7. Dynamics & Age

Outline

7.1. Dynamics & Age

7.2. Temporal Information

7.3. Search in Web Archives

7.4. Historical Document Collections

7.1. Dynamics & Age

- The Web is **highly dynamic**: new content is continuously added; old content is deleted and potentially lost forever
- Web archives (e.g., archive.org, internetmemory.org) have been preserving **old snapshots of web pages** since 1996
- **Improved digitization** (e.g., OCR) have allowed (newspaper) archives to make old documents (e.g., from 1700s) searchable
- Challenges & Opportunities:
  - *How to index highly redundant document collections like web archives?*
  - *How to make use of temporal information such as publication dates?*
  - *How to search documents written in archaic language?*

How Dynamic is the Web?

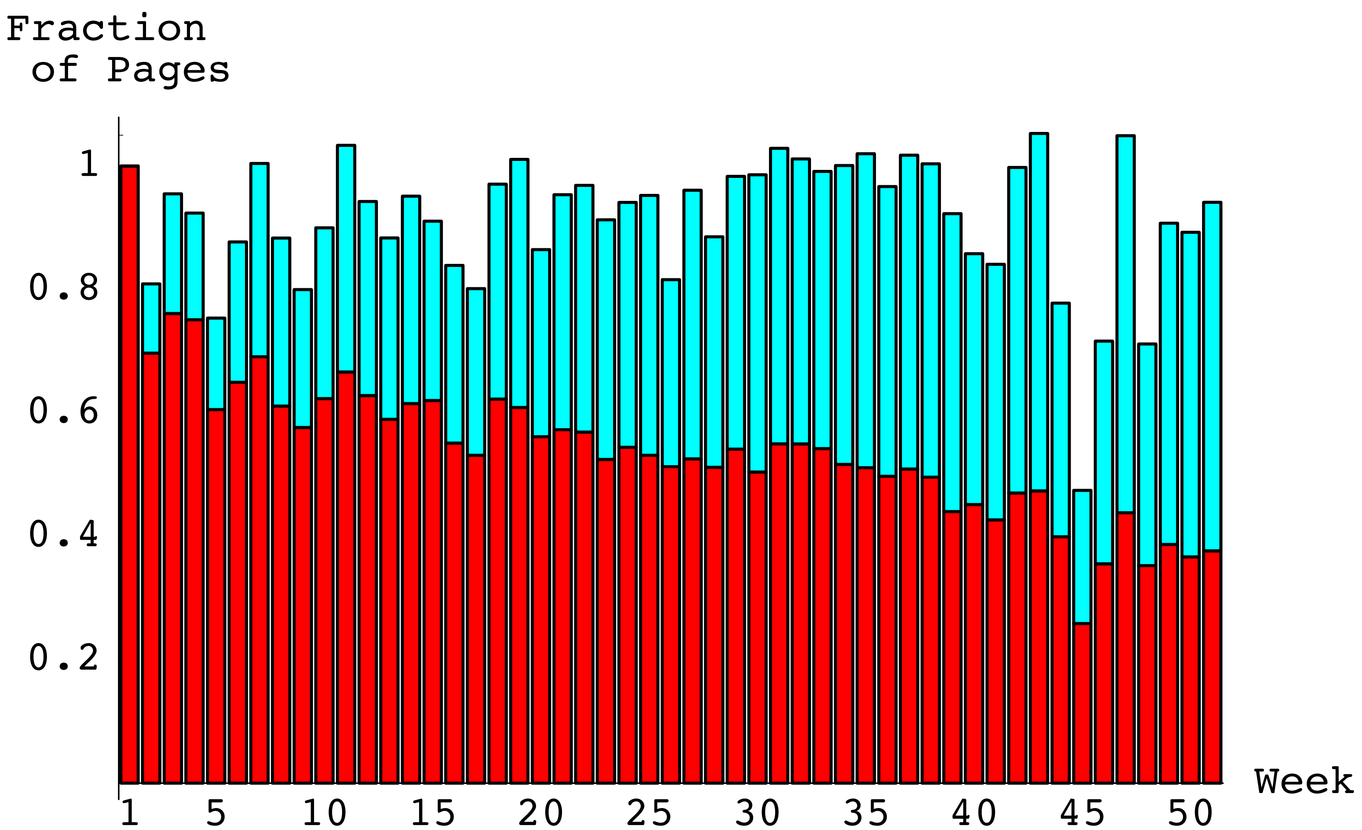
- Ntoulas et al. [9] study the dynamics of the Web in '02–'03

- Data: Weekly crawls of **154 web sites** over one year
  - **top-ranked web sites** from topical categories in Google Directory (extension of DMOZ) from different top-level domains
  - at most **200K web pages per web site** per weekly crawl

Domain	Fraction of pages in domain
.com	41%
.gov	18.7%
.edu	16.5%
.org	15.7%
.net	4.1%
.mil	2.9%
misc	1.1%

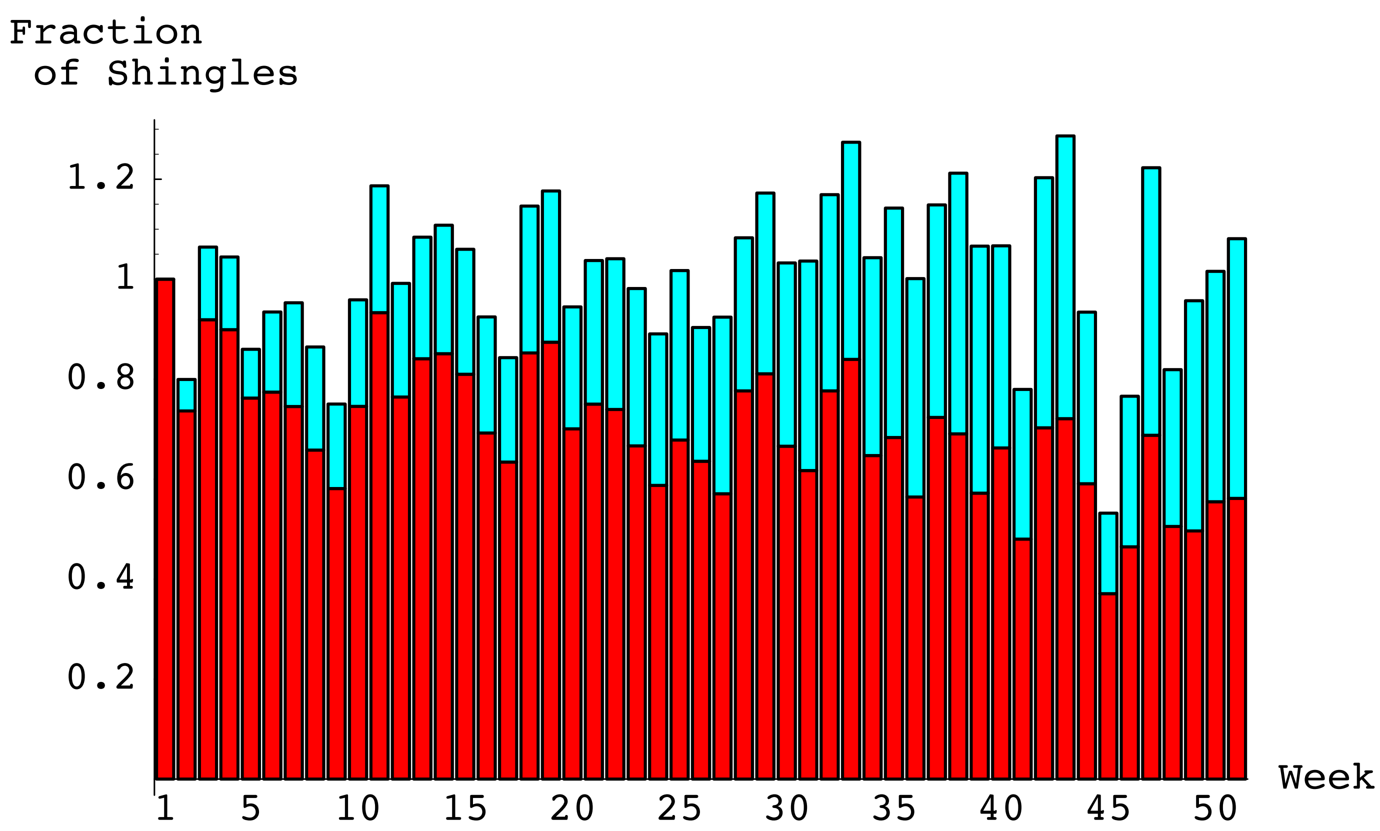
How Dynamic are Web Pages?

- Web pages:
  - on average **8% new web pages** per week
  - **peek** in creation of new pages **at the end of each month**
  - after **9 months** about **50% of web pages** have been deleted



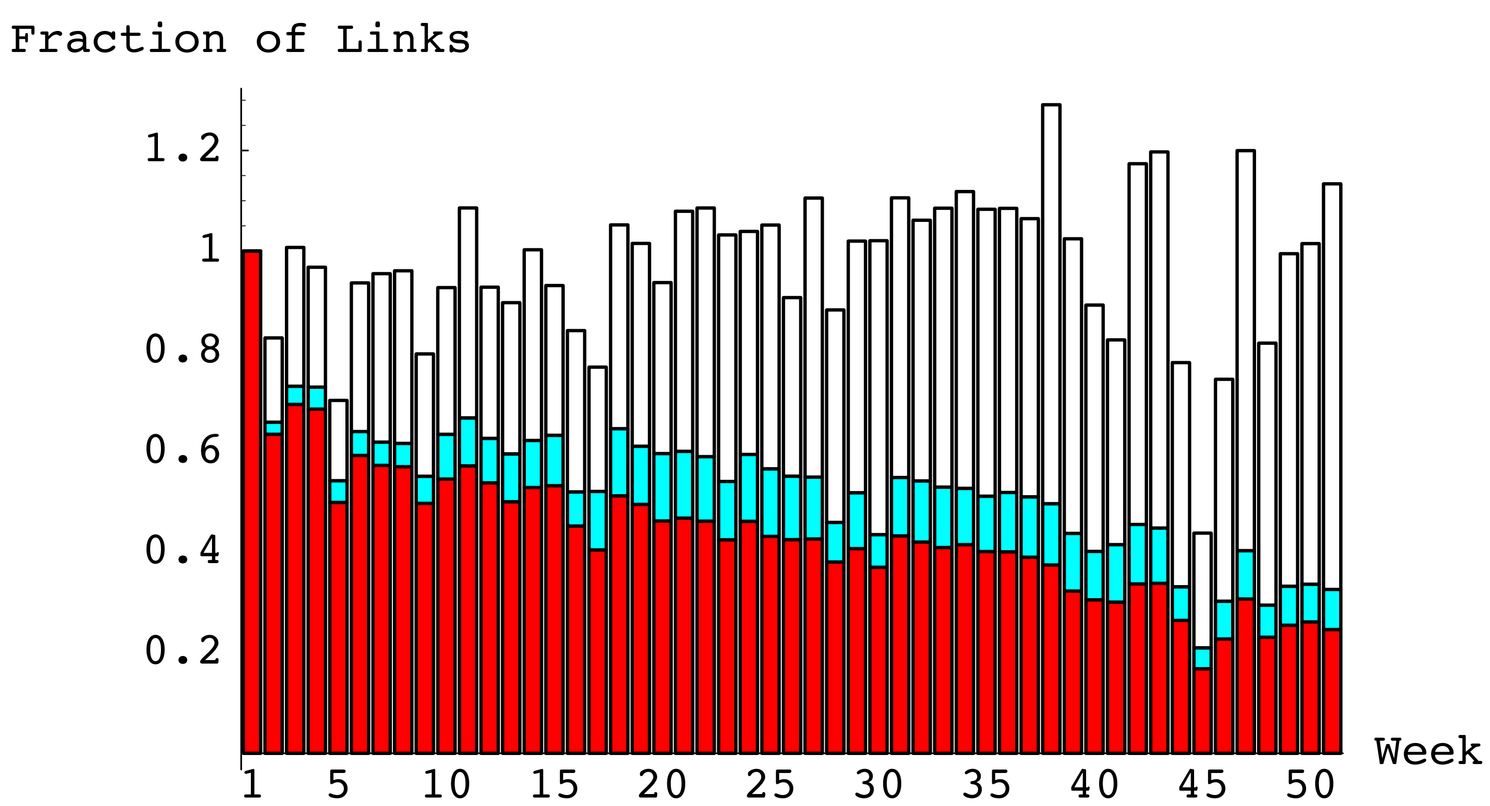
How Dynamic is the Content?

- Content: Based on *w*-shingles (contiguous sequence of *w* words)
  - **after one year** more than **50% of shingles** are still available
  - each week about **5% of new shingles** are created



How Dynamic is the Link Structure?

- Hyperlinks:
  - after one year **only 24% of links** are still available
  - on average **25% of new links** are created **every week**



How Dynamic is the (Visited) Web?

- Adar et al. [1] conducted a fine-grained study of the visited Web

- Data: **Hourly fetches** of **55K web pages** over 5 weeks
  - selected based on **access statistics** from Live Search toolbar
  - selection balances frequently visited and infrequently visited web pages
  - **more fine-grained fetches** for web pages with high change activity

How Dynamic are (Visited) Web Pages?

- Change of web page measured using
  - **average time** between changes (*Hours)* determined using content checksums
  - **average Dice coefficient** (*Dice*) between adjacent versions as word sets

$$D(W_i, W_j) = \frac{2 \cdot |W_i \cap W_j|}{|W_i| + |W_j|}$$

		Inter-version means	
		Hours	*Dice*
Total		123	.7940
Visitors	2	138	.8022
	3 - 6	125	.8268
	7 - 38	106	.8252
	39+	102	.8123
Domain	.gov	169	.8358
	.edu	161	.8753
	.com	126	.7882
	.net	125	.7642
	.org	95	.8518
URL depth	5+	199	.6782
	4	176	.7401
	3	167	.7363
	2	127	.7804
	1	104	.8200
	0	80	.8584
Category	Industry/trade	218	.6649
	Music	147	.8013
	Porn	137	.7649
	Personal pages	88	.8288
	Sports/recreation	66	.8975
	News/magazines	33	.8700

7.2. Temporal Information

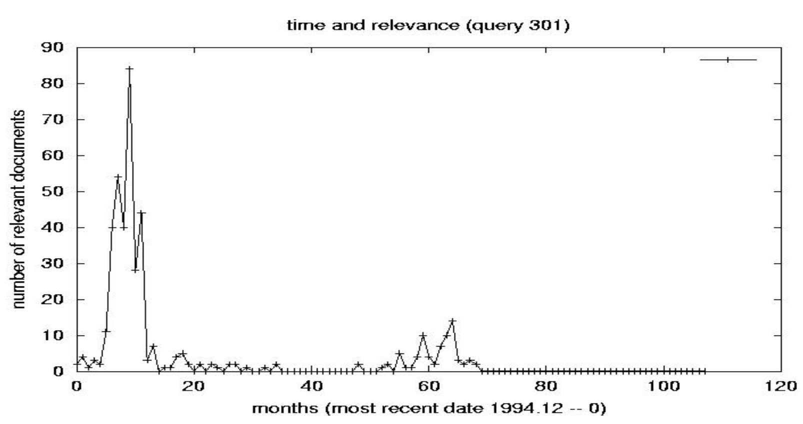
- Documents come with different kinds of **temporal information**
  - **publication dates** indicating when the document was published
  - **temporal expressions** (e.g., last month, January 9th 2014, in the '90s) indicating which time periods the document's content talks about
- Queries can be **temporally classified** along several dimensions
  - …whether they can refer to a single or multiple time periods
    - **temporally unambiguous** (e.g., fifa world cup 2014, battle of waterloo)
    - **temporally ambiguous** (e.g., summer olympics, world war)

Temporal Information

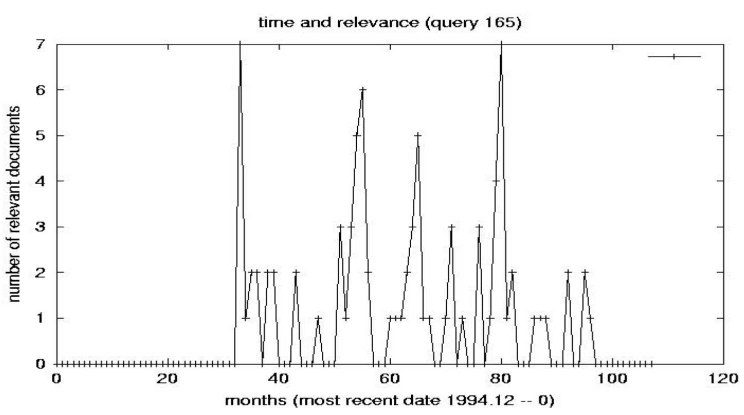
- …whether a time period is explicitly mentioned or implicitly assumed
  - **explicitly temporal** (e.g., fifa world cup 2014, presidential election 2008)
  - **implicitly temporal** (e.g., superbowl, london bombing)
- …whether they aim for information about the past, present, or future
  - **past** (e.g., historic map of rome, news reports about moon landing)
  - **recent** (e.g., paris terrorist attack, tesla stock price, lithuania euro)
  - **future** (e.g., lisa pathfinder launch, academy awards 2015)
- …whether they can refer to any time period at all
  - **atemporal** (e.g., muffin recipe, side effects of paracetamol, muscle cramps)

7.2.1. Temporal Document Priors

- Li and Croft [7] develop an approach based on language models targeted at **queries favoring more recent documents**
- Example: Publication dates of relevant documents in TREC



Query 301: international organized crime



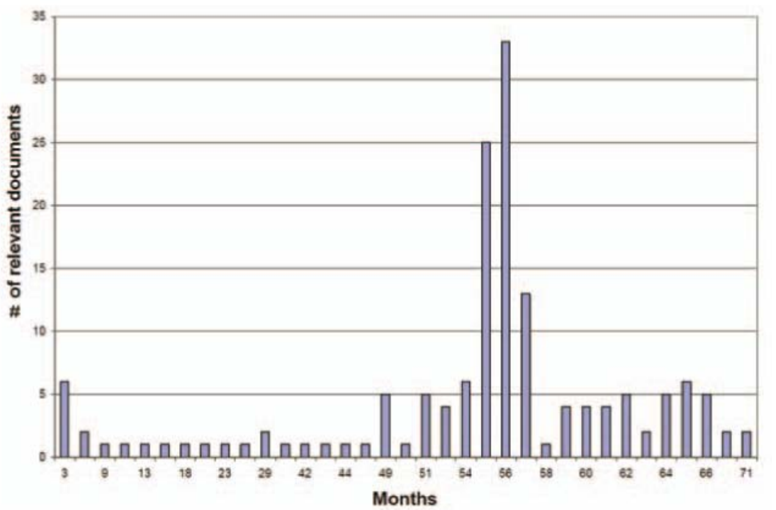
Query 165: tobacco company advertising and the young

- Query-likelihood approach with **temporal document prior** P[d] depending on publication date **t** of document and current date **c**

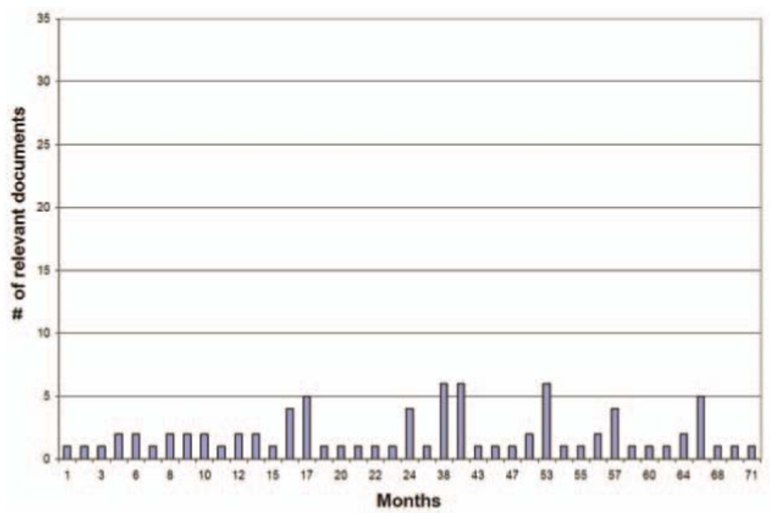
$$\mathrm{P}[\,d \mid q\,] \propto \mathrm{P}[\,d\,] \cdot \prod_{v} \mathrm{P}[\,v \mid d\,] \qquad \mathrm{P}[\,d\,] = \lambda\, e^{-\lambda\,(c-t)}$$

7.2.2. Temporal Query Profiles

- Dakka et al. [4] target **general time-sensitive queries** using an approach based on language models
- <u>Example</u>: Publication dates of relevant documents in TREC



Query 311: industrial espionage



Query 304: endangered species (mammals)

- <u>Idea</u>: Estimate **temporal document prior** from publication dates of **pseudo-relevant documents** retrieved for the query

Temporal Query Profiles

- Let R denote the set of **pseudo-relevant documents** (e.g., top-50 from baseline), a **temporal query profile** is estimated as

$$P[\,t \mid q\,] = \sum_{d \in R} P[\,t \mid d\,] \, \frac{P[\,q \mid d\,]}{\sum_{d' \in R} P[\,q \mid d'\,]} \qquad P[\,t \mid d\,] = \mathbb{1}(d \text{ published at } t)$$

- Temporal query profile is **smoothed in two ways**
  - using linear interpolation with the **temporal collection profile** to account for fluctuations in publication volume

$$P[\,t \mid D\,] = \frac{1}{|D|} \sum_{d \in D} P[\,t \mid d\,]$$

  - using a **moving average** to account for longer lasting events

$$\overline{P}[\,t \mid q\,] = \frac{1}{w} \sum_{i=0}^{w-1} P[\,t - i \mid q\,]$$

Temporal Query Profile

- Temporal query profile is integrated as **document prior** with **t** as the publication date of document **d**

$$\mathrm{P}\left[\, q \mid d \,\right] = \mathrm{P}\left[\, t \mid q \,\right] \cdot \prod_{v} \mathrm{P}\left[\, v \mid d \,\right]$$

7.2.3. Temporal Expressions

- Berberich et al. [3] develop an approach based on language models targeted at **explicitly temporal queries** that mention a **temporal expression** (e.g., michael jordan 1990s)

- Standard retrieval models treat **temporal expressions** as terms and are **unaware of their inherent semantics** (e.g., ‘90s is different from 1990s and 2005 is different from March 2005)

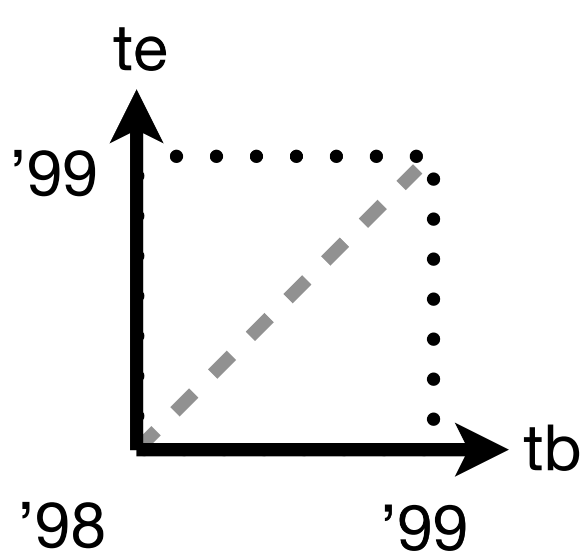
- **Temporal expressions are vague**, i.e., the precise time interval they refer to is uncertain and this uncertainty needs to be reflected
  - in the 1990s can refer to [1992, 1995], [1990, 1999], [1992, 1993], etc.
  - in 2002 can refer to [2002/01/01, 2002/12/31], [2002/05/04, 2002/07/02], etc.

Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)
- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- Example: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)

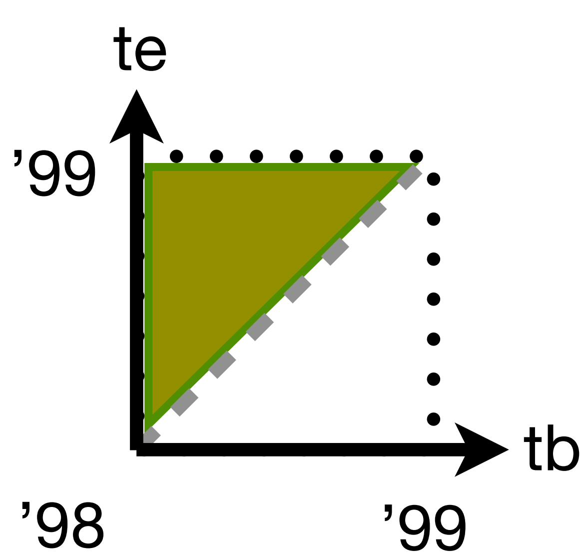


Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)
- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- Example: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)



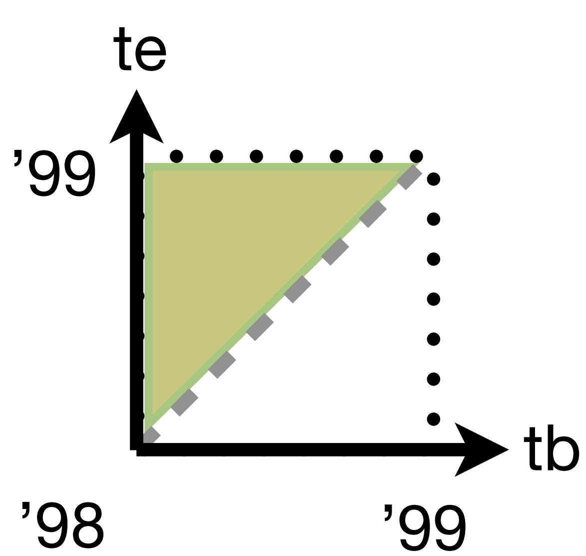
Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)

- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- Example: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)



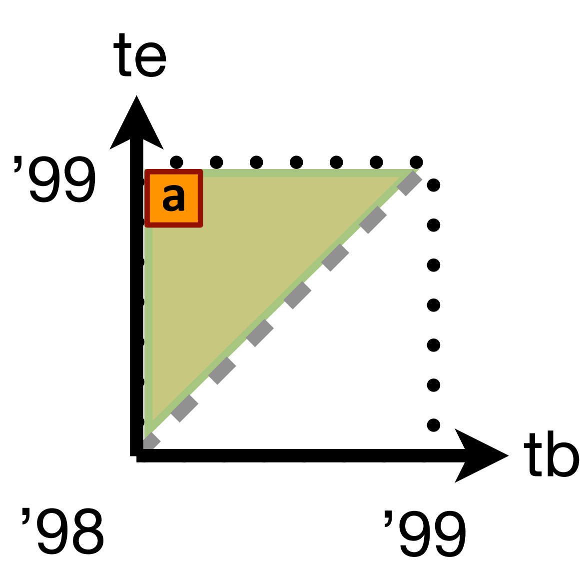
Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)

- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- <u>Example</u>: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)



(a) [1998/01/01, 1998/12/31]

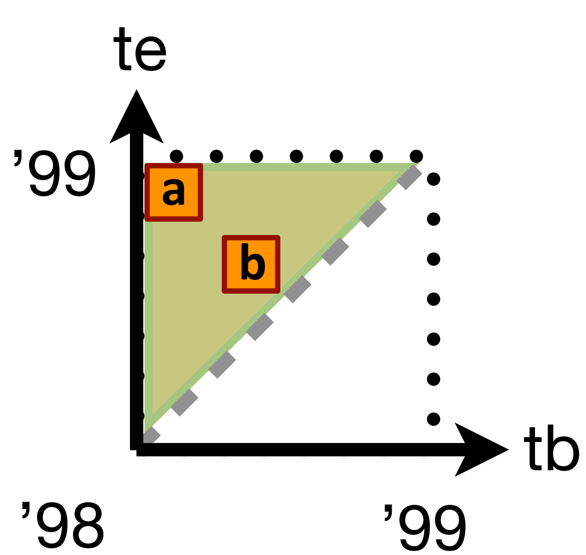
Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)

- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- Example: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)



(a) [1998/01/01, 1998/12/31]

(b) [1998/07/12, 1998/07/12]

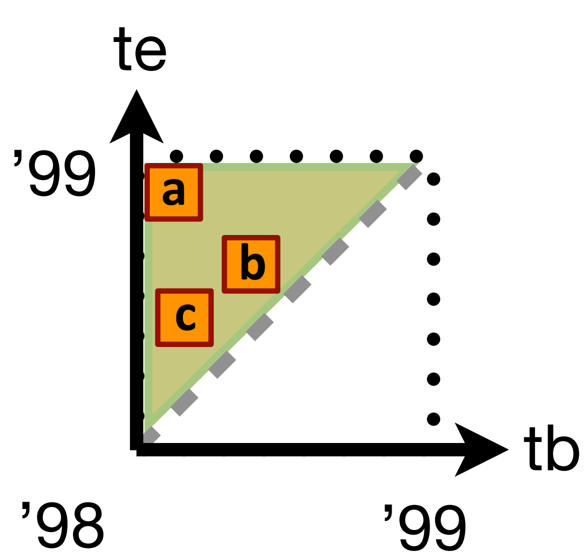
Temporal Expression Model

- Temporal expressions are **modeled as sets of time intervals** and denoted as four-tuples (tb_l, tb_u, te_l, te_u)

- Temporal expression T = (tb_l, tb_u, te_l, te_u) **can refer to any time interval** [tb, te] such that the following holds

$$tb_l \leq tb \leq tb_u \quad \wedge \quad tb \leq te \quad \wedge \quad te_l \leq te \leq te_u$$

- <u>Example</u>: Temporal expression in 1998 represented as (1998/01/01, 1998/12/31, 1998/01/01, 1998/12/31)



(a) [1998/01/01, 1998/12/31]

(b) [1998/07/12, 1998/07/12]

(c) [1998/02/07, 1998/02/22]

Document and Query Models

- Documents are modeled as a **set of textual terms** d_{text} and a **set of temporal expressions** d_{time}
- Queries are modeled as a **set of textual terms** q_{text} and a **set of temporal expressions** q_{time}
- Query-likelihood approach **assuming independence** between textual terms and temporal expressions

$$\mathrm{P}\left[\, q \,|\, d \,\right] = \mathrm{P}\left[\, q_{text} \,|\, d_{text} \,\right] \times \mathrm{P}\left[\, q_{time} \,|\, d_{time} \,\right]$$

- Query likelihood of **textual part** $P[q_{text} \,|\, d_{text}]$ is estimated using unigram language model with Jelinek-Mercer smoothing (mixing parameter: γ)

Language Model for Temporal Expressions

- Query likelihood of **temporal part** $P[q_{time} \mid d_{time}]$ is estimated
  - assuming independence between temporal expressions

$$P\left[\, q_{time} \mid d_{time} \,\right] = \prod_{Q \in q_{time}} P\left[\, Q \mid d_{time} \,\right]$$

  - assuming uniform probability for temporal expressions from document **d**

$$P\left[\, Q \mid d_{time} \,\right] = \frac{1}{|d_{time}|} \sum_{T \in d_{time}} P\left[\, Q \mid T \,\right]$$

  - assuming uniform probability for time intervals that **Q** can refer to

$$P\left[\, Q \mid T \,\right] = \frac{1}{|Q|} \sum_{[q_b,\, q_e] \in Q} P\left[\, [q_b,\, q_e] \mid T \,\right]$$

Language Model for Temporal Expressions

- assuming uniform probability for time intervals that T can refer to

$$P\left[\,[q_b,\, q_e] \mid T\,\right] = \frac{1}{|T|}\, \mathbb{1}([q_b,\, q_e] \in T)$$

- P[Q | T] can be simplified as

$$P\left[\,Q \mid T\,\right] = \frac{|T \cap Q|}{|T| \cdot |Q|}$$

treating temporal expressions as sets of time intervals

- P[Q | d_{time}] is smoothed with collection model P[Q | D_{time}] using Jelinek-Mercer smoothing (mixing parameter: λ)

Experimental Evaluation

- Document Collection: The New York Times Annotated Corpus (1.8 million newspaper articles published between ’87 and ’07)
- Queries: 40 queries in total gathered using **crowdsourcing**
  - related to **four topics** *sports*, *culture*, *technology*, *world affairs*
  - **five temporal granularities** (day, month, year, decade, century)

Queries

	Sports	**Culture**
Day	boston red sox [october 27, 2004] ac milan [may 23, 2007]	kurt cobain [april 5, 1994] keith harring [february 16, 1990]
Month	stefan edberg [july 1990] italian national soccer team [july 2006]	woodstock [august 1994] pink floyd [march 1973]
Year	babe ruth [1921] chicago bulls [1991]	rocky horror picture show [1975] michael jackson [1982]
Decade	michael jordan [1990s] new york yankees [1910s]	sound of music [1960s] mickey mouse [1930s]
Century	la lakers [21st century] soccer [21st century]	academy award [21st century] jazz music [21st century]
	Technology	**World Affairs**
Day	mac os x [march 24, 2001] voyager [september 5, 1977]	berlin [october 27, 1961] george bush [january 18, 2001]
Month	thomas edison [december 1891] microsoft halo [june 2000]	poland [december 1970] pearl harbor [december 1941]
Year	roentgen [1895] wright brothers [1905]	nixon [1970s] iraq [2001]
Decade	internet [1990s] sewing machine [1850s]	vietnam [1960s] monica lewinsky [1990s]
Century	musket [16th century] siemens [19th century]	queen victoria [19th century] muhammed [7th century]

Precision / nDCG

	P@5	N@5	P@10	N@10
LM ($\gamma = 0.25$)	0.33	0.34	0.30	0.32
LM ($\gamma = 0.75$)	0.38	0.39	0.37	0.38
LMTU-EX ($\gamma = 0.25,\ \lambda = 0.75$)	0.53	0.51	0.49	0.49
LMTU-EX ($\gamma = 0.5,\ \lambda = 0.75$)	**0.54**	**0.52**	**0.51**	**0.49**

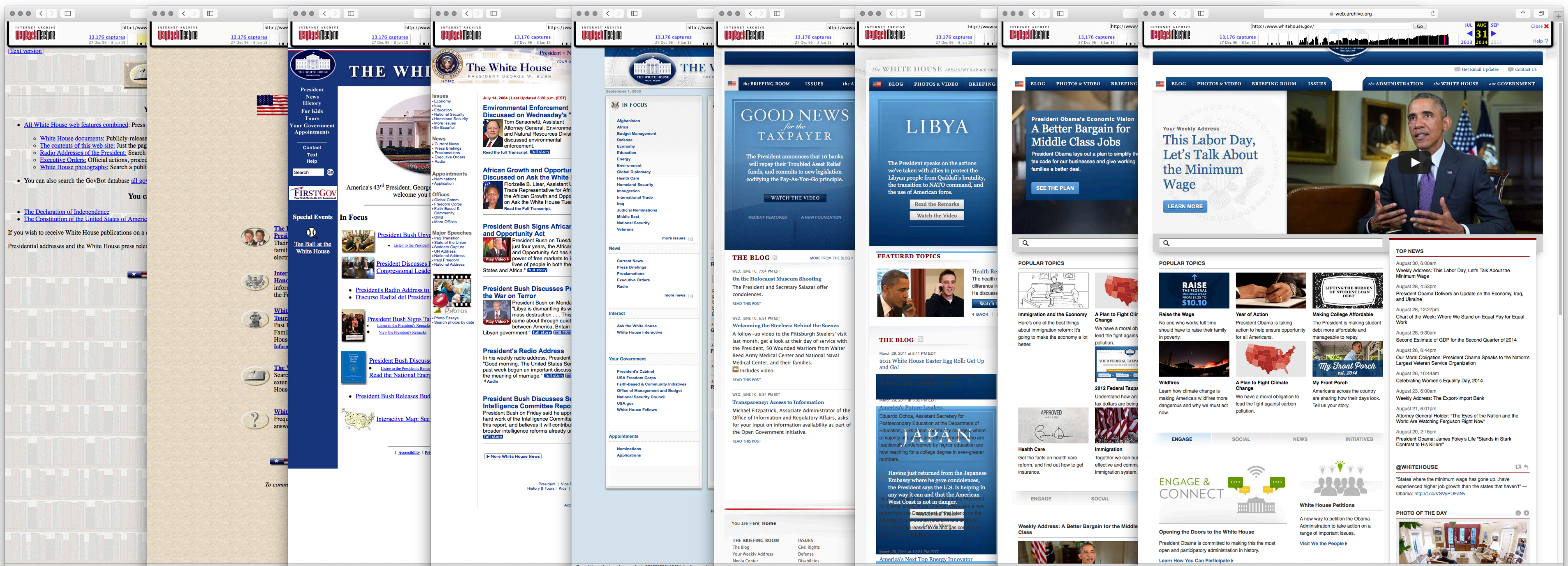
7.3. Search in Web Archives

- Web archives (e.g., archive.org, internetmemory.org) **preserve old snapshots** of URLs (web pages, images, etc.)

- Internet Archive has harvested **435 billion web pages** (including embedded media files) **since 1996**

7.3. Search in Web Archives

- Web archives (e.g., archive.org, internetmemory.org) **preserve old snapshots** of URLs (web pages, images, etc.)



- Internet Archive has harvested **435 billion web pages** (including embedded media files) **since 1996**

Search in Web Archives

- Challenges & Opportunities:
  - **vast volume** of web archives (Internet Archive: 435 billion snapshots)
  - **longitudinal coverage** of web archives (Internet Archive: 1996 – now)
  - document versions (snapshots of the same document) taken at nearby times exhibit a **high degree of redundancy** allowing for **compression**
  - document versions come with a **valid-time interval**, indicating when the version was current, which allows for **more effective search**

7.3.1. Non-Redundant Indexing

- Zhang and Suel [11] devise an approach to index **highly-redundant document collections** (e.g., web archives)

- <u>Ideas</u>:

  - **break up documents** into shorter segments

  - **segments should be shared** between overlapping documents

  - use a **two-level index structure** to index associations between **words-and-segments** and **segments-and-documents**

Segmenting Documents

- Problem: How to break up documents into smaller segments so that segments are **shared between overlapping** documents

d_1

a	a	c
b	a	b
c	c	b

d_2

a	c	b
a	b	c
c	b	a

- **Hash breaking** (as a naïve approach)
  - compute **hash code** h[i] for each term **d[i]** in document
  - break document **at all indices** i such that h[i] % w = 0

Segmenting Documents

- Problem: How to break up documents into smaller segments so that segments are **shared between overlapping documents**

d_1

a	a	c
b	a	b
c	c	b

d_2

a	c	b
a	b	c
c	b	a

- **Hash breaking** (as a naïve approach)
  - compute **hash code** h[i] for each term **d[i]** in document
  - break document **at all indices** i such that h[i] % w = 0

Segmenting Documents

- Problem: How to break up documents into smaller segments so that segments are **shared between overlapping** documents

- **Hash breaking** (as a naïve approach)
  - compute **hash code** h[i] for each term **d[i]** in document
  - break document **at all indices** i such that h[i] % w = 0

Segmenting Documents

- Problem: How to break up documents into smaller segments so that segments are **shared between overlapping** documents

d$_1$

a	a	c
b	a	b
c	c	b

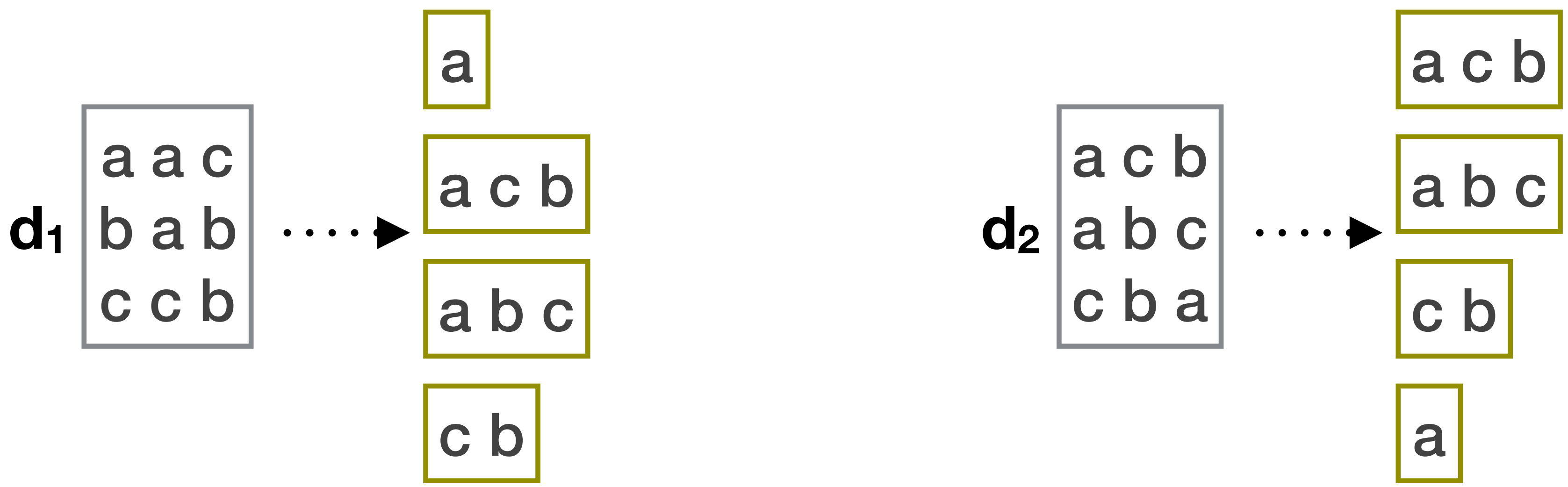
d$_2$

a	c	b
a	b	c
c	b	a

- **Hash breaking** (as a naïve approach)
  - compute **hash code** h[i] for each term **d[i]** in document
  - break document **at all indices** i such that h[i] % w = 0

Segmenting Documents

- Problem: How to break up documents into smaller segments so that segments are **shared between overlapping** documents

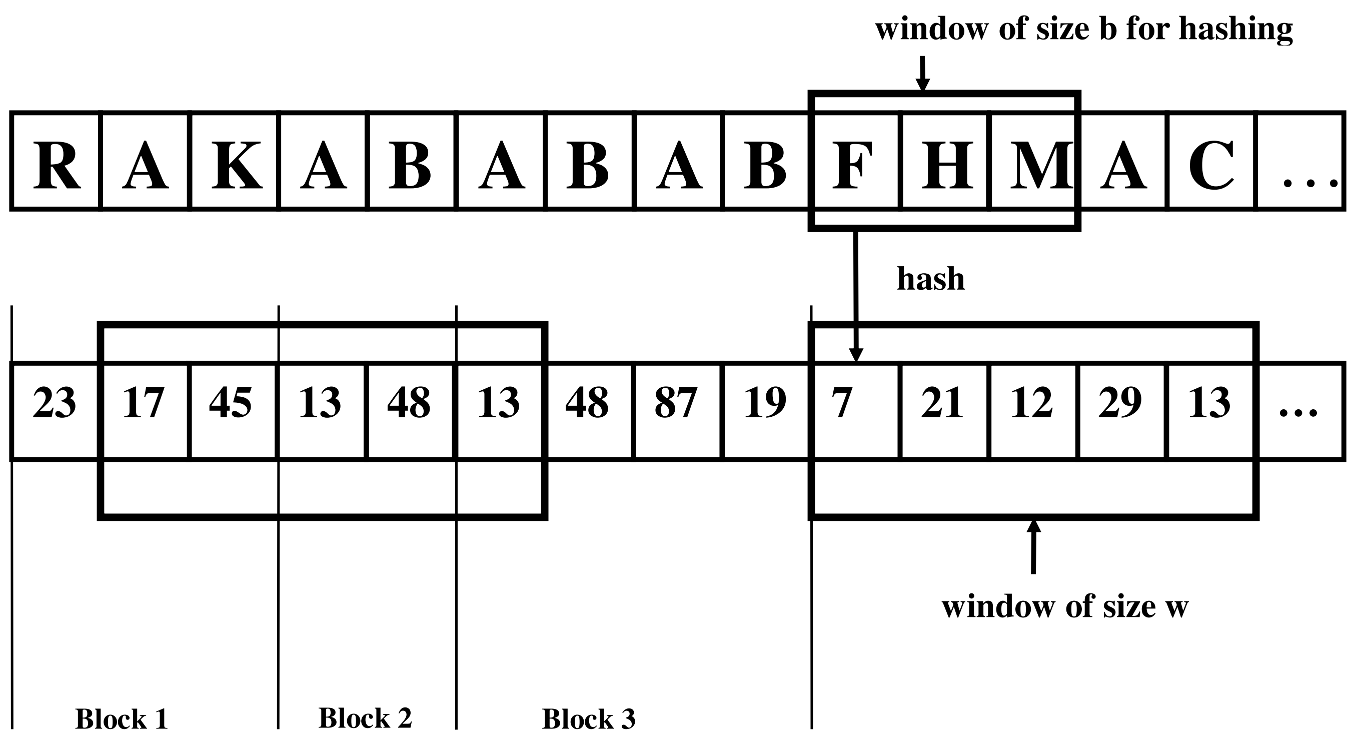


- **Hash breaking** (as a naïve approach)
  - compute **hash code** h[i] for each term **d[i]** in document
  - break document **at all indices** i such that **h[i]** % w = 0

Winnowing

- **Winnowing** [10] (as a better approach with guarantees)
  - compute **hash code** h[i] for all subsequences d[i … i+b-1] of length b
  - **slide window** of size **w** over the array of hash codes h[0 .. |d|-b]
    - if h[i] is **strictly smaller** than all other values h[j] in current window
      then **cut the document** between i and i -1
    - if there are **multiple positions** i in the current window with minimal value h[i]
      - if we have previously cut directly before one of them, then don't perform a cut
      - otherwise, cut before the rightmost position i having minimal value h[i]

Winnowing



- Winnowing **guarantees** that two documents having a subsequence of length at least w+b+1 in common share at least one segment
- **Maximum segment length** is w
- **Expected sequence length** is (w+1)/2

Query Processing

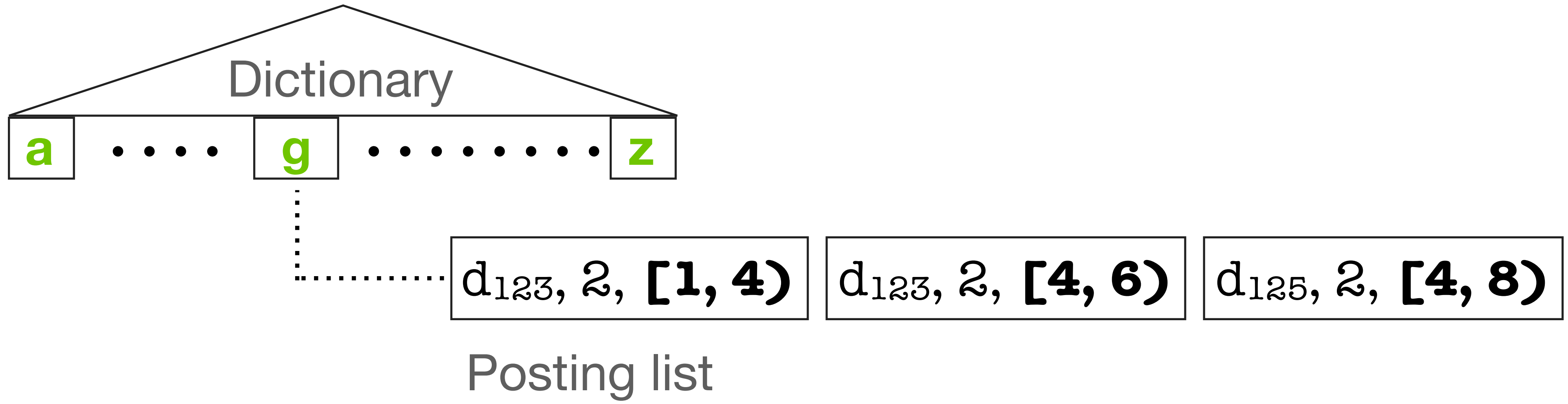
- Query processing **needs to be adapted** to reflect that the same segment can occur in many documents
  - when seeing a segment in a posting list of the first index, **look up documents containing it** in the second index
  - **effectiveness of skipping** for conjunctive queries is **reduced**
    - terms could be spread over different segments in a document
    - segments can be contained in documents with arbitrary document identifiers

7.3.2. Time-Travel Text Search

- Berberich et al. [2] develop an approach to support time-travel text search on **version document collections**
- **Time-travel keyword query** q@t combines keywords **q** with a time of interest **t** to search "as of" the indicated time in the past
- Ideas:
  - **coalesce postings** belonging to temporally adjacent versions if their payloads (e.g., score) are almost the same
  - **partition the index along time** to improve query processing performance and

Time-Travel Inverted Index

- Time-travel inverted index adds a **valid-time interval** $[t_b, t_e)$ to postings indicating when the information therein was current

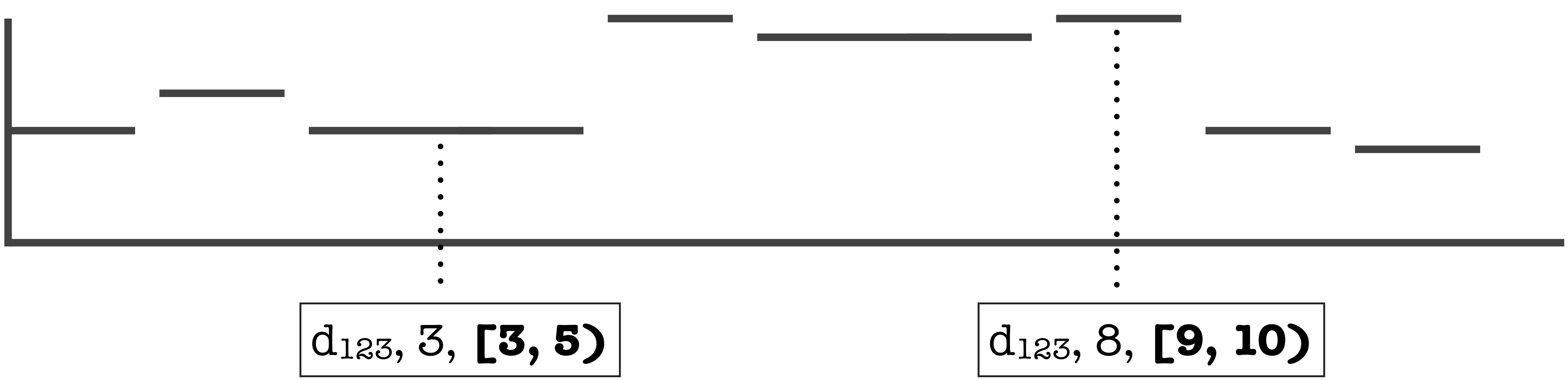


- Time-travel keyword query **q@t** is processed by **reading posting lists** for keywords in **q** and **filtering out postings** whose valid-time interval does not contain **t**, i.e.:

$$t \notin [t_b, t_e)$$

Temporal Coalescing

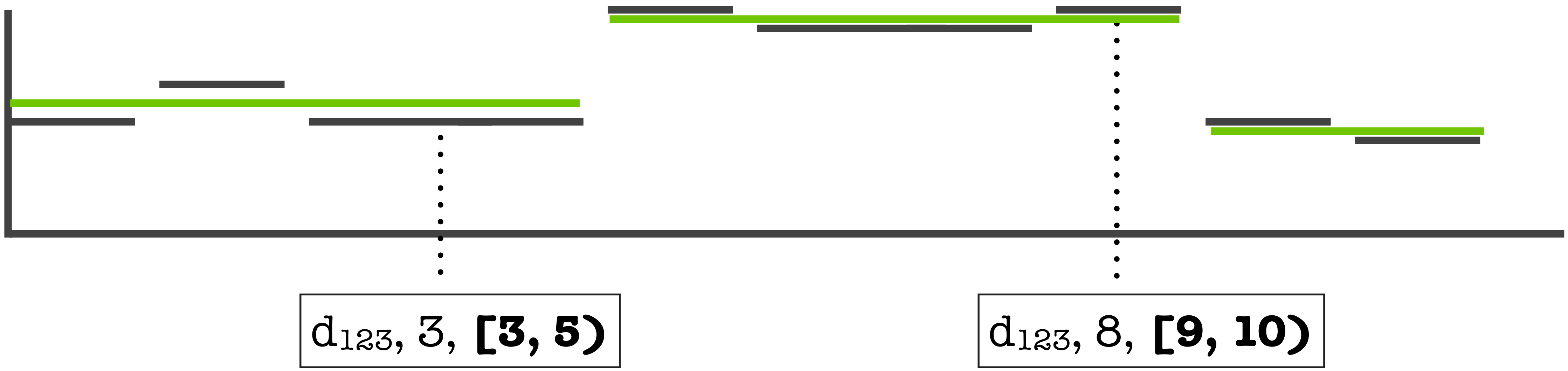
- Naïve application of time-travel inverted index results in **one posting per keyword per document version**
- <u>Observation</u>: Postings belonging to **temporally adjacent versions of the same document** often have similar payloads



- <u>Idea</u>: Coalesce (i.e., group together) postings having similar payloads to **reduce index size**

Temporal Coalescing

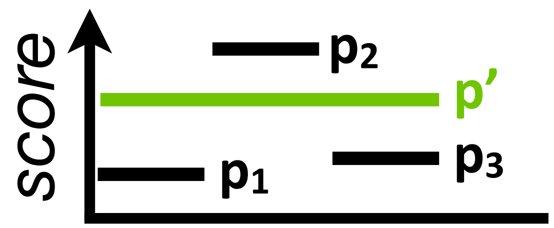
- Naïve application of time-travel inverted index results in **one posting per keyword per document version**
- <u>Observation</u>: Postings belonging to **temporally adjacent versions of the same document** often have similar payloads



- <u>Idea</u>: Coalesce (i.e., group together) postings having similar payloads to **reduce index size**

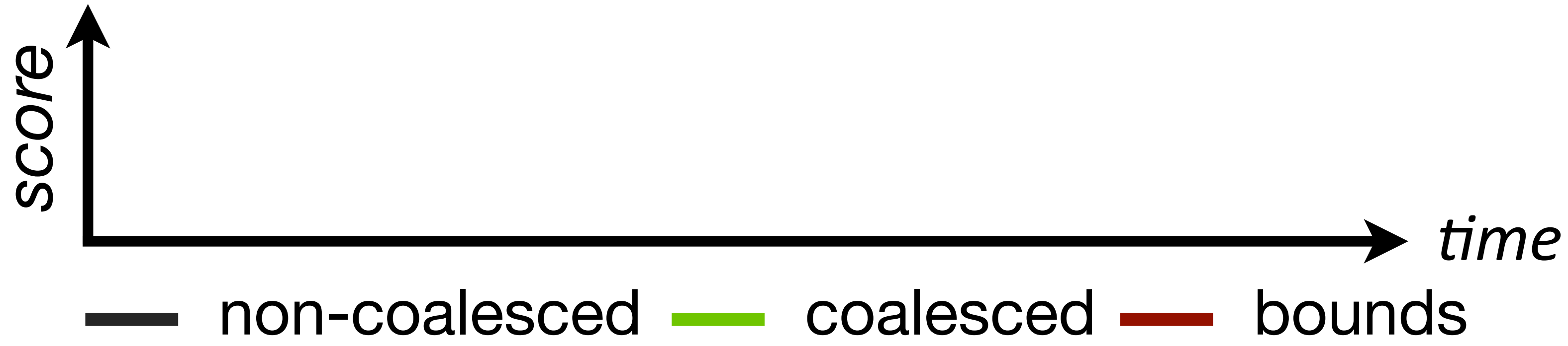
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



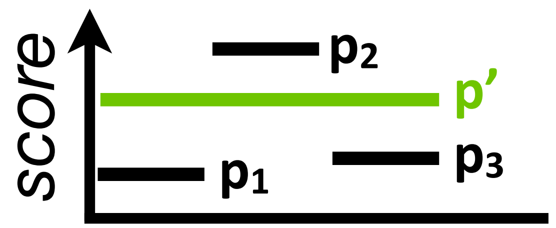
$$\forall p_i \in I \; : \; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



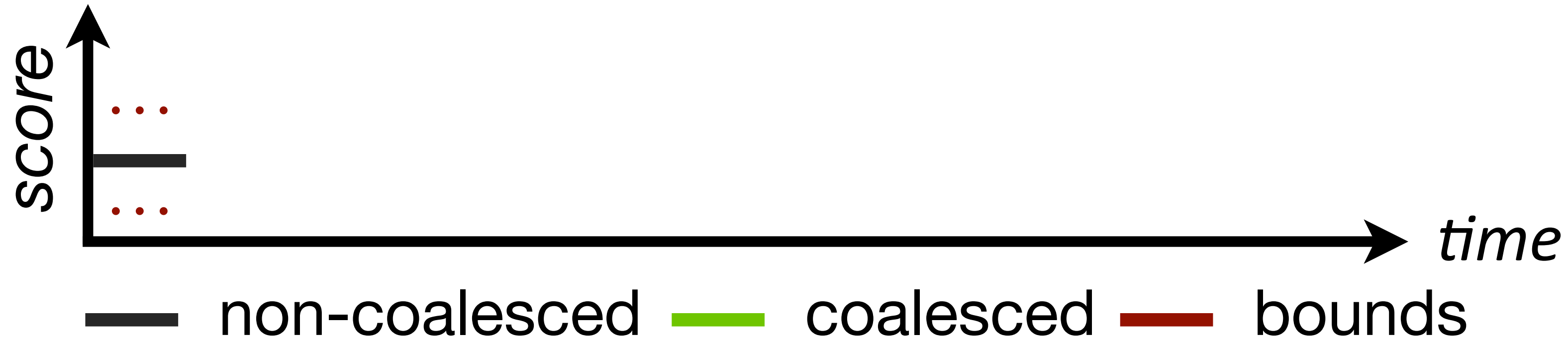
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



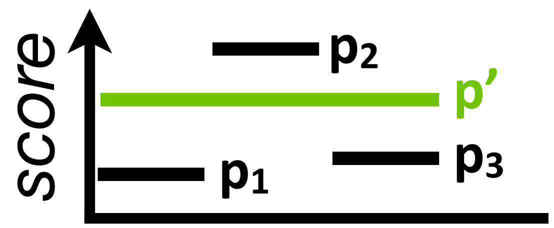
$$\forall p_i \in I \; : \; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



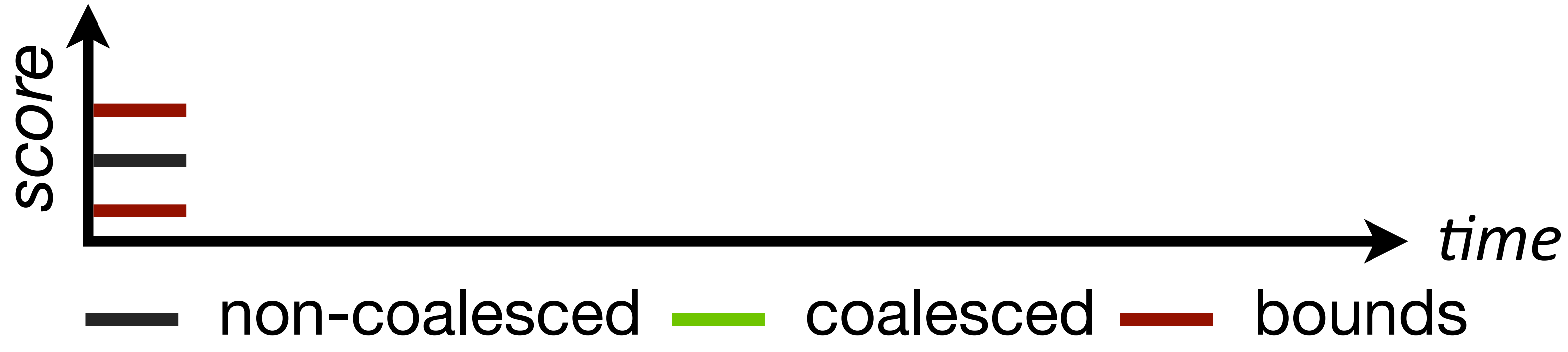
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



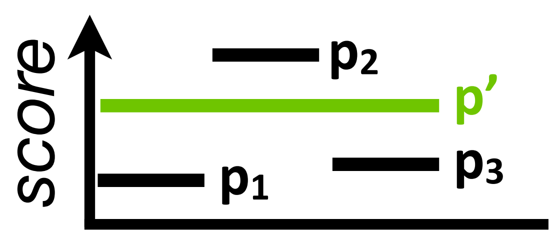
$$\forall p_i \in I \ : \ \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



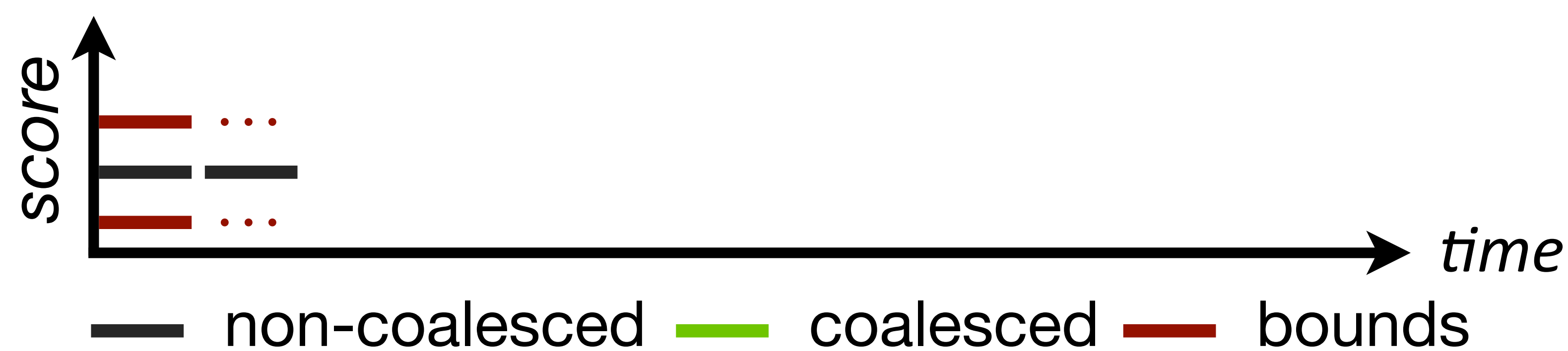
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



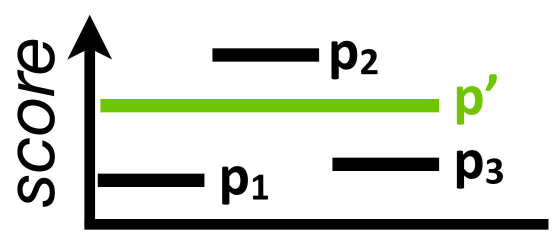
$$\forall p_i \in I \; : \; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



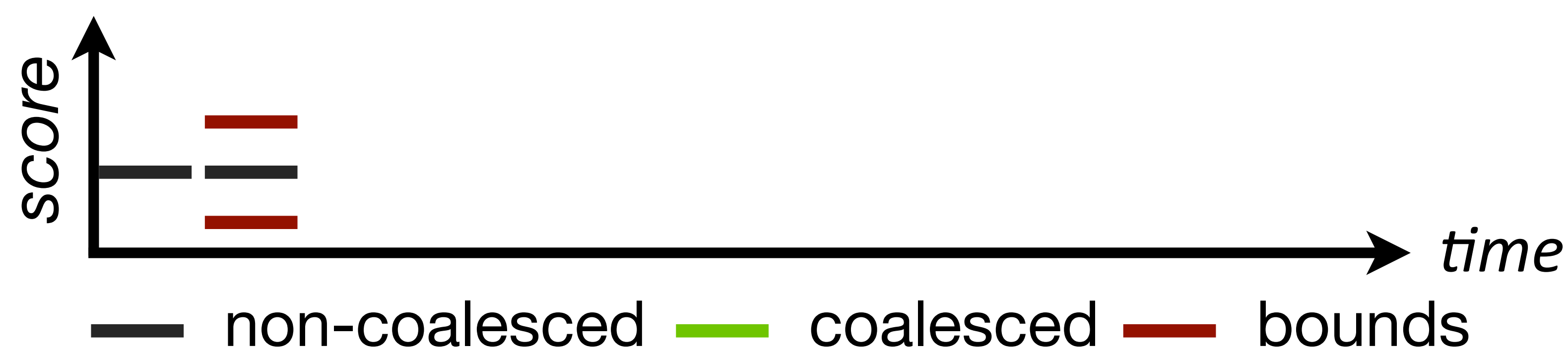
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



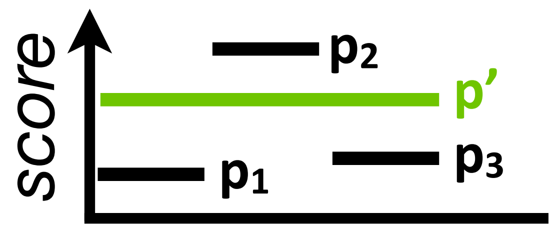
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



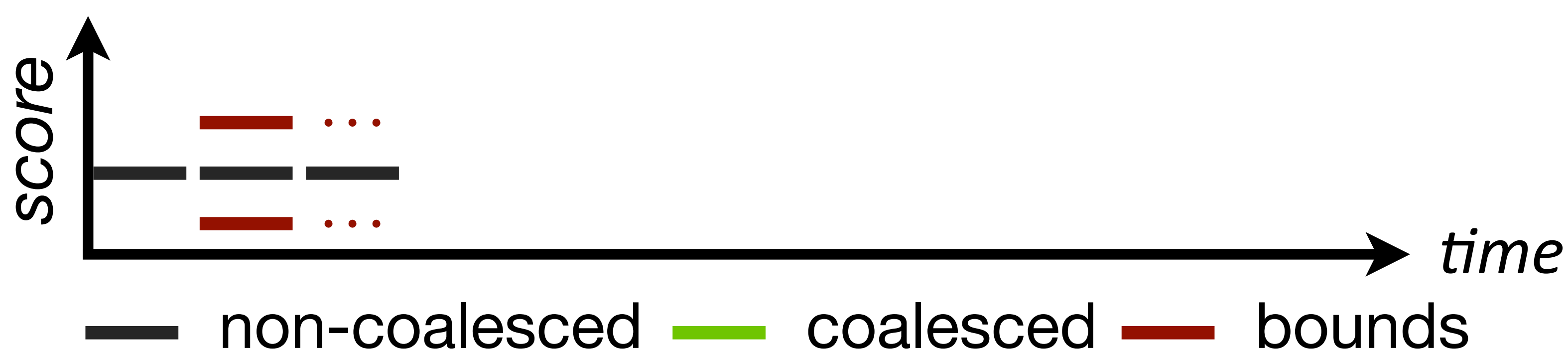
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



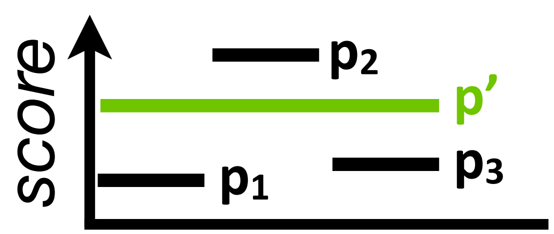
$$\forall p_i \in I \; : \; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



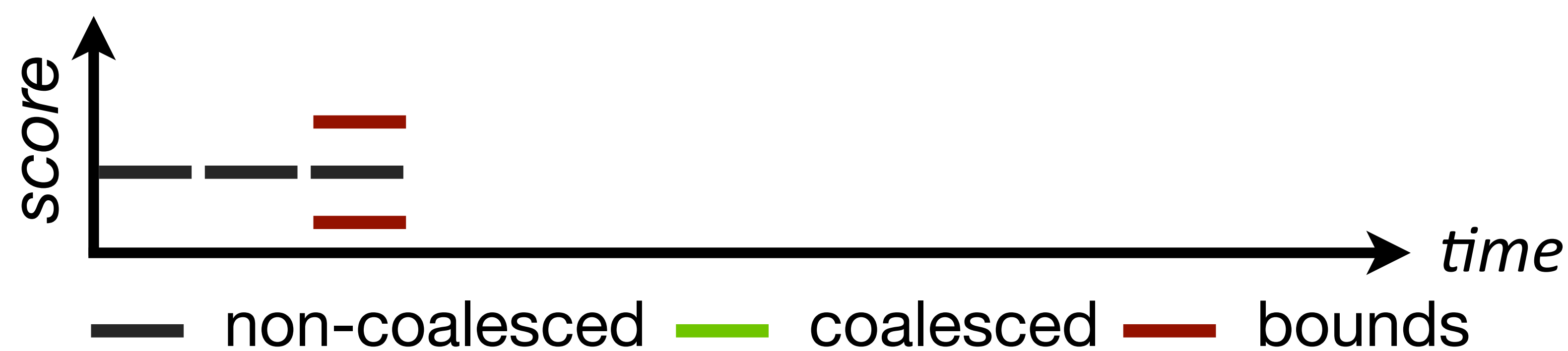
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



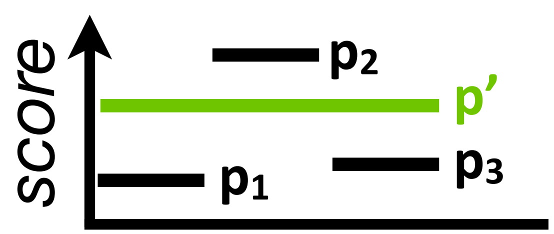
$$\forall p_i \in I \ : \ \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



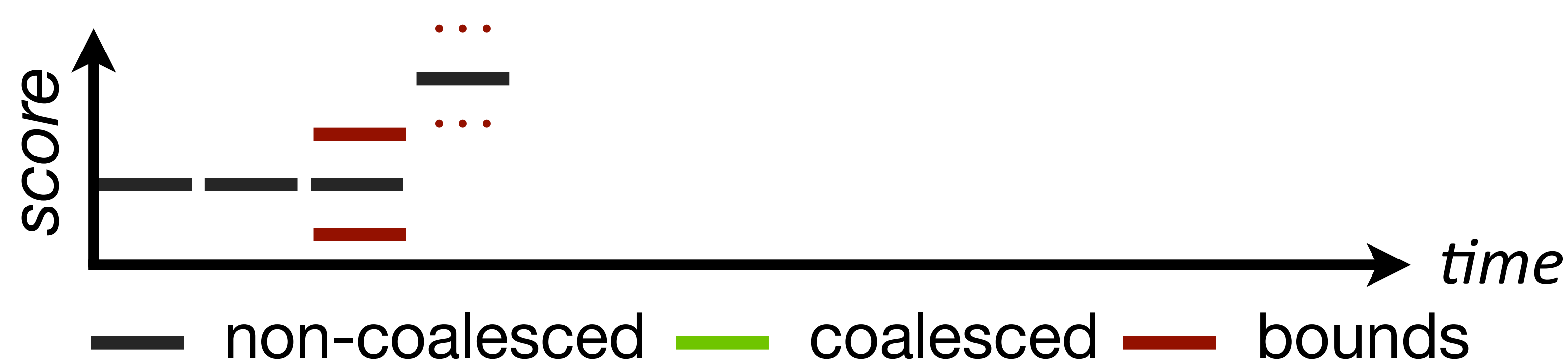
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



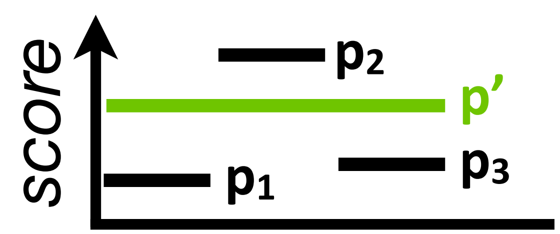
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



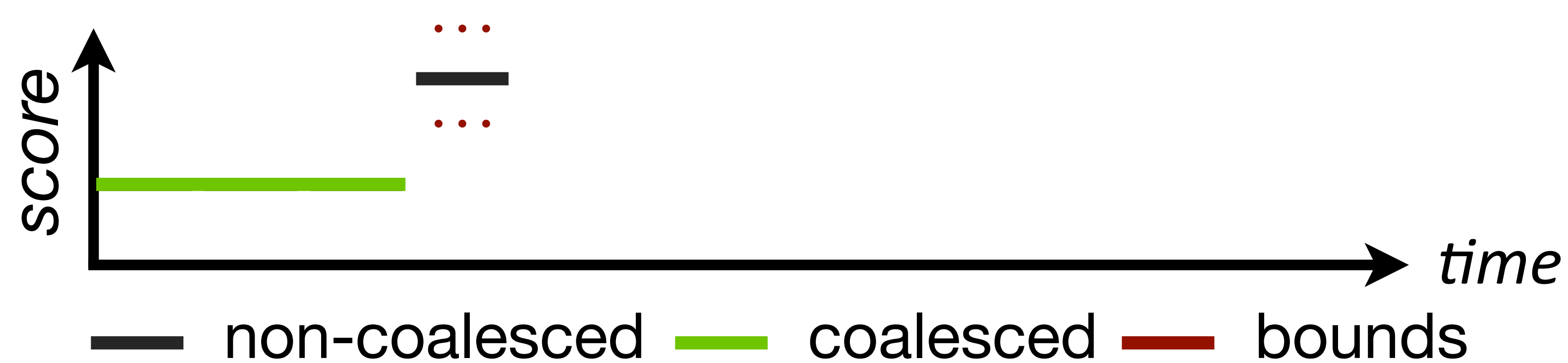
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



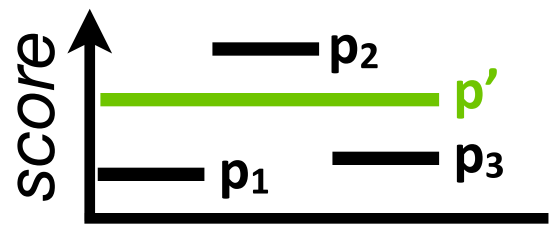
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



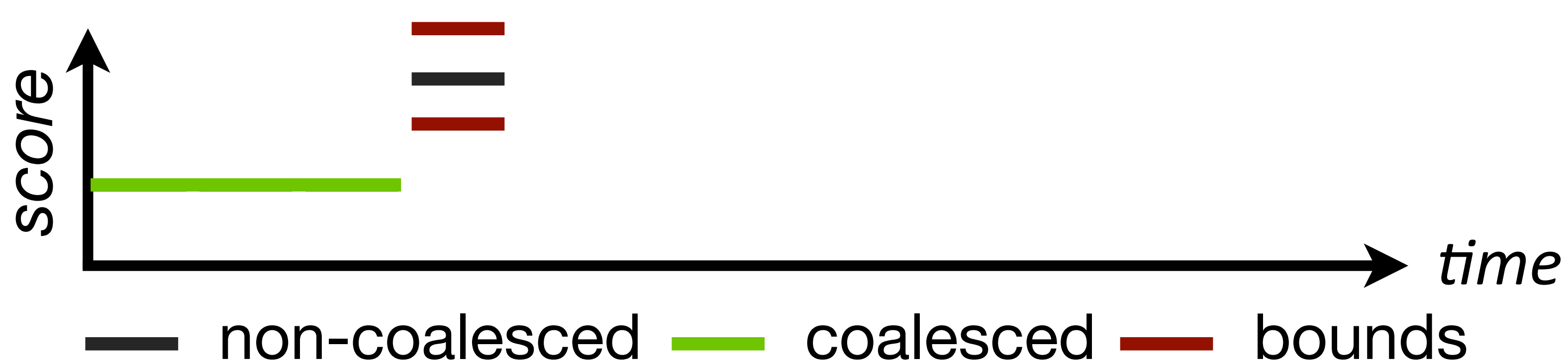
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



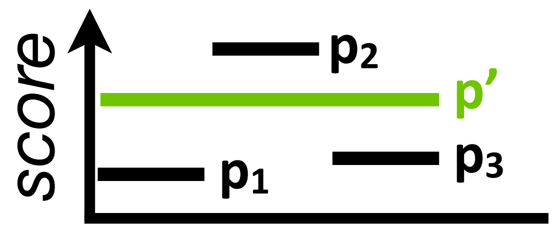
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



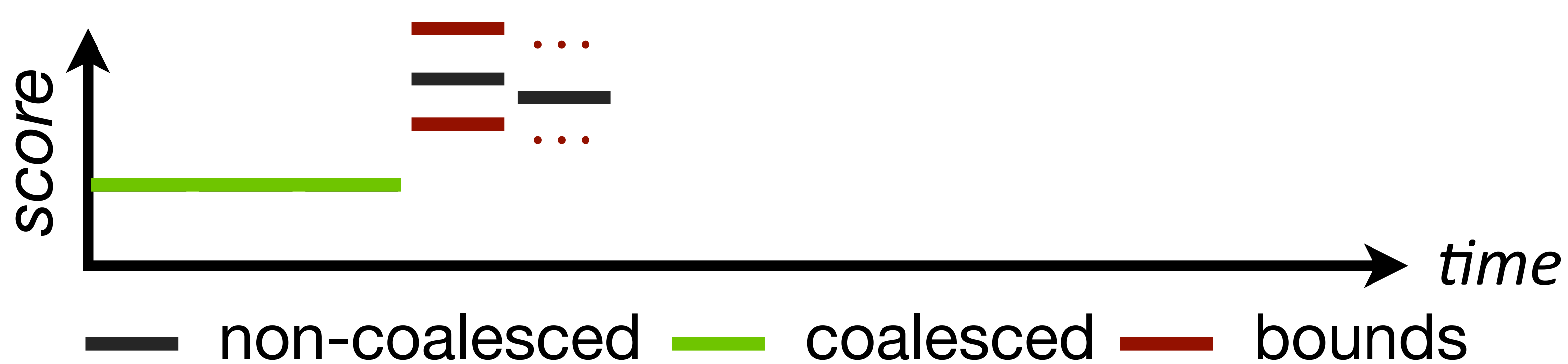
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



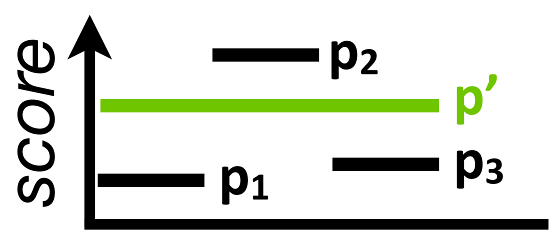
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



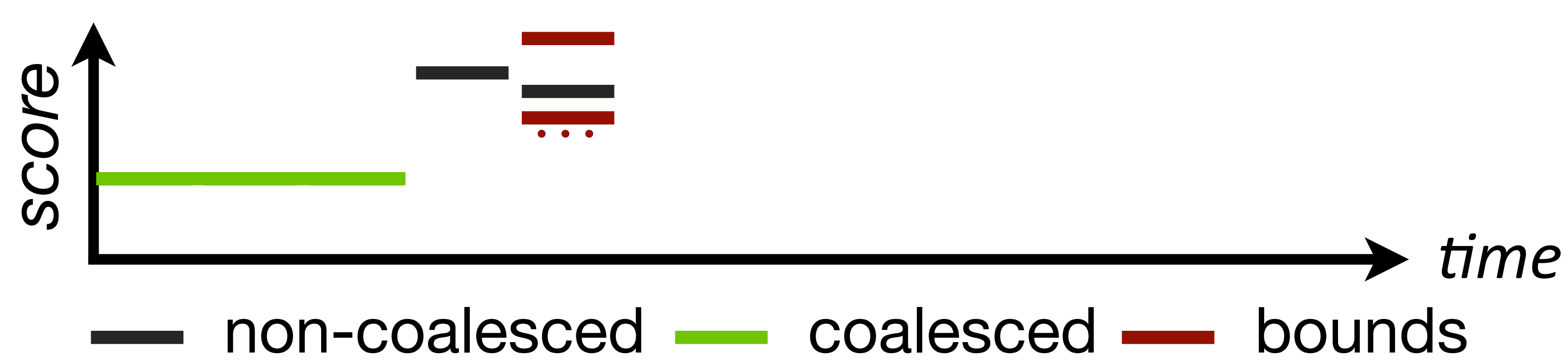
Temporal Coalescing

- <u>Problem Statement</u>: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



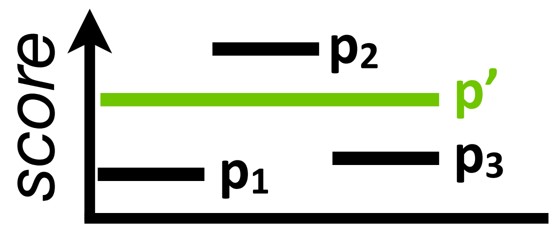
$$\forall\, p_i \in I \ : \ \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



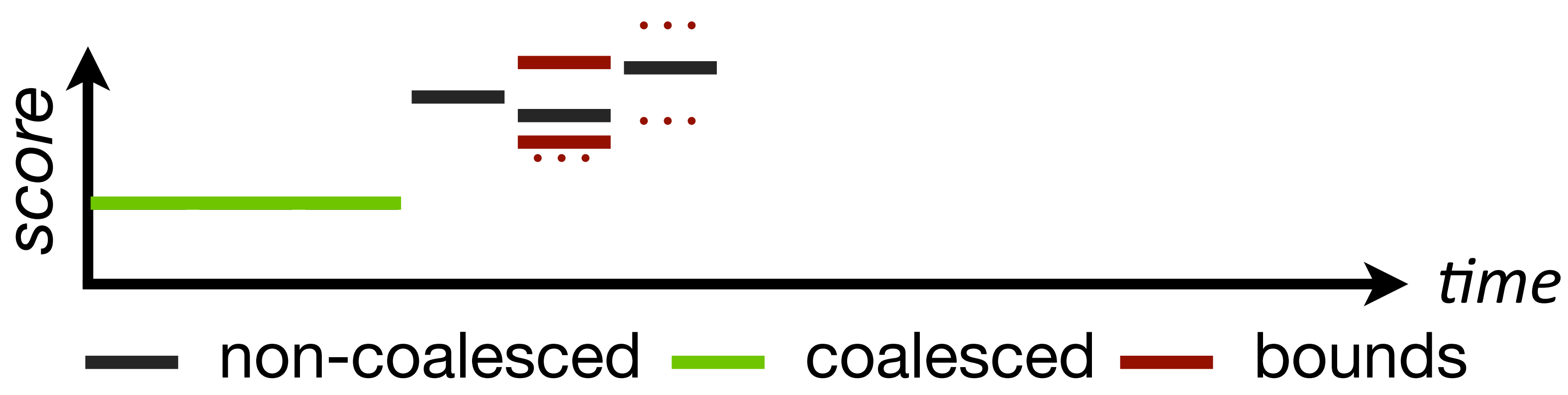
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



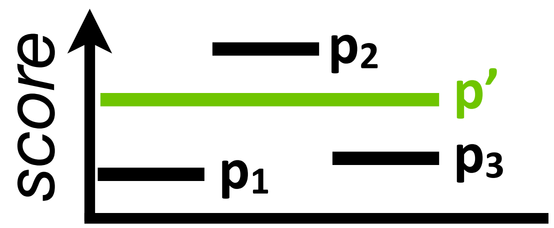
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



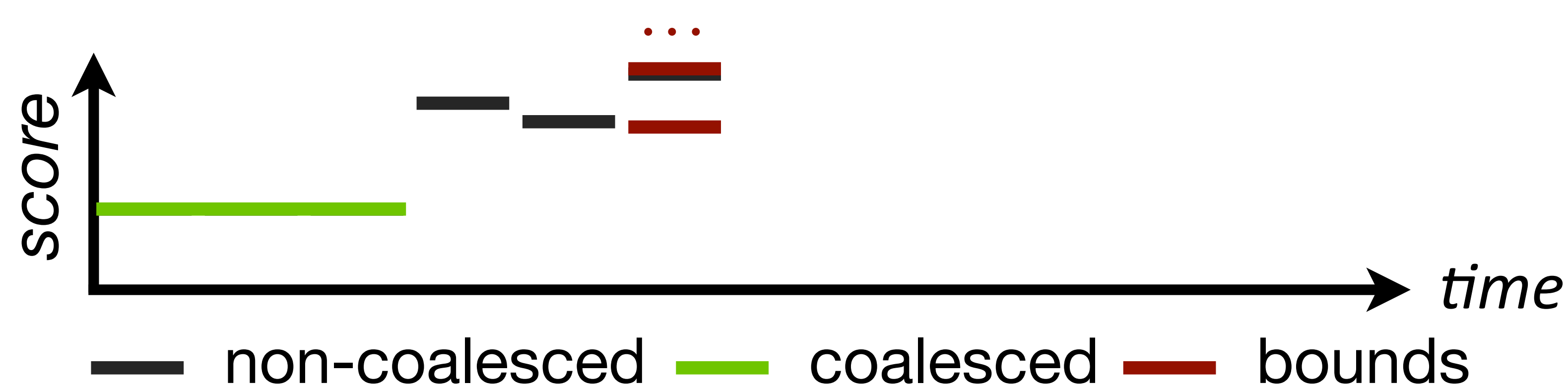
Temporal Coalescing

- <u>Problem Statement</u>: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



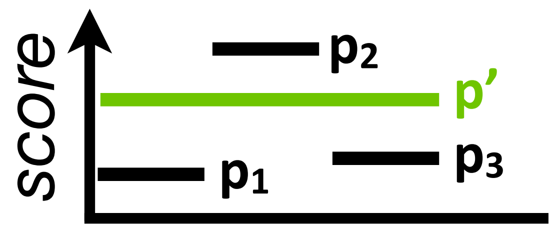
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



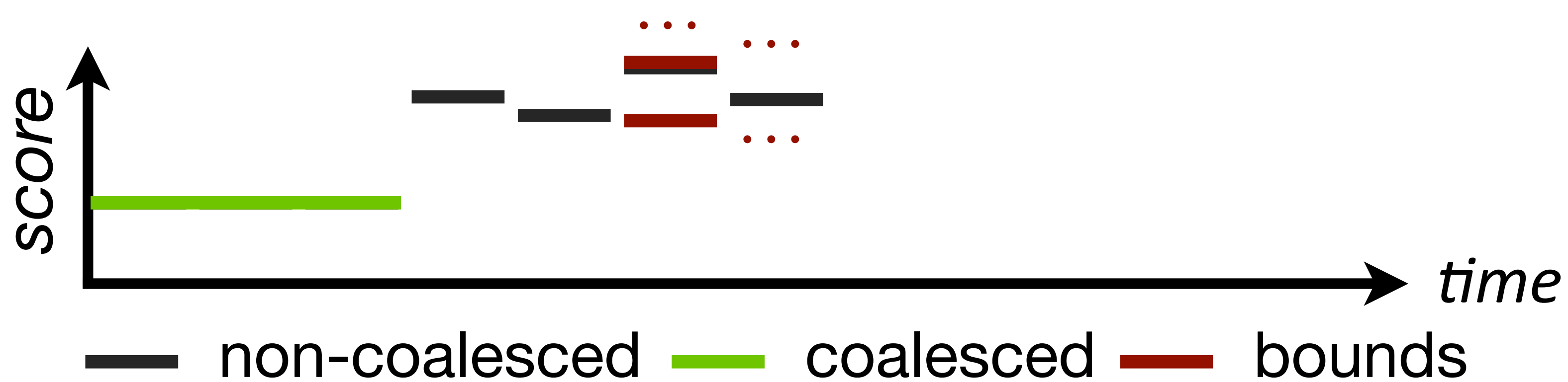
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



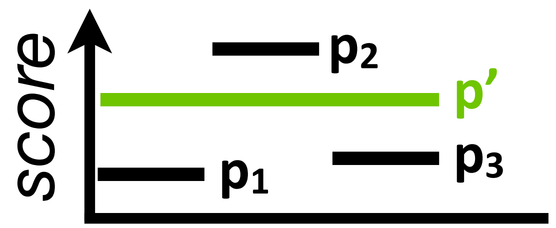
$$\forall p_i \in I \; : \; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



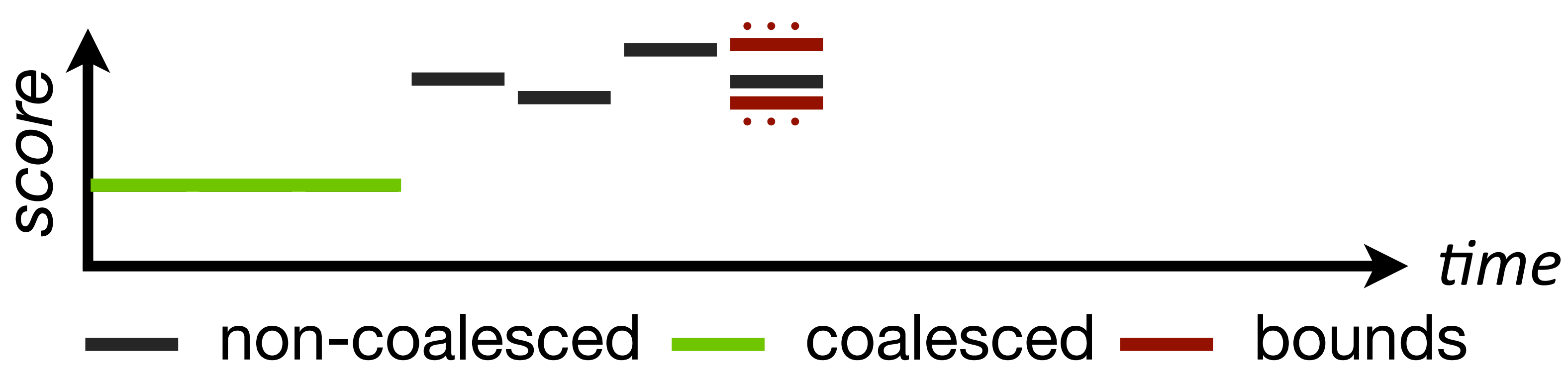
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



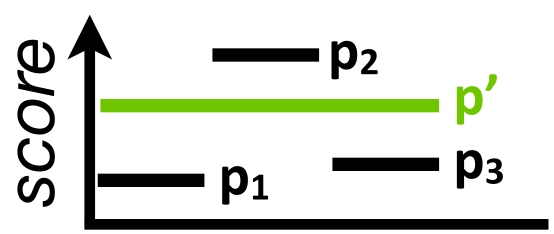
$$\forall p_i \in I \ : \ \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



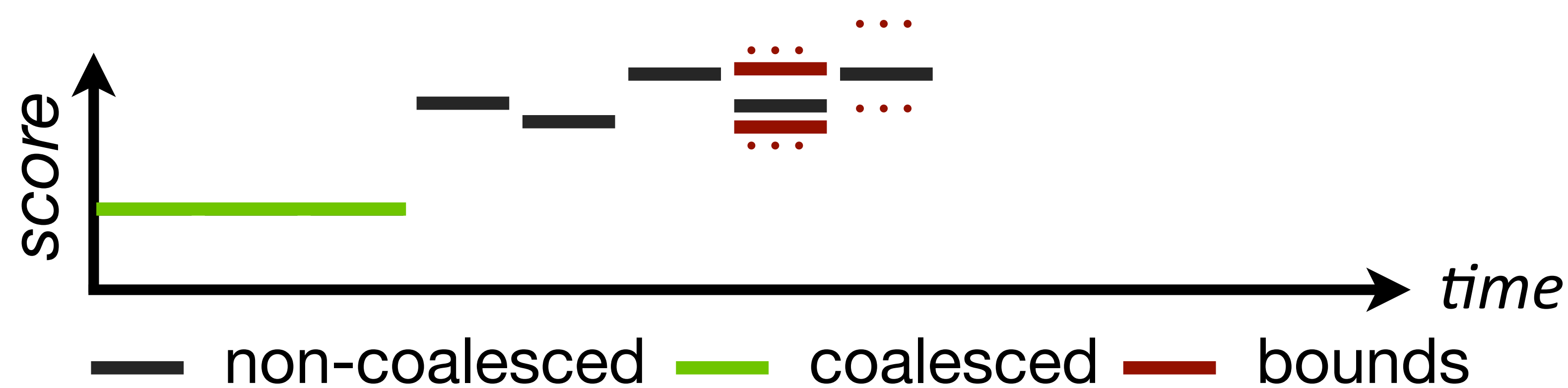
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



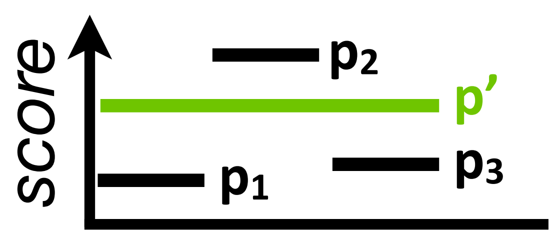
$$\forall p_i \in I \ : \ \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



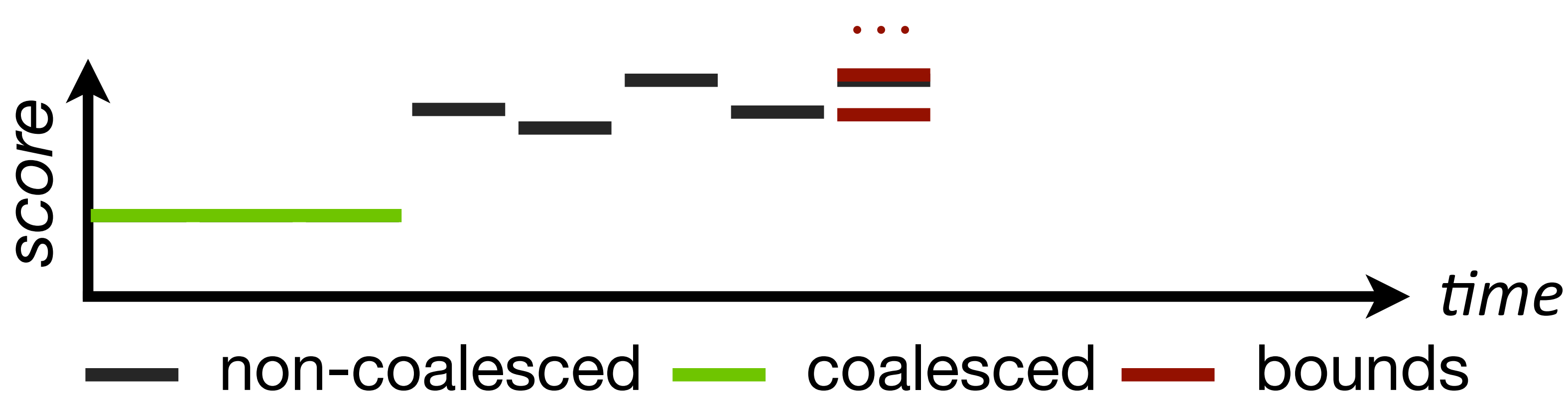
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



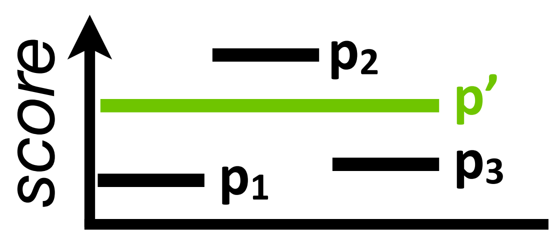
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \le \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



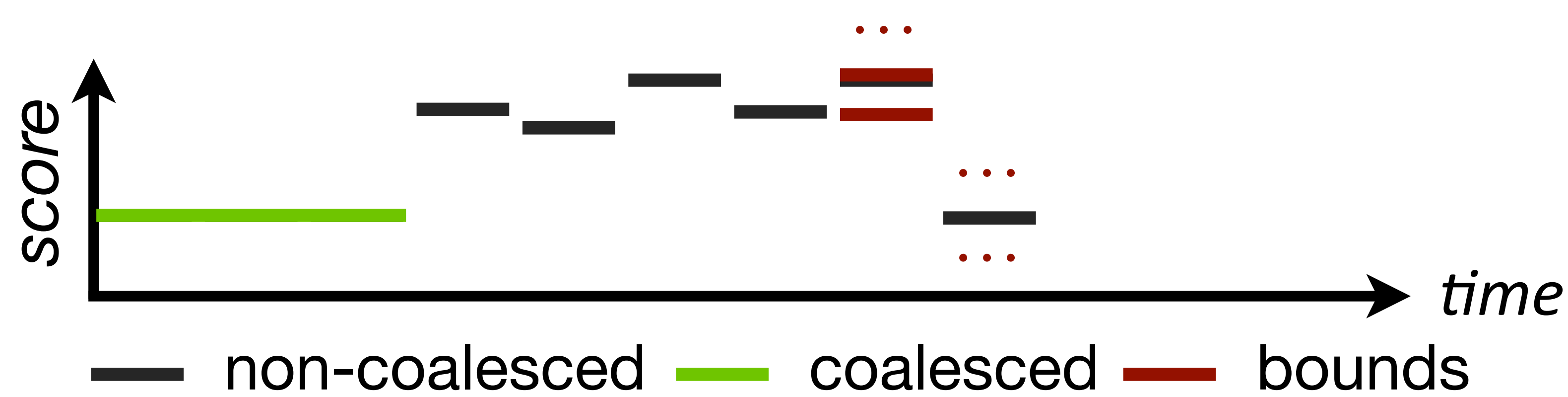
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



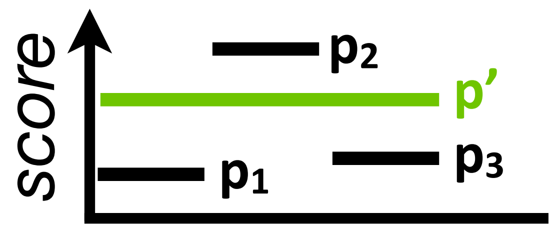
$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



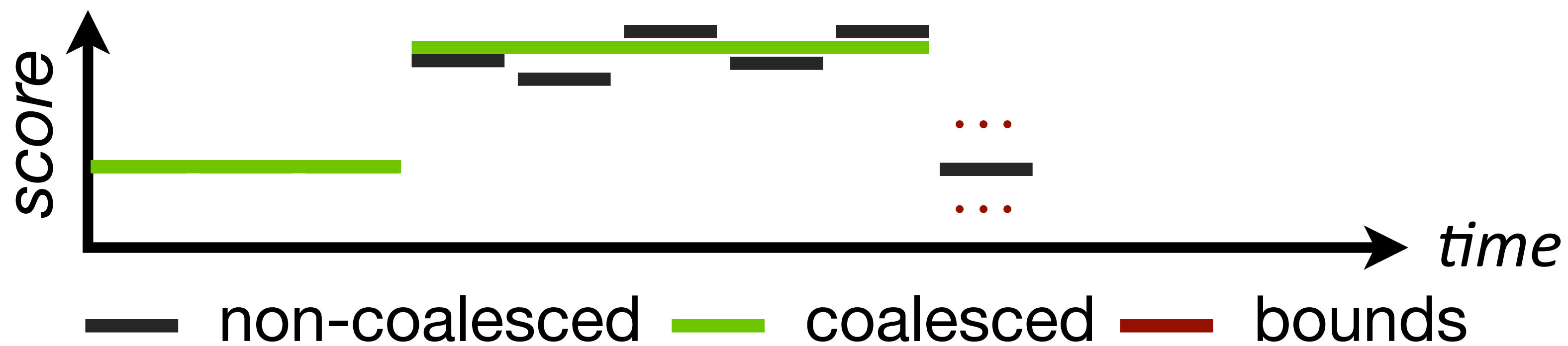
Temporal Coalescing

- Problem Statement: Given a sequence **I** of postings for term **v** in document **d**, determine a **minimal-length output sequence** O that keeps the **relative approximation error** below a threshold ε



$$\forall\, p_i \in I \;:\; \frac{|p_i - \hat{p}|}{p_i} \leq \epsilon$$

- Optimal output sequence can be determined using a **greedy one-pass algorithm** in time O(|I|)



Temporal Coalescing

Input: Sequence I of temporally adjacent postings $\langle p_1, \ldots, p_n \rangle$ for document **d** each with valid-time interval $[t_b, t_e)$, and score **s**

Output: Sequence O

$O = \langle\ \rangle$; $D = d$; $LOW = p_1.s - p_1.s \times \varepsilon$; $UP = p_1.s + p_1.s \times \varepsilon$; $TB = p_1.t_b$ // initialize

for each posting p_i from input sequence I

  $low = p_i.s - p_i.s \times \varepsilon$; $up = p_i.s + p_i.s \times \varepsilon$ // lower and upper bound

  if $[LOW, UP] \cap [low, up] \neq \emptyset$ // can p_i be coalesced?

    $LOW = \max(low, LOW)$, $UP = \min(up, UP)$

  else

    $TE = p_i.t_b$;$O = O \cup \{(D, [TB, TE), (LOW + UP) / 2)\}$ // coalesced posting

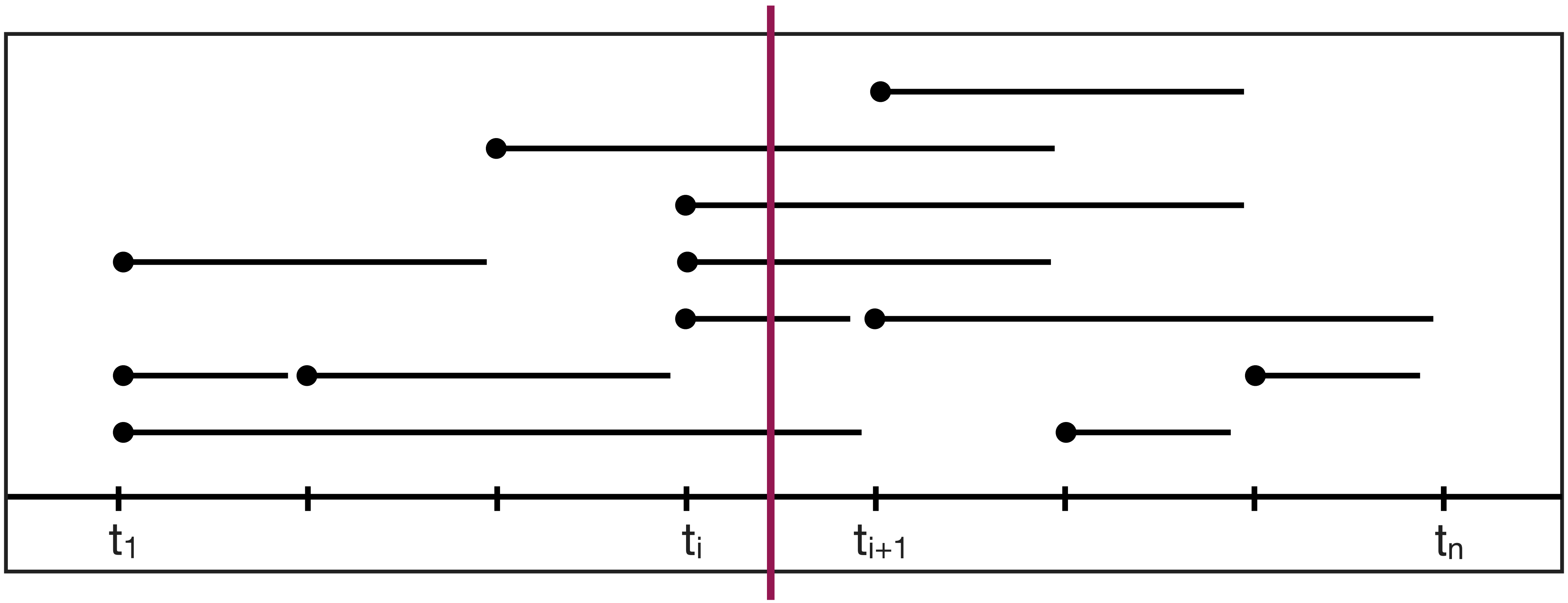
    $LOW = low$; $UP = up$; $TB = p_i.t_b$ // re-initialize

  if $i = n$

    $TE = p_i.t_e$; $O = O \cup \{(D, [TB, TE), (LOW + UP) / 2)\}$ // last posting

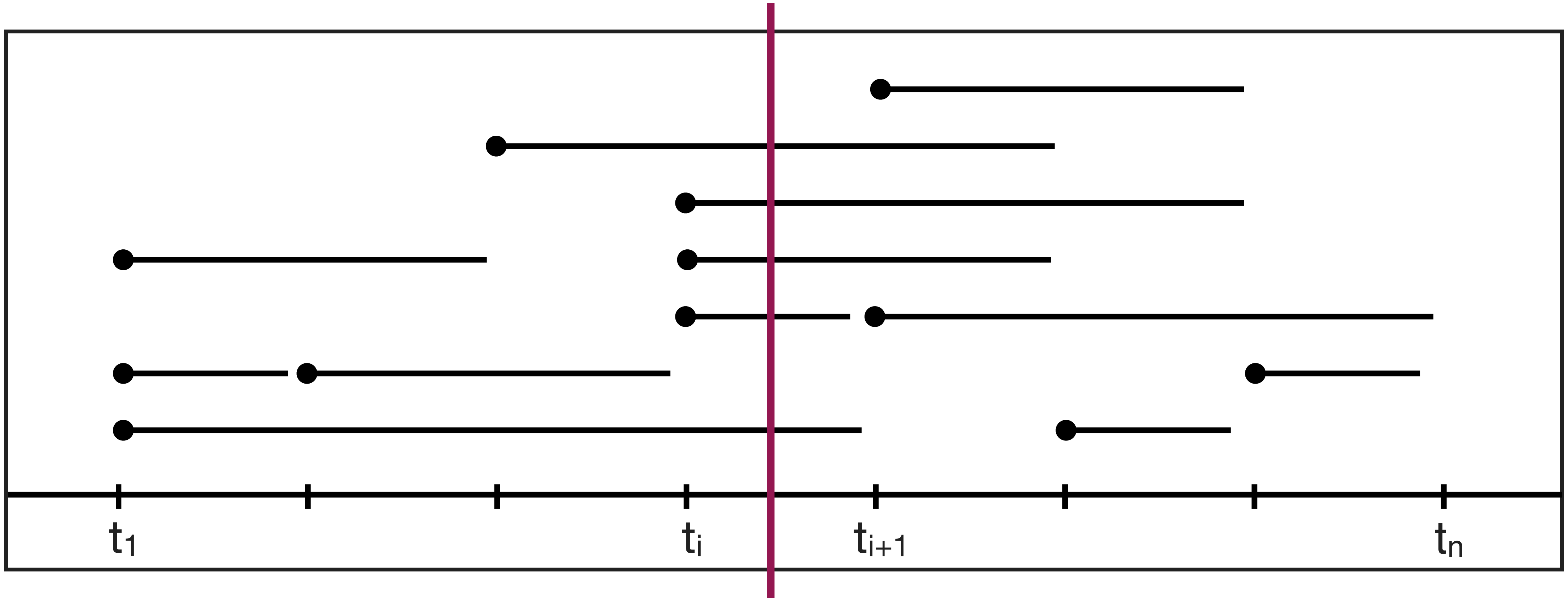
Index Partitioning

- Problem: Query processing needs to read entire posting lists, although **many postings can be discarded** for a query q@t



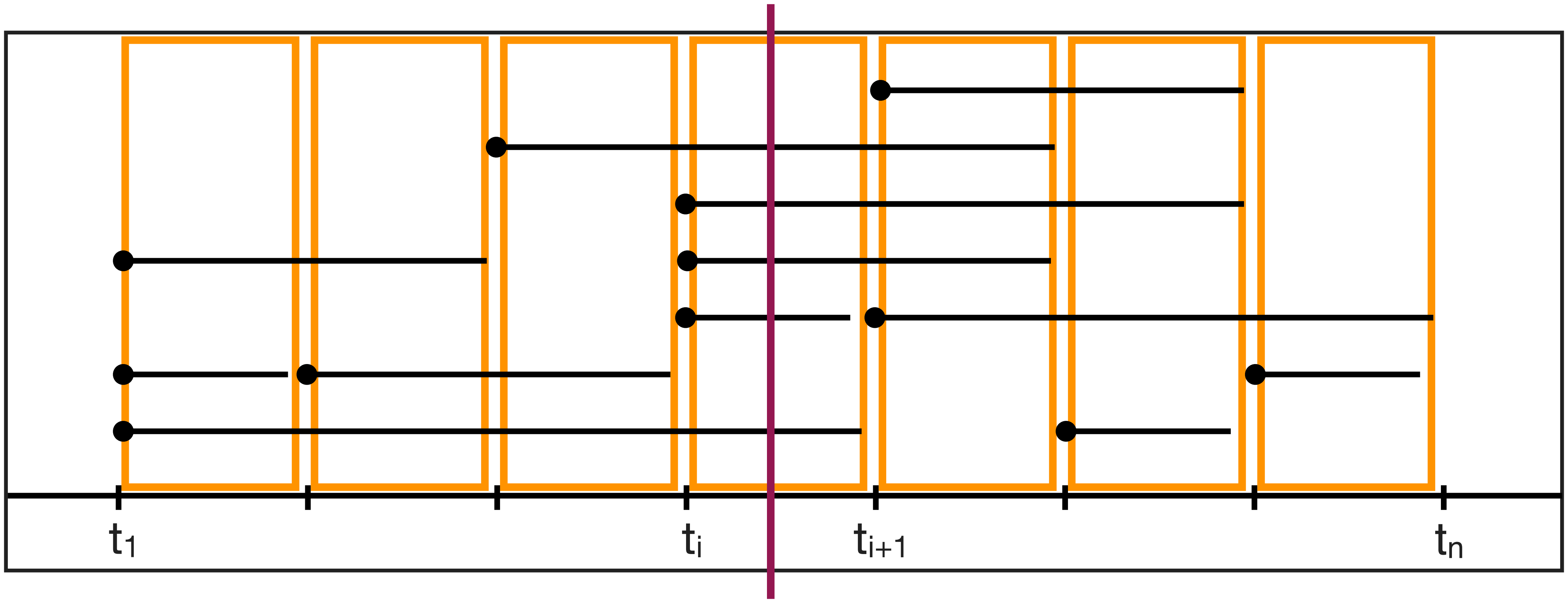
- Idea: **Partition** each posting list **along the time dimension**, so that the posting list for time interval $[t_i, t_j)$ contains all postings whose valid-time interval overlaps with it

Index Partitioning



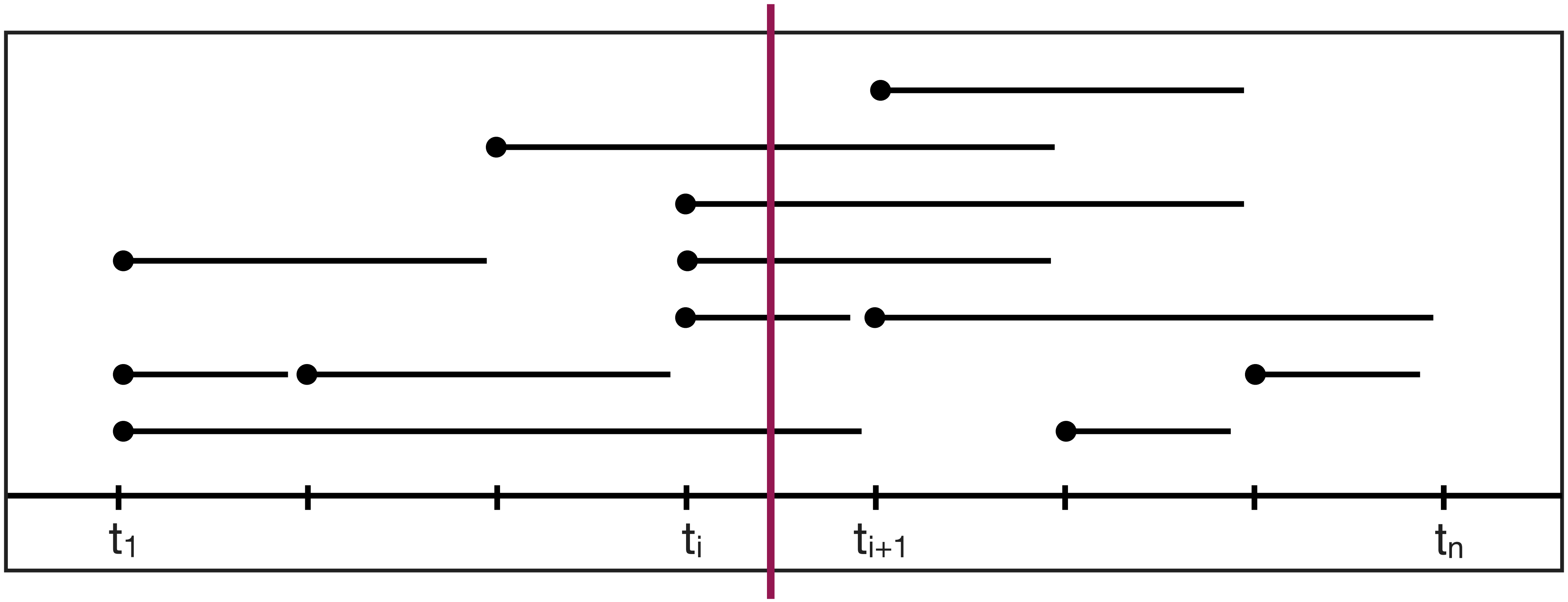
- **Trade-off** between index size and query-processing performance
  - **space optimal** S_{opt} (poor performance): use a single partition $[t_1, t_n)$
  - **performance optimal** P_{opt} (poor space): use partitions $[t_i, t_{i+1})$

Index Partitioning



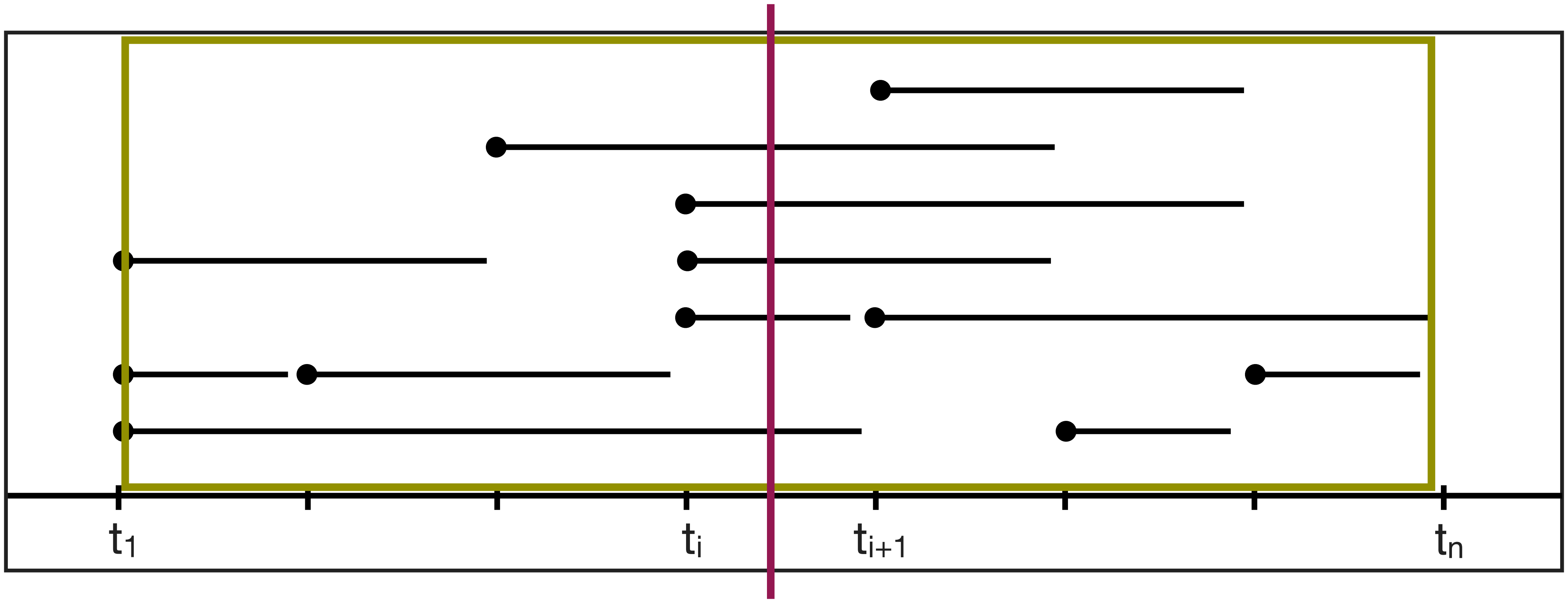
- **Trade-off** between index size and query-processing performance
  - **space optimal** S_{opt} (poor performance): use a single partition $[t_1, t_n)$
  - **performance optimal** P_{opt} (poor space): use partitions $[t_i, t_{i+1})$

Index Partitioning



- **Trade-off** between index size and query-processing performance
  - **space optimal** S_{opt} (poor performance): use a single partition $[t_1, t_n)$
  - **performance optimal** P_{opt} (poor space): use partitions $[t_i, t_{i+1})$

Index Partitioning



- **Trade-off** between index size and query-processing performance
    - **space optimal** S_{opt} (poor performance): use a single partition $[t_1, t_n)$
    - **performance optimal** P_{opt} (poor space): use partitions $[t_i, t_{i+1})$

Index Partitioning

- Idea: Define **optimization problem** to systematically trade off index space vs. query-processing performance
  - determine a **partitioning** P of $[t_1, t_n)$
  - s(P) : number of postings under partitioning P
  - c(t, P) : number of postings read to process time point t under P
- **Performance guarantee PG** ensures that cost for any time point is within a factor γ of best performance achieved by $\mathbf{P_{opt}}$

$$\arg\min_P s(P) \quad \text{s.t.} \quad \forall\, t \in [t_1, t_n) \;:\; c(t, P) \leq \gamma \cdot c(t, P_{opt})$$

- Optimal solution computable using **dynamic programming** over **prefix subproblems** $[t_1, t_i)$

7.4. Historical Document Collections

- Improved **digitization methods** (e.g., OCR) have resulted in (very) **old documents** now being **digitally available**

- <u>Examples</u>:
    - The New York Times Archive (1851 – today)
    - The Times Archive (1785 – now)
    - Google Books (~1500 – now)
    - HathiTrust (~1500 – now)

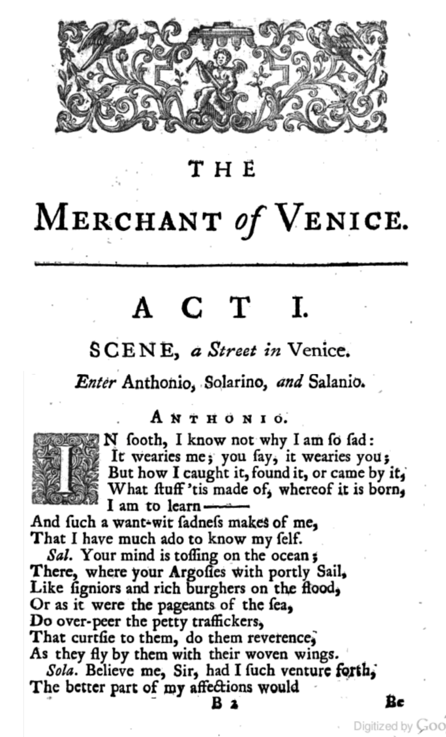

"All the News That's Fit to Print."

The New York Times.

TITANIC SINKS FOUR HOURS AFTER HITTING ICEBERG; 866 RESCUED BY CARPATHIA, PROBABLY 1250 PERISH; ISMAY SAFE, MRS. ASTOR MAYBE, NOTED NAMES MISSING

The Lost Titanic Being Towed Out of Belfast Harbor.

PARTIAL LIST OF THE SAVED.


THE

MERCHANT of VENICE.

ACT I.

SCENE, a Street in Venice.

Enter Anthonio, Solarino, and Salanio.

ANTHONIO.

IN ſooth, I know not why I am ſo ſad:
It wearies me; you ſay, it wearies you;
But how I caught it, found it, or came by it,
What ſtuff 'tis made of, whereof it is born,
I am to learn——
And ſuch a want-wit ſadneſs makes of me,
That I have much ado to know my ſelf.
Sal. Your mind is toſſing on the ocean;
There, where your Argoſies with portly Sail,
Like ſigniors and rich burghers on the flood,
Or as it were the pageants of the ſea,
Do over-peer the petty traffickers,
That curtſie to them, do them reverence;
As they fly by them with their woven wings.
Sola. Believe me, Sir, had I ſuch venture forth,
The better part of my affections would

B 2 Be

Digitized by Google

Historical Document Collections

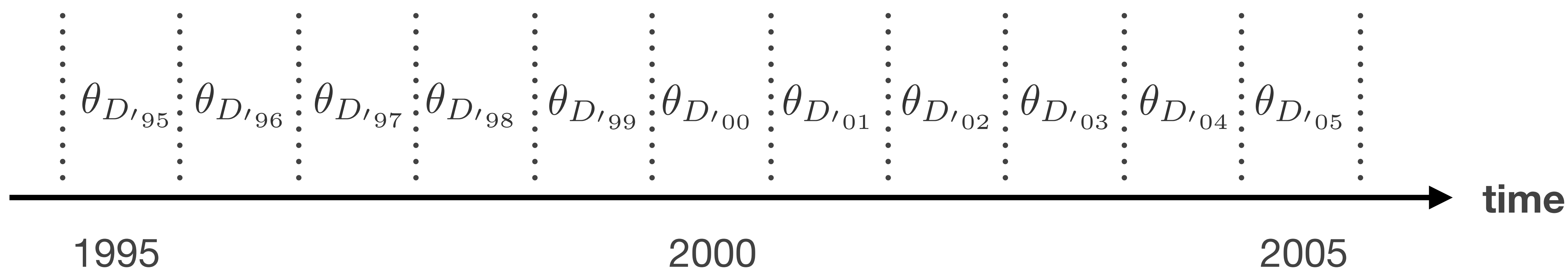
- Challenges & Opportunities:
  - **unknown publication dates** of documents can be estimated based on similar documents with known publication dates
  - **vocabulary gap** between today's queries and old documents needs to be bridged for effective information retrieval
  - **longitudinal document collections** allow analyses that give insights into, e.g., the evolution of language and historic events

7.4.1. Document Dating

- Problem: **Publication dates** of documents are **unknown**
  - in **historical document collections** due to lack of information
  - on the **Web** due to unreliable usage of the HTTP last-modified field
- de Jong et al. [5] employ **language models** to date documents
- Requirements: Document collection D with **known dates** which
  - is **sufficiently large** to avoid overfitting to individual documents
  - covers the **same domain** as the documents to be dated
  - covers the **period** from which documents to be dated originate

Document Dating

- Fix a **temporal granularity** (e.g., decade, year, month) and **partition** the document collection D into **disjoint partitions** $D_1, \ldots, D_n$ so that all documents in D_i have been published during the **i**-th **time period** (e.g., decade)



- **Unigram language model** with Dirichlet smoothing θ_{D_i} is estimated for each partition D_i

Document Dating

- **Document with unknown publication date** d is dated as having been published in time interval i*

$$\arg\min_{i*} KL(\theta_{D_{i*}} \| \theta_d)$$

- Approach achieves **precision of ~30%** in experiments on Dutch newspaper articles published between ’99 and ’05

7.4.2. Historical Document Retrieval

- Information retrieval on historical document collection suffers from a **vocabulary gap** between today's queries and old documents
  - **language evolution** (e.g., "and if he hear thee, thou wilt anger him")
  - **terminology evolution** (e.g., Leningrad/Saint Petersburg)
- Koolen et al. [6] treat the problem as a **cross-language information retrieval** problem by translating documents using **rewriting rules** mined from the document collection

Historical Document Retrieval

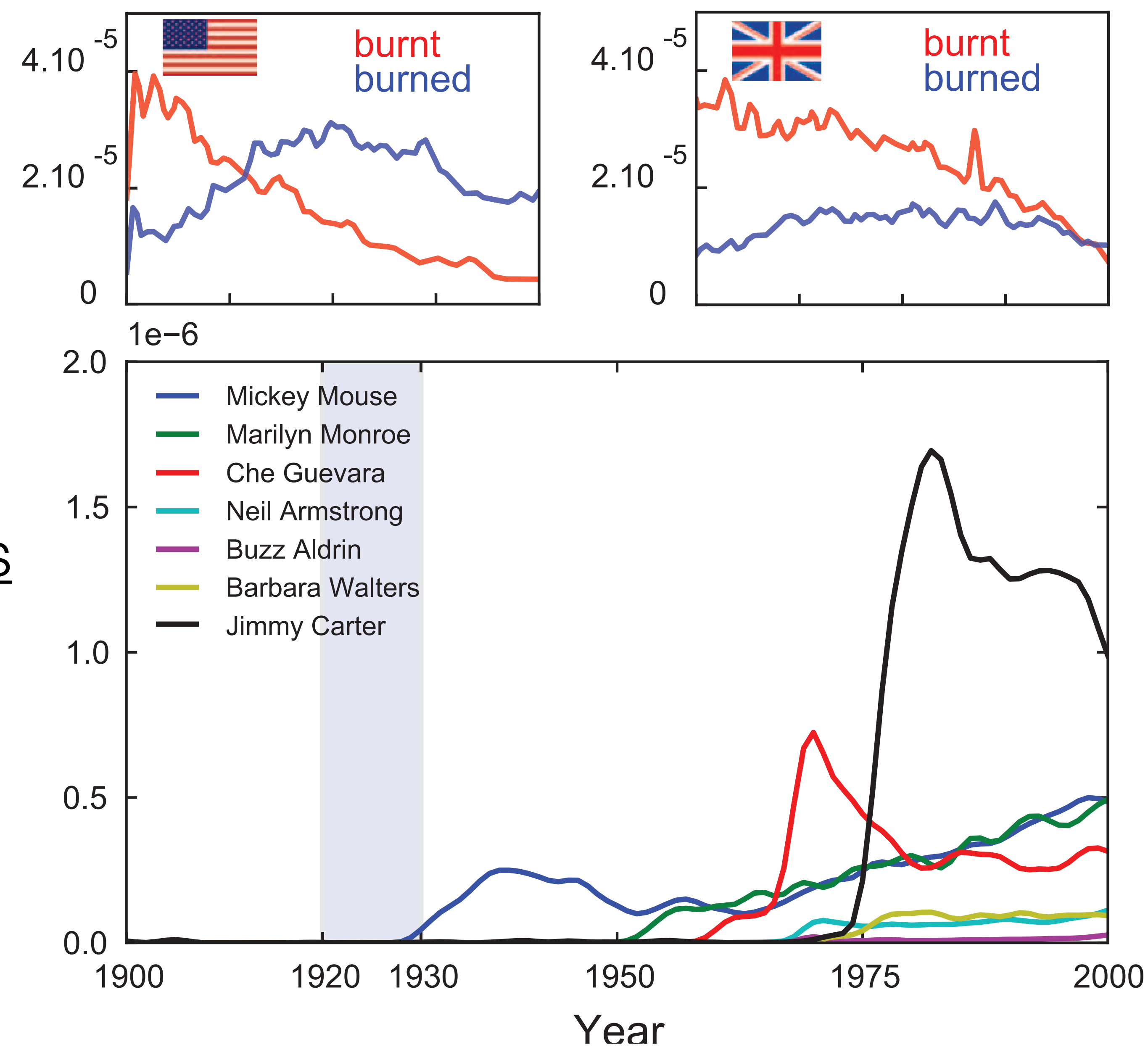
- **Phonetic Sequence Similarity**
  - **transcribe** historical and modern words **into phonemes**
    veeghen (historical) ⟶ v e g @ n, *vegen* (modern) ⟶ v e g @ n
  - find **pairs of historical and modern word** with same pronouncation
  - split words into **sequences of consonants and vowels**

historical:	**v**	**ee**	**gh**	**e**	**n**
modern:	**v**	**e**	**g**	**e**	**n**

  - **align sequences** and spot rewritings (e.g., ee ⟶ e, gh ⟶ g)
  - rewritings that are **often observed** become **rewriting rules**

7.4.3. Culturomics

- Michel et al. [8] use **n-gram statistics** computed for every year in the Google Books corpus to conduct analysis, e.g., about
    - **language evolution**
    - **popularity of celebrities**
    - historic events
- **Data & Demo** available at: https://books.google.com/ngrams



Summary

- **Web is highly dynamic**, hyperlinks more than web pages more than shingles; degree of dynamics depends on characteristics of the website and/or web page

- **Temporal information** (e.g., publication dates and temporal expressions) can be leveraged for more effective IR

- **Web archives** keep often highly-similar old snapshots of web pages, allowing for efficient indexing and time-travel text search

- **Historical document collections** contain documents published long time ago, are challenging to search, but insightful to analyze

References

[1] **E. Adar, J. Teevan, S. T. Dumais, J. L. Elsass**: *The Web Changes Everything: Understanding the Dynamics of Web Content*, WSDM 2009

[2] **K. Berberich, S. Bedathur, T. Neumann, G. Weikum:** *A Time Machine for Text Search*, SIGIR 2007

[3] **K. Berberich, S. Bedathur, O. Alonso, G. Weikum:** *A Language Modeling Approach for Temporal Information Needs*, ECIR 2010

[4] **F. de Jong, H. Rohde, D. Hiemstra:** *Temporal Language Models for the Disclosure of Historical Text*, Royal Netherlands Academy of Arts and Sciences, 2005

[5] **W. Dakka, L. Gravano, P. G. Ipeirotis:** *Answering General Time-Sensitive Queries*, TKDE 24(2), 2012

References

[6] **M. Koolen, F. Adriaans, J. Kaamps, M. de Rijke:** *A Cross-Language Approach to Historic Document Retrieval*, ECIR 2006

[7] **X. Li and W. B. Croft:** *Time-Based Language Models*, CIKM 2003

[8] **J.-B. Michel et al.:** *Quantitative Analysis of Culture Using Millions of Digitized Books*, Science 331, 2011

[9] **A. Ntoulas, J. Cho, C. Olston:** *What's New on the Web? The Evolution of the Web from a Search Engine Perspective*, WWW 2004

[10] **S. Schleimer, D. S. Wilkerson, A. Aiken:** *Winnowing: Local Algorithms for Document Fingerprinting*, SIGMOD 2003

[11] **J. Zhang and T. Suel:** *Efficient Search in Large Textual Collections with Redundancy*, WWW 2007