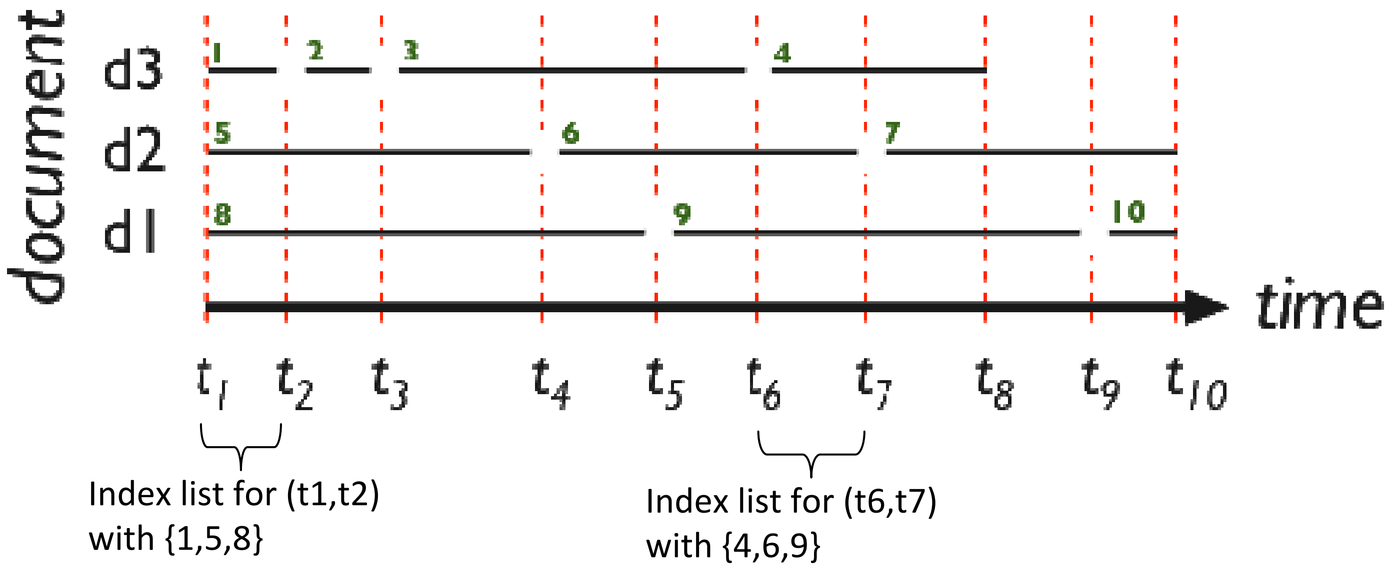


We read 10 postings,
but only {1, 5, 8} are needed

Tuning Query Performance: POPT

Idea:

Materialize smaller sublists containing only postings that ***overlap*** with a smaller interval



Maintaining a sublist for each elementary interval yields optimal query performance

Tuning Index Performance

Two extreme solutions up to now:

- ***space-optimal***: keep only a single list (SOPT)
- ***performance-optimal***: keep one list per elementary time-interval (POPT)

Now: two systematic techniques to trade-off space and performance

- ***performance-guarantee***: consumes minimal space while retaining a performance guarantee (PG)
- ***space-bound***: achieves best performance while not exceeding a space limit (SB)

Performance Guarantee (PG)

- consumes ***minimal*** space
- ***guarantees*** that for any t ***at most*** $\gamma \cdot n_t$ ***postings are read*** where n_t is the number of postings that exist at time t

Optimal solution computable for discrete time by means of induction (on the number of time points) ***in*** $O(T^2)$ ***time and*** $O(T^2)$ ***space*** (where T is the number of distinct timestamps in the list)

- start with elementary intervals (length 1)
- compute optimal solution for intervals of length k+1 from solutions for intervals of length≤k

Space Bound (SB)

- achieves ***minimal expected processing cost*** (i.e., expected length of the list that is scanned)
- consumes ***at most*** $\kappa \cdot n$ ***space*** where n is the length of the original list

Optimal solution computable using dynamic programming in $O(n^4)$ ***time*** and $O(n^3)$ space

Approximate solution computable in $O(T^2)$ ***time*** and $O(T)$ space using simulated annealing

Experimental Evaluation: Setup

Implementation:

Java, Oracle 10g

Datasets:

- WIKI: Revision history of English Wikipedia (2001-2005)
  892K documents / 13,976K versions / 0.7 TBytes
- UKGOV: Weekly crawls of 11 .gov.uk sites (2004-2005)
  502K documents / 8,687K versions / 0.4 TBytes

Queries:

- 300 keyword queries from AOL query log that most frequently produced a result click on en.wikipedia.org / .gov.uk
- Each keyword query is assigned one time point per month in the collection’s lifespan (18K / 7.2K time-travel queries in total)

Experimental Evaluation: Setup

Implementation:

Java, Oracle 10g

Datasets:

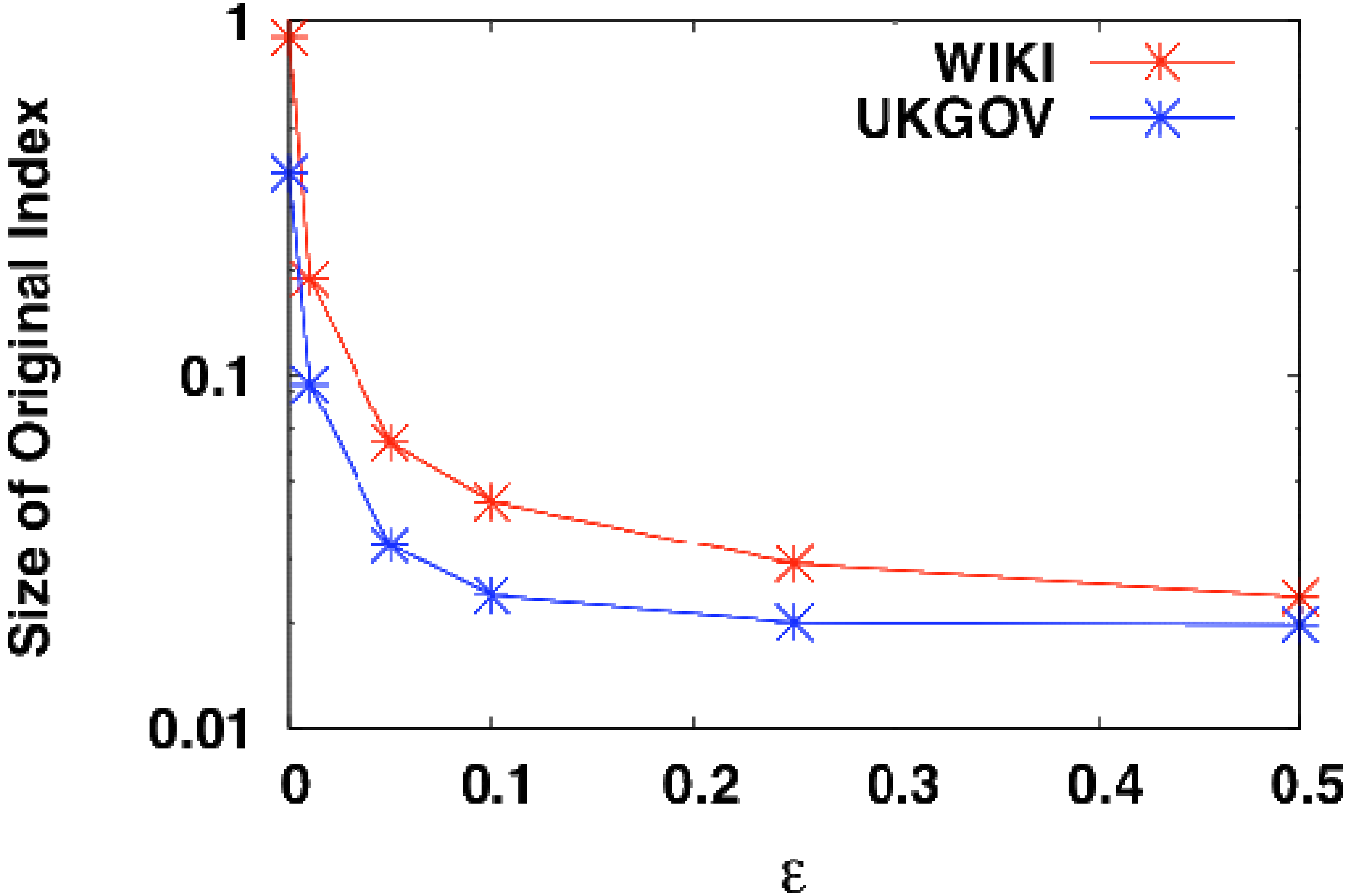
- WIKI: Revision history of English Wikipedia (2001-2005)
  892K documents / 13,976K versions / 0.7 TBytes
- UKGOV: Weekly crawls of 11 .gov.uk sites (2004-2005)

WIKI:
ten commandments, abraham lincoln, da vinci code, harlem renaissance…

UKGOV:
1901 uk census, british royal family, migrant worker statistics, witness intimidation…

Approximate Temporal Coalescing

Indexes computed for different values of threshold ε



At the same time provides excellent result quality

Sublist Materialization - Setup

Start with index created by ATC for $\varepsilon = 0.10$

For terms in query workloads (422/522) apply

- SOPT and POPT
- PG for γ varying between 1.10 and 3.00
- SB for κ varying between 1.10 and 3.00

Report

- Space, i.e., total number of postings in materialized sublists
- Expected Processing Cost (EPC), i.e., expected length of scanned list for random term and time

Performance Guarantee

	WIKI		UKGOV	
	Space	EPC	Space	EPC
P_{OPT}	14,428%	100%	11,406%	100%
S_{OPT}	100%	963%	100%	147%

Performance Guarantee

	WIKI		UKGOV	
	Space	EPC	Space	EPC
$\gamma = 1.10$	1,004%	106%	616%	103%
$\gamma = 1.50$	295%	132%	233%	117%
$\gamma = 2.00$	195%	160%	163%	125%
$\gamma = 3.00$	145%	207%	132%	133%

EPC = Expected Processing Cost

Space Bound

	WIKI		UKGOV	
	Space	EPC	Space	EPC
P_{OPT}	14,428%	100%	11,406%	100%
S_{OPT}	100%	963%	100%	147%

Space Bound

	WIKI		UKGOV	
	Space	EPC	Space	EPC
κ = 3.00	288%	139%	273%	107%
κ = 2.00	194%	171%	180%	119%
κ = 1.50	146%	214%	131%	131%
κ = 1.10	109%	406%	104%	145%

EPC = Expected Processing Cost

Web Dynamics

Part 6 – Searching the Past

6.1 Time-travel problems

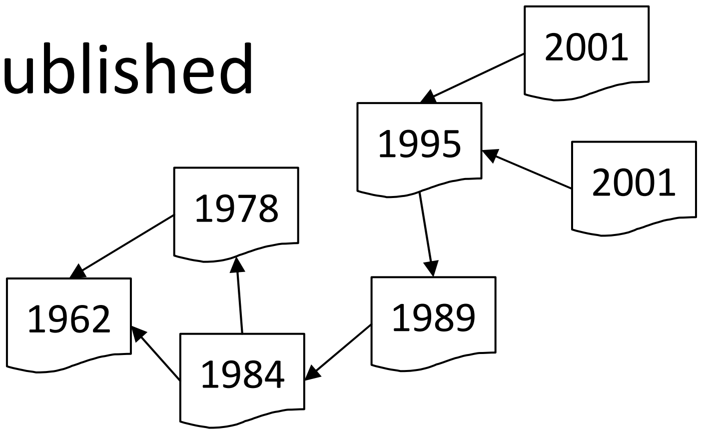
6.2 Efficient Time-Travel Search

6.3 Temporal measures of page importance

Differences between Citations and Links

- Citations in printed documents (papers)
  - never change once paper is published
  - mostly to recent documents

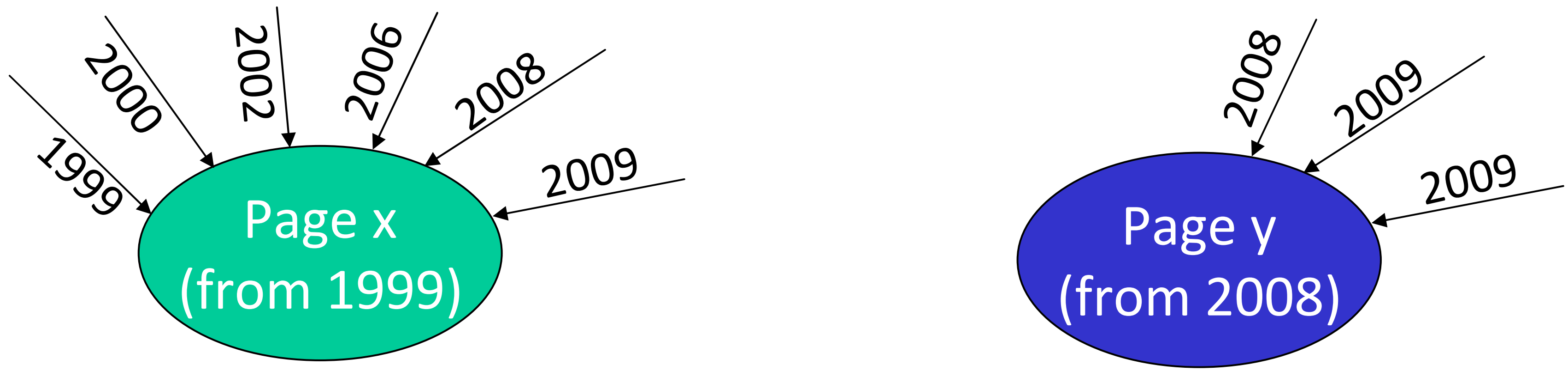
  ⇒ Old papers hardly cited, **negative authority bias**



- Links on the Web
  - frequently change after page is published
  - old (but updated!) pages still get many new links

  ⇒ Old pages have **positive authority bias**

Temporal Development of Links



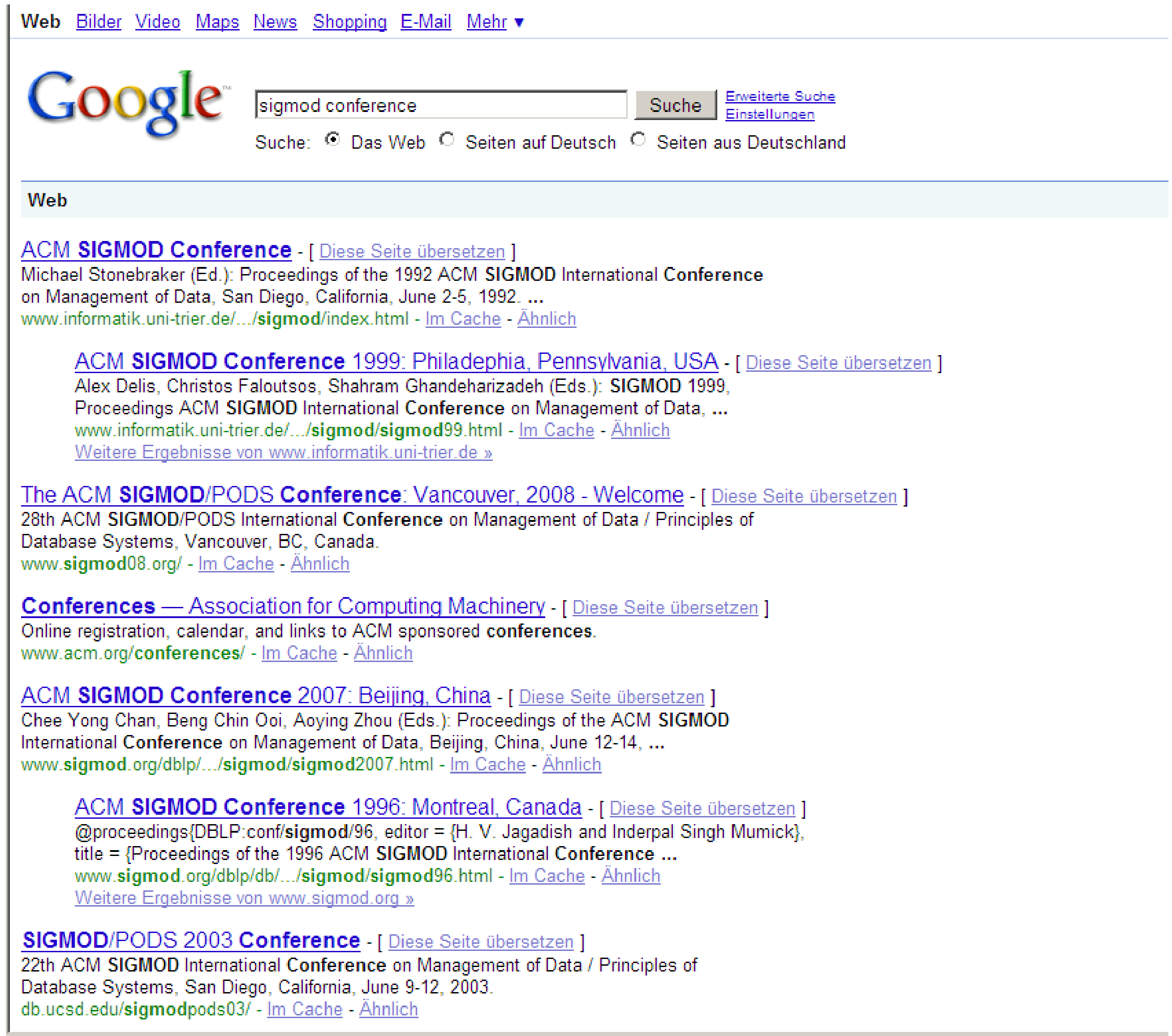
- PageRank (HITS, …): x more authoritative than y
- But:
  - x has 6 links in 10 years
  - y has 3 links in 2 years

  ⇒ y a lot more dynamic and up-to-date than x,
  but difficult to beat x's "temporal advantage"
- Whi … 1999?

Temporal notions of authority required!

Example: Search for SIGMOD conference

Old pages dominate over page for 2009 conference



Modelling Temporal Changes

For each page p, maintain

- timestamp of creation $TS_C(p)$
- timestamp of deletion $TS_D(p)$
- set of timestamps of modifications $TS_M(p)$

(timestamp: amount of time units since time 0)

Analogous definitions for link (x,y):

- timestamp of creation $TS_C(x,y)$: time when (x,y) added
- timestamp of deletion $TS_D(x,y)$: time when (x,y) del'ed
- timestamp $TS(x,y)$: last modification time of page x

Timestamped Link Profile (TLP)

Goal: Measure the „activity“ of a topic on the Web

$\Rightarrow$ Construction of ***Timestamped Link Profile***:

- Collect set of Web pages for the topic (e.g., by collecting results of keyword queries)
- Collect set of inlinks *(x,y)* to these pages (provided by search engines: **`link:url`**)
- Compute temporal distribution of timestamps of inlinks (partitioning time range into intervals)

Based on *limited sample* of the inlinks

Timestamps usually available for some inlinks only (last-modified timestamp of page)

Example TLP

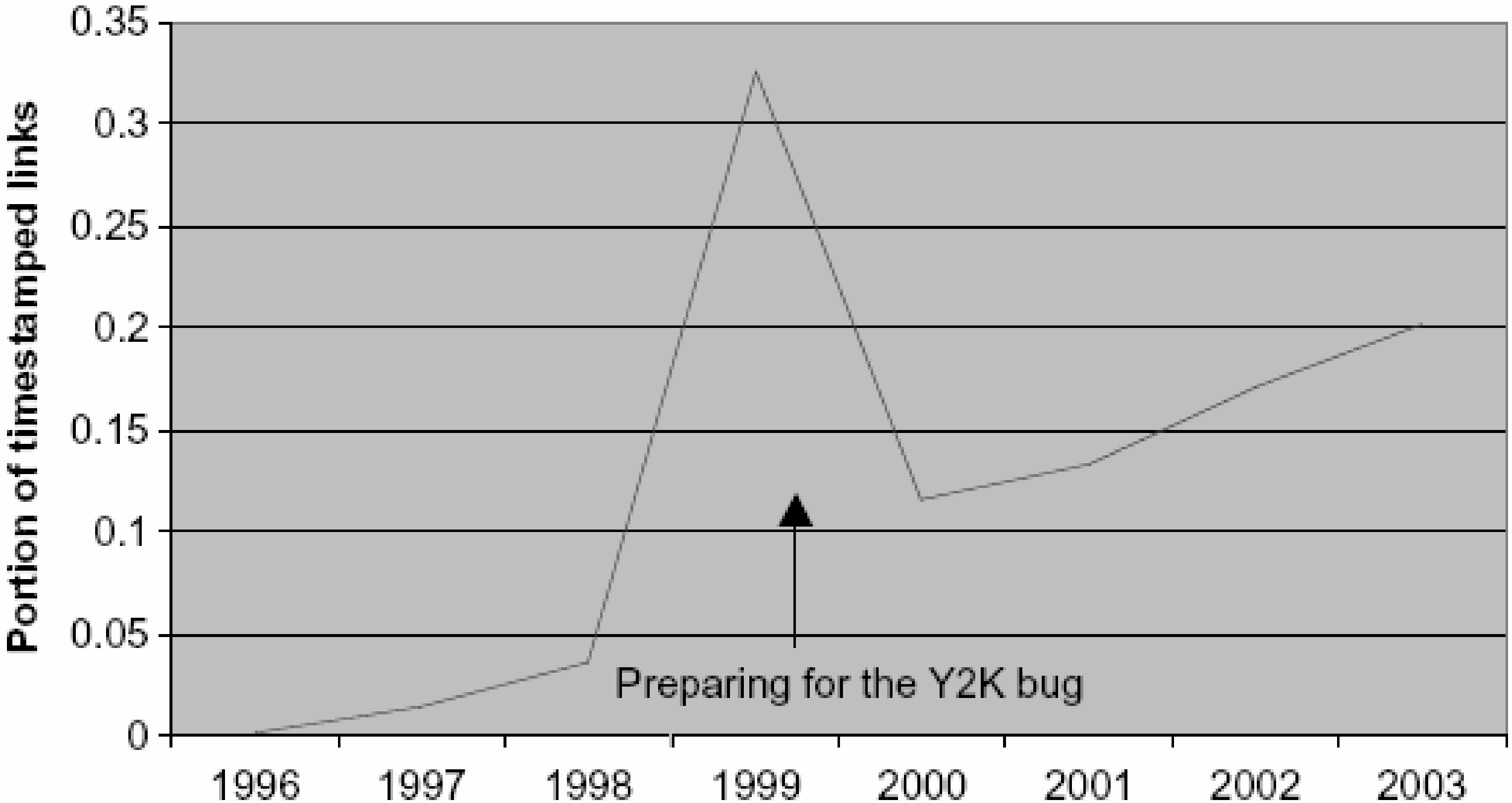


FIG. 2. TLP for the theme "Year 2000 Bug" (~750 timestamped links).

Amitay et al., JASIST 2004

Towards Timely Authorities

Goal: Determine ***currently*** authoritative pages
(opposed to those authoritative years ago,
but still around)

Intuition of [Amitay et al.]:

- Deviate from uniform link weight in HITS etc
- Give more weight to recent links:

$$weight(x,y) \propto 1/age(x,y)$$
$$= 1/(currentTime - TS(x,y))$$

(with linear or exponential decay)

Authoritative Pages in the Past

Goal: extend this approach towards

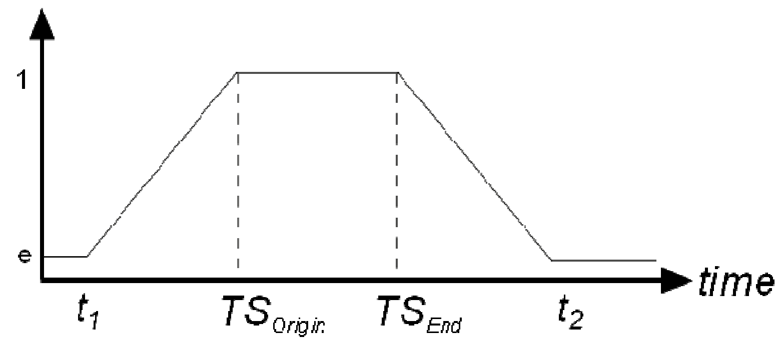
- finding important pages at ***any interval*** in the past
- including ***page activity*** as quality measure

Consider ***interval of interest*** $ti=[TS_{Origin},TS_{End}]$ with additional ***tolerance interval*** $[t1,t2]$ where pages are less interesting, but still relevant to user ($t1 \leq TS_{Origin}$, $t2 \geq TS_{End}$)

Freshness

Freshness measures relevance of timestamp to interval of interest:

$$f(ts) = \begin{cases} \text{if } TS_{Origin} \le ts \le TS_{End}: & 1 \\ \text{if } t_1 \le ts < TS_{Origin}: & \dfrac{1-e}{TS_{Origin} - t_1} \cdot (ts - t_1) + e \\ \text{if } TS_{End} < ts \le t_2: & \dfrac{e-1}{t_2 - TS_{End}} \cdot (ts - TS_{End}) + 1 \\ \text{otherwise}: & e \end{cases}$$



Freshness of node *x*: $f(x) = f(TS(x))$

Freshness of edge *(x,y)*: $f(x,y) = f(TS(x,y))$

Activity

Activity of set TS of timestamps measures frequency of change with respect to interval of interest:

$$a(TS) = \begin{cases} if\ \ \mathrm{TS} \cap [t_1, t_2] \neq \emptyset : & \sum_{t1}^{t2} \{f(ts) | ts \in TS\} \\ otherwise : & e \end{cases}$$

Activity of node *x*: $a(x) = a(TS_M(x))$

Activity of edge (*x,y*): $a(x,y) = a(TS_M(x,y))$

Restricting the Graph to an Interval

For graph G and interval of interest $ti=[ts,te]$ with tolerance interval $[t1,t2]$, consider ***time projection*** $G_{ti}=(V_{ti},E_{ti})$ of $G=(V,E)$:

$$V_{ti}=\{v \in V \mid TS_C(v) \le t_2 \wedge TS_D(v) \ge t_1\}$$

$$E_{ti}=\{(x,y) \in E \mid (x,y) \in V_{ti} \times V_{ti} \wedge TS_C(x,y) \le t_2 \wedge TS_D(x,y) \ge t_1\}$$

Special case $t_1=t_2$: G_{ti} snapshot of G as of time t_1

Towards Temporal PageRank

Standard definition of PageRank:

$$r(y) = \sum_{(x,y)\in E} (1-\varepsilon) \cdot \frac{r(x)}{\text{outdegree}(x)} + \frac{\varepsilon}{n}$$

Generalized version allowing for ***non-uniform*** transition and random jump probabilities:

$$r(y) = \sum_{(x,y)\in E} (1-\varepsilon) \cdot t(x,y) \cdot r(x) + \varepsilon \cdot s(y)$$

- ***t(x,y)*** describes transition probabilities
- ***s(y)*** describes random jump probabilities

Temporal Pagerank (T-Rank)

- **Modified *PageRank*** on G_{ti}
- Transition probabilities $t(x,y)$ depend on **freshness** of nodes and edges
- Random jump probabilities depend on **freshness and activity** of nodes and edges

T-Rank – Transitions

- Transitions favor ***fresh*** nodes/edges
- Coefficients w_{ti}: probabilities that random surfer follows *(x,y)* with probabilities proportional to
  - freshness of node *y*
  - freshness of edge *(x,y)*
  - average (mean) freshness of incoming edges of node *y*

$$t(x,y) = w_{t1} \cdot \frac{f(y)}{\sum_{(x,z)\in E} f(z)} + w_{t2} \cdot \frac{f(x,y)}{\sum_{(x,z)\in E} f(x,z)} + w_{t3} \cdot \frac{avg\{f(\upsilon,y) \mid (\upsilon,y) \in E\}}{\sum_{(x,w)\in E} avg\{f(\upsilon,w) \mid (\upsilon,w) \in E\}}$$

T-Rank – Random Jumps

- Random jumps favor ***fresh and active*** nodes/edges
- Coefficients w_{si} probabilities that random surfer jumps to node *y* with probabilities proportional to
  - freshness and activity of node *y*
  - average (mean) freshness and activity of incoming edges of node *y*

$$s(y) = w_{s1} \cdot \frac{f(y)}{\sum_{z \in V} f(z)} + w_{s2} \cdot \frac{a(y)}{\sum_{z \in V} a(z)} +$$

$$w_{s3} \cdot \frac{avg\{f(\upsilon, y) \mid (\upsilon, y) \in E\}}{\sum_{z \in V} avg\{f(w, z) \mid (w, z) \in E\}} + w_{s4} \cdot \frac{avg\{a(\upsilon, y) \mid (\upsilon, y) \in E\}}{\sum_{z \in V} avg\{a(w, z) \mid (w, z) \in E\}}$$

T-Rank Experiment: DBLP

Digital Bibliography & Library Project (DBLP) freely available bibliographic dataset (as XML)

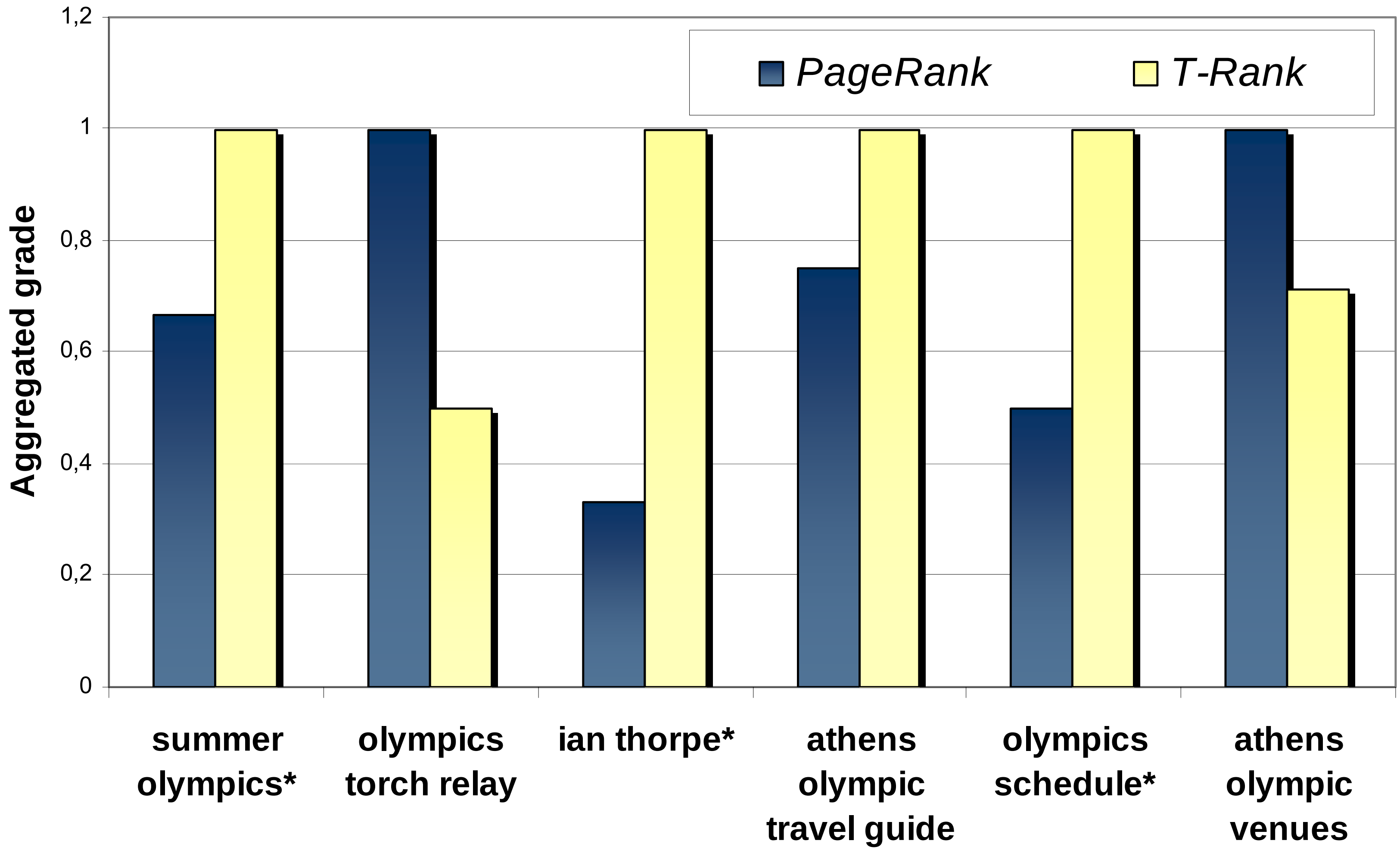
Evolving graph derived from DBLP: Authors as nodes, citations as edges

	***PageRank** 2000s*	***T-Rank** 2000s*
1	E. F. Codd	Jim Gray
2	Michael Stonebraker	Michael Stonebraker
3	Jim Gray	Jeffrey D. Ullman
4	Donald D. Chamberlin	Philip A. Bernstein
5	Jeffrey D. Ullman	Hector Garcia-Molina
6	Philip A. Bernstein	Jeffrey F. Naughton
7	Raymond A. Lorie	Donald D. Chamberlin
8	Morton M. Astrahan	David J. DeWitt
9	Kapali P. Eswaran	Jennifer Widom
10	John Miles Smith	Rakesh Agrawal

T-Rank Experiment: Web

- **Theme: Olympic Games 2004**
  - ~200K thematically related Web pages
  - 9 crawls in period July 26th to September 1st
- **Blind test** comparing *PageRank* and *T-Rank*
  - Users asked to **grade quality** of given top-*10* lists
  - Half of the queries drawn from Google Zeitgeist

T-Rank Experiment: Web



Berberich et al, Internet Mathematics 2006

References

Time-Travel Search:

- Klaus Berberich et al.: ***A Time Machine for Text Search***, SIGIR Conference, 2007
- Klaus Berberich et al.: ***FluxCapacitor: Efficient Time-Travel Text Search***, VLDB Conference, 2007

Temporal Link Analysis:

- L. Adamic & B.A. Huberman: ***The Web's hidden order***, CACM 44(9), 2001
- Einat Amitay et al.: ***Trend Detection Through Temporal Link Analysis***, Journal of the American Society for Information Science and Technology 55, pp. 1-12, 2004
- Ricardo Baeza-Yates et al.: ***Web Structure, Dynamics and Page Quality***, SPIRE Conference, 2002
- Klaus Berberich et al.: ***Time-Aware Authority Ranking***, Internet Mathematics 2(3), 2006
- Klaus Berberich et al.: ***A Pocket Guide to Web History***, SPIRE Conference, 2007
- Philip S. Yu et al.: ***On the Temporal Dimension of Search, WWW Conference***, 2004