Document Representation and Retrieval

Unstructured Data

The University of Western Ontario

Goals

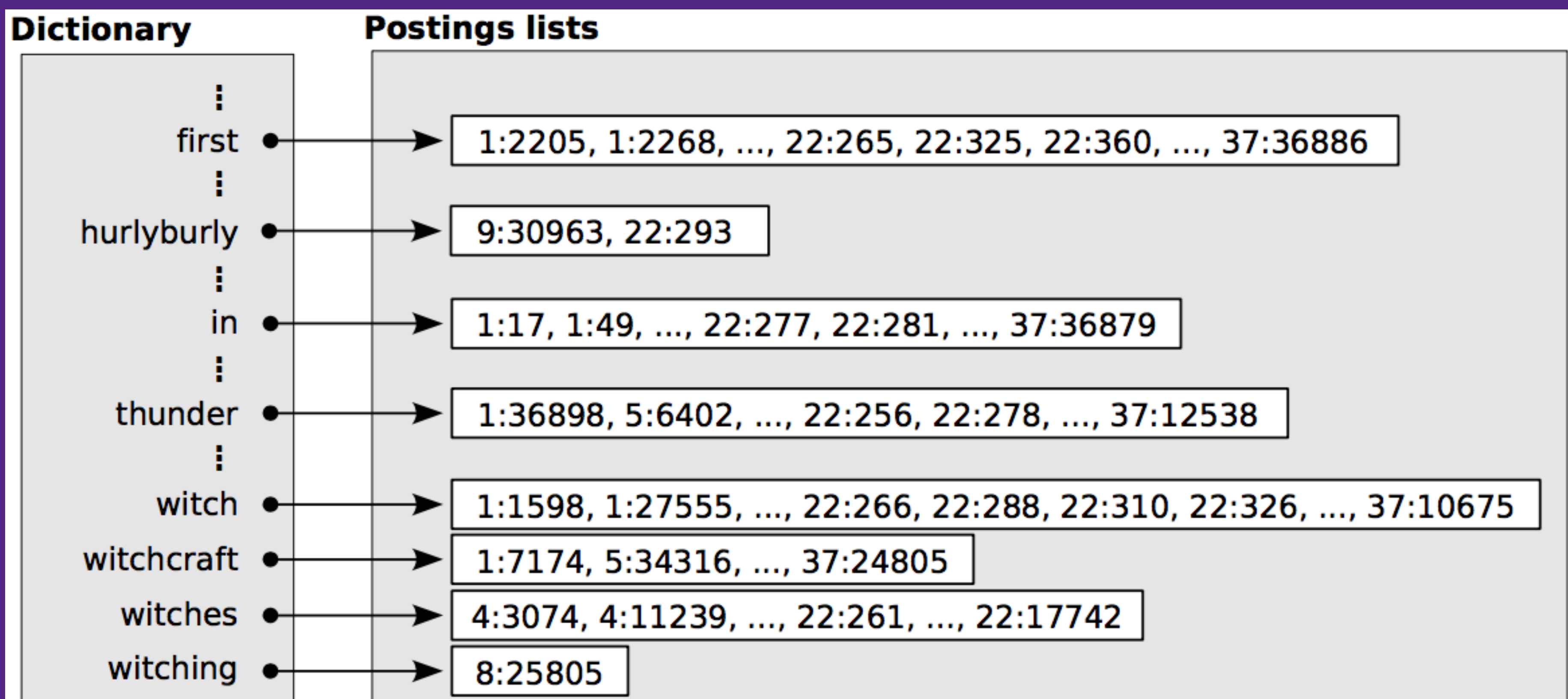
- Discuss information retrieval methods from the 60s to the state of the art
- Understand why developments were made over time in this area
- Understand how different strategies produce different results

- We will discuss these same strategies in future for different tasks (besides retrieval.)

Term-based Search and Boolean Search

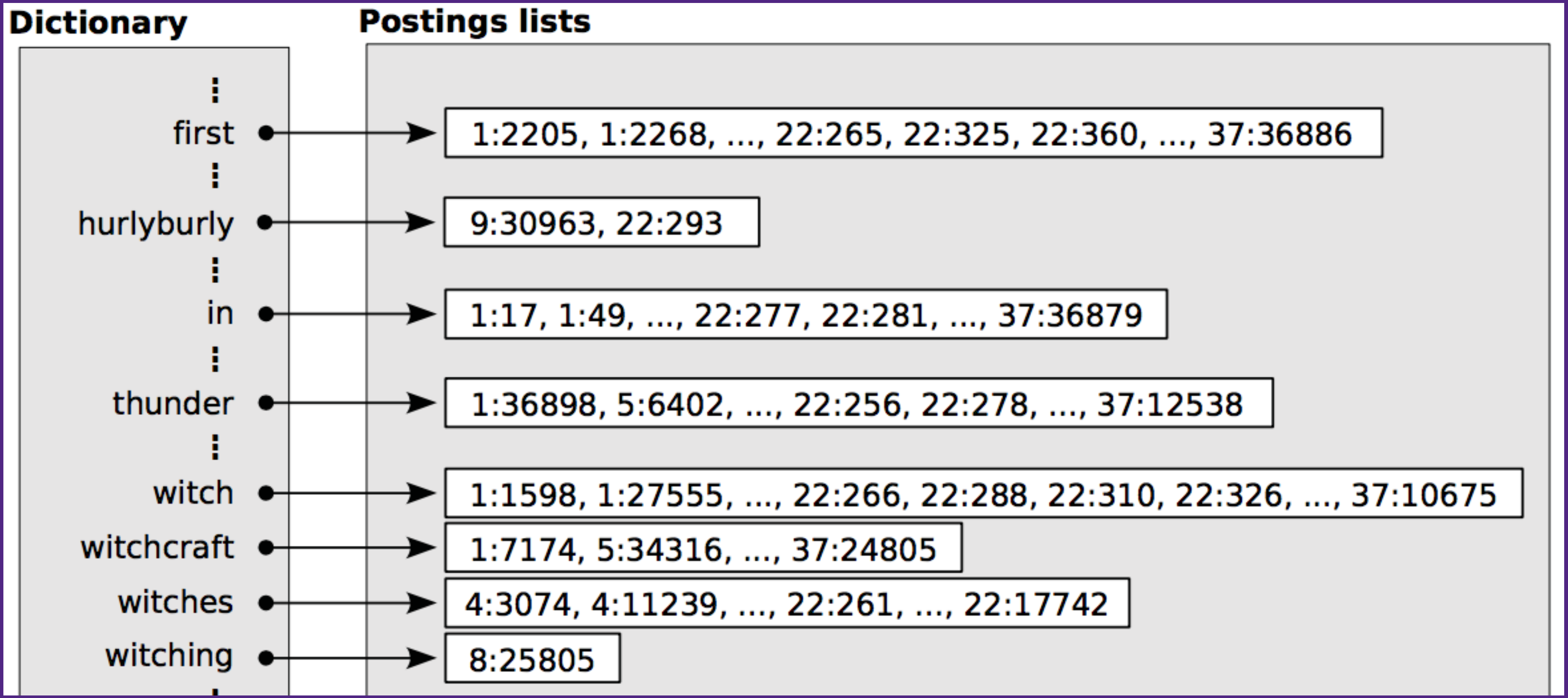
Term search

- Are there documents that contain the term "witch" in our corpus?
- A) yes B) No



Term search

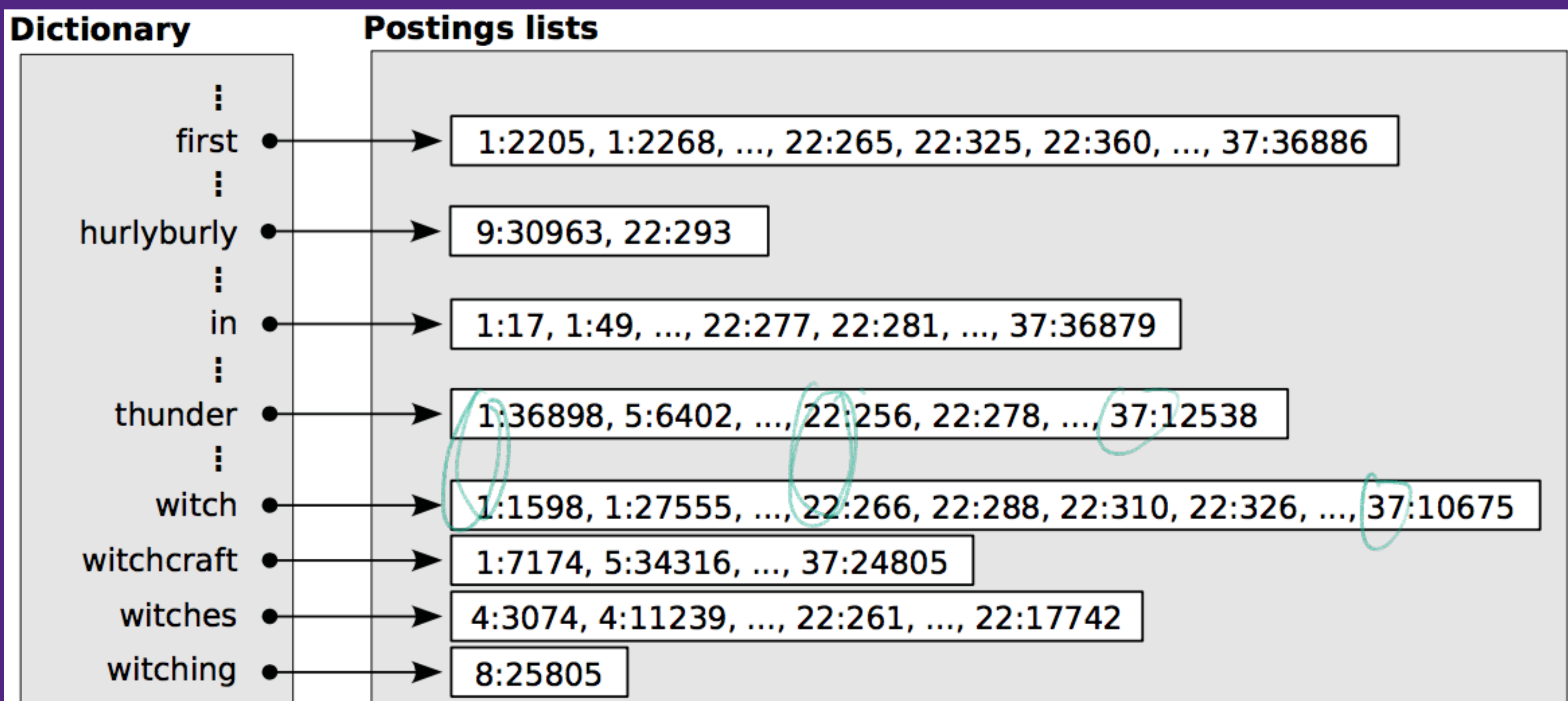
- Are there documents that contain the term "witch" in our corpus? Which one?
- A) 22 B) 1 C) 22 and 1 D) 37 E) A, B, and D



Boolean search

- Which documents that contain the term “witch” AND the word “thunder” in our corpus? D

A) 1 B) 22 C) 37 D) A,B, and C E) None

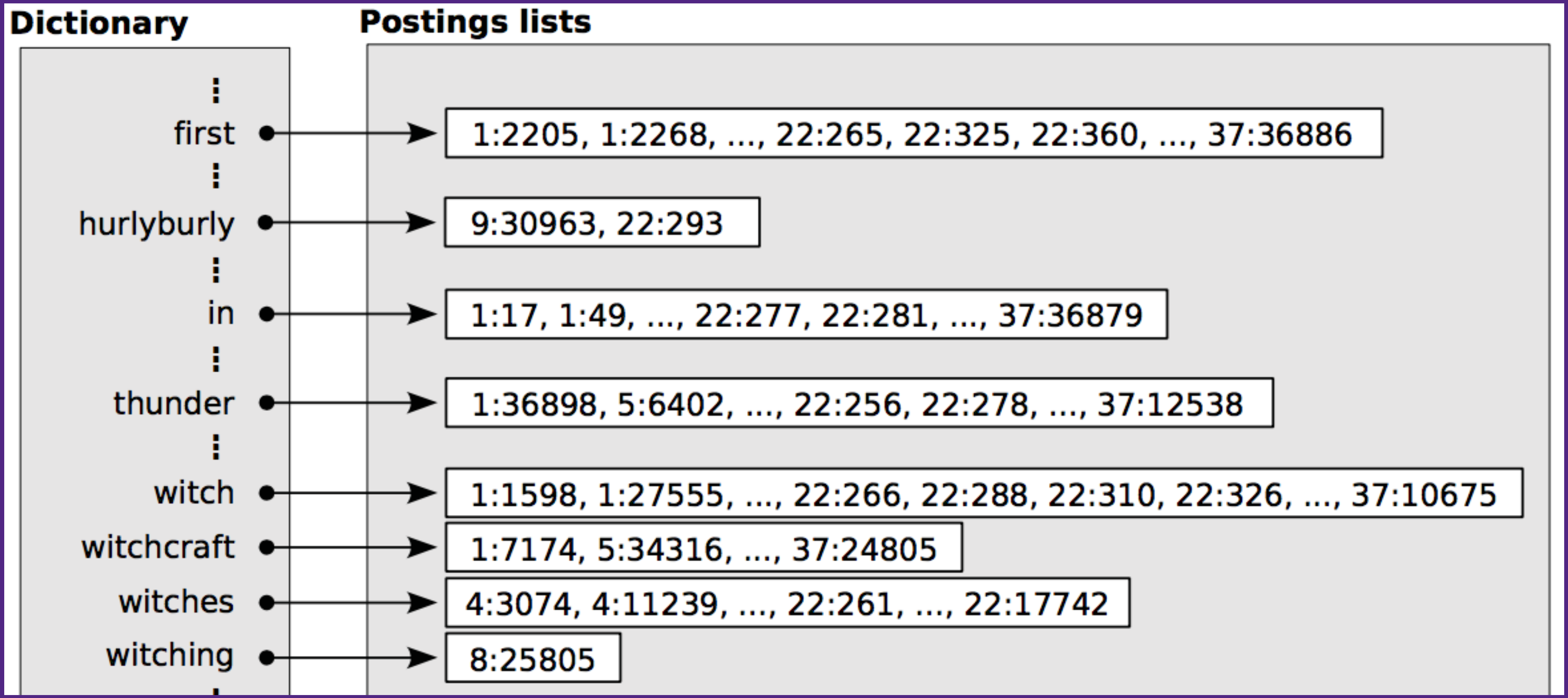


The “Boolean model” of retrieval

- (witch AND thunder) -> Docs 1, 22, 37, …
- (witch OR thunder) -> Docs 1, 5, 22, 37, …
- (witch OR witches OR witching) AND (NOT thunder) -> Doc 8

(hurlyburly AND witching)

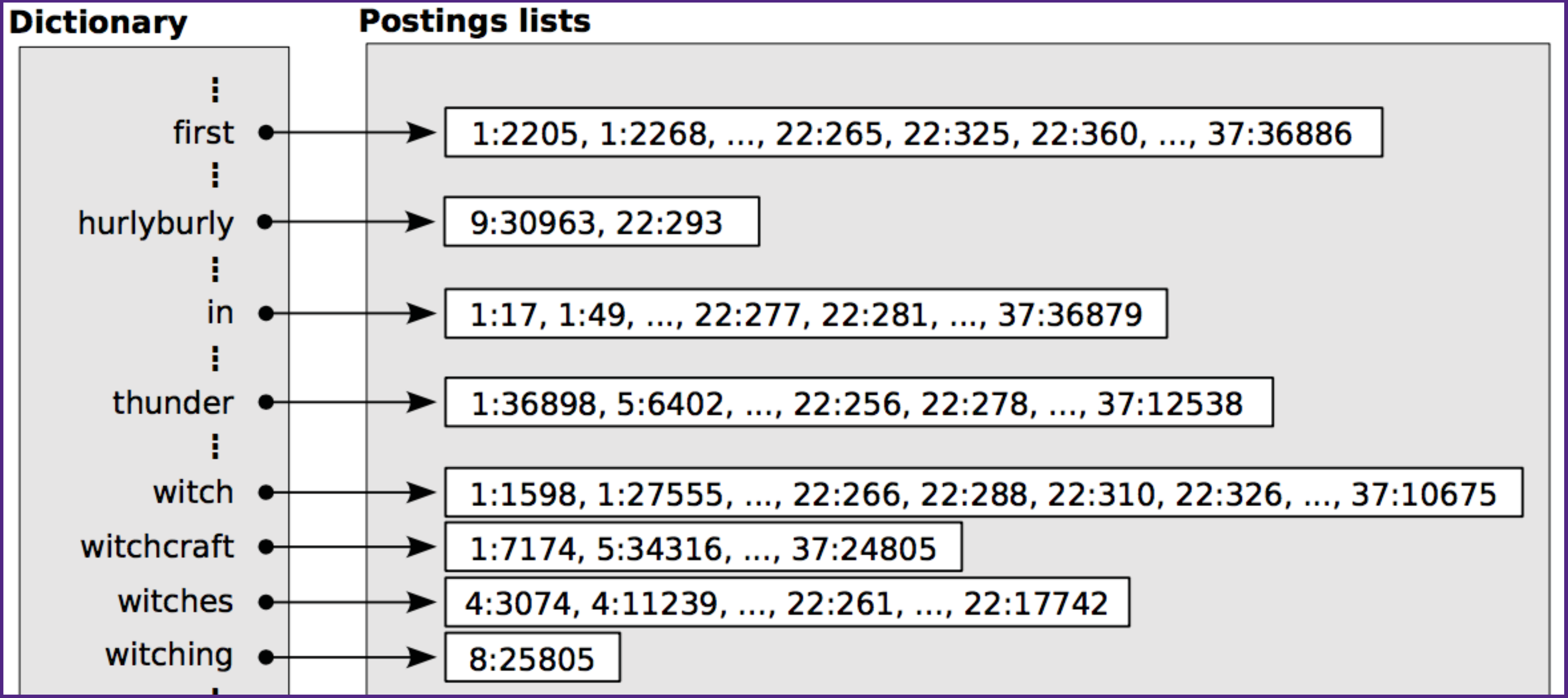
- A) yes B) No



(hurlyburly AND witching)

- What if we stem?

A) yes B) No



Query processing

- If you stem or stop the documents, should probably stem or stop the query. (Why?)
- Tell the user?

Boolean Search – Pros and Cons

Pros

Cons

Boolean Search – Pros and Cons

Pros

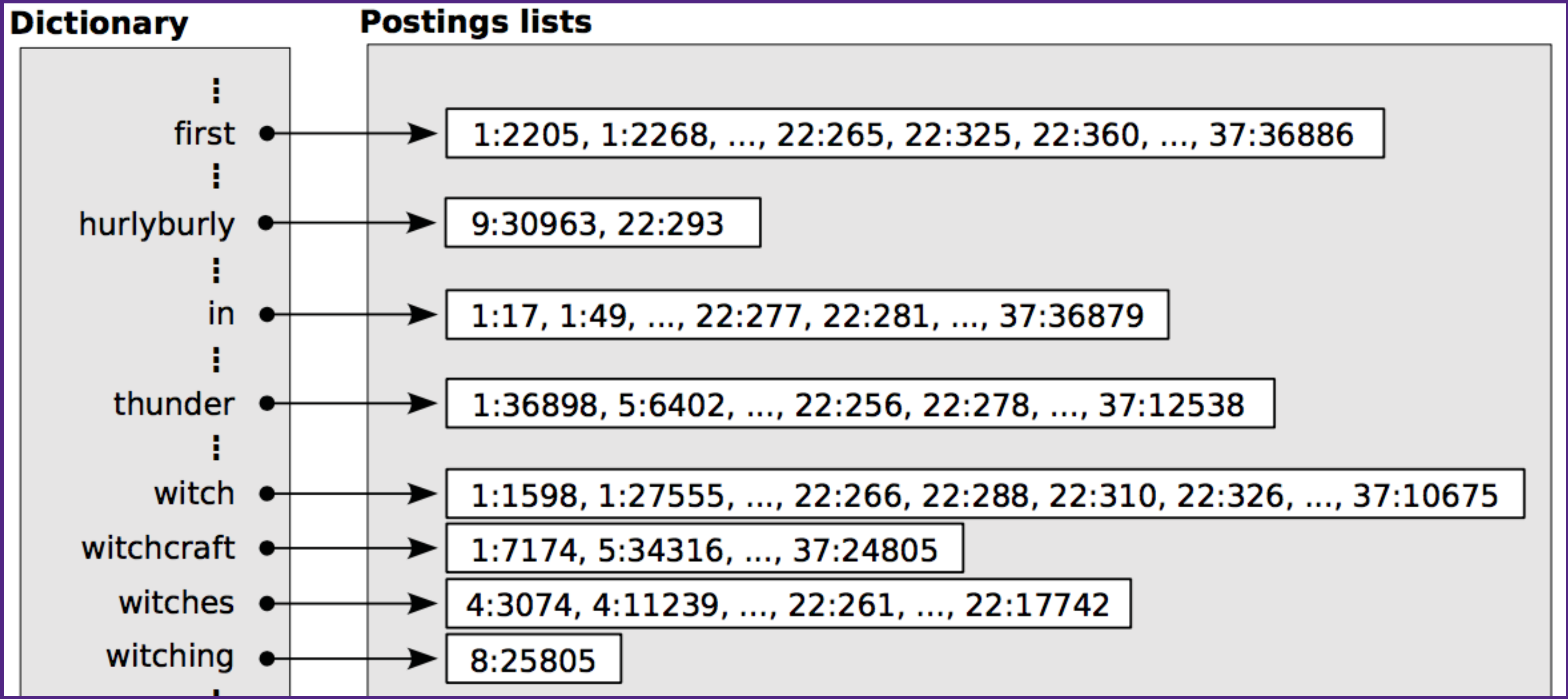
- Easy to understand*
  - *if preprocessing is understood
- Comprehensive results
  - Important for tasks like systematic review
- Efficient

Cons

- “Feast or famine"
- Coarse (one “witch” is same as 1000 “witch”)
- Can be cumbersome
  - May lead to queries trying desperately to filter out irrelevant documents

Aside: Phrase search

- Are there documents with the phrase "first witch" in our corpus? A) Yes B) No



Vector Representations

Vectors

- Ordered list of p numbers, $\mathbf{v} = [v_1, v_2, \ldots, v_p]$
- Add, subtract gives new vector of same length
  - $\mathbf{v} + \mathbf{w} = [v_1+w_1, v_2+w_2, \ldots, v_p+w_p]$
  - $\mathbf{v} - \mathbf{w} = [v_1-w_1, v_2-w_2, \ldots, v_p-w_p]$

the length after added. Subtracted are the same.

- Multiplication by a matrix can produce different length
- Dot Product gives scalar (single number)
  - $\mathbf{v} \cdot \mathbf{w} = v_1w_1 + v_2w_2 + \ldots + v_pw_p$

Vector Representation

informatics. system

- Consistent way of mapping an *object* (e.g. word, document, image, video, whatever) to a *vector*.
- "Consistent" means "same object maps to same vector."
- We then operate on the vectors instead of the original objects to learn and use structure

“One-hot” Encoding

first	1	0	0	0	0	0	0	0	…
hurlyburly	0	1	0	0	0	0	0	0	…
in	0	0	1	0	0	0	0	0	…
thunder	0	0	0	1	0	0	0	0	…
witch	0	0	0	0	1	0	0	0	…
witchcraft	0	0	0	0	0	1	0	0	…
witches	0	0	0	0	0	0	1	0	…
…	…	…	…	…	…	…	…	…	…

Bag of Words Representation

The Bag of Words “Vector model” Dense Representation

DocID	first	hurlyburly	in	thunder	witch	witchcraft	witches	witching	…
1	2	0	2	1	2	2	0	0	…
4	0	0	0	0	0	0	2	0	…
5	0	0	0	1	0	1	0	0	…
8	0	0	0	0	0	0	0	1	…
9	0	1	0	0	0	0	0	0	…
22	3	1	2	2	4	0	2	0	…
37	1	0	1	1	1	1	0	0	…
…	…	…	…	…	…	…	…	…	…

The Bag of Words "Vector model" Sparse Representation

DocID	Words
1	first:2, in:2, thunder:1, witch:2, witchcraft:2
4	witches:2
5	thunder:1, witchcraft:1
8	witching:1
9	hurlyburly:1
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2
37	first:1, in:1, thunder:1, witch:1, witchcraft:1
…	…

Query representation

- A query is a (tiny) document
- “thunder witchcraft” -> {thunder:1, witchcraft:1}

Document Similarity

Dot Product

BoW Similarity

- Which is most similar to {witchcraft:1, thunder:1}? Why? B.

A) 1 B) 5 C) 22 D) 37 E) All of above

DocID	Words
1	first:2, in:2, thunder:1, witch:2, witchcraft:2
4	witches:2
5	thunder:1, witchcraft:1
8	witching:1
9	hurlyburly:1
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2
37	first:1, in:1, thunder:1, witch:1, witchcraft:1
…	…

BoW Dot product

- Let *d*[*term*] be count of term in document *d*, *q*[*term*] be count of term in query *q*.

- Consider:
  - *d*[*term1*]**q*[*term1*] + *d[term2*]**q*[*term2*] + …
  - over all terms in our vocabulary

- When is this 0?
- When is it >0?
- Can it be negative?

Similarity – BoW Dot product

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	$1\times1+2\times1=3.$
4	witches:2	0
5	thunder:1, witchcraft:1	$1\times1+1\times1=2.$
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	$2\times1=2.$
37	first:1, in:1, thunder:1, witch:1, witchcraft:1	$1\times1+1\times1=2.$
…	…	

Similarity – BoW Dot product

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	3
4	witches:2	0
5	thunder:1, witchcraft:1	2
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	2
37	first:1, in:1, thunder:1, witch:1, witchcraft:1	2
…	…	

BoW Dot product similarity

- Mimics boolean "OR"
  - If at least one term matches, similarity > 0
  - If no terms match, similarity == 0
- More occurrence of matching terms -> higher similarity
- Now we can **rank** search results by similarity

Similarity – BoW Dot product

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	3
4	witches:2	0
5	thunder:1, witchcraft:1	2
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	2
37	first:10, in:10, thunder:5, witch:10, witchcraft:10	?
…	…	

Similarity – BoW Dot product

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	3
4	witches:2	0
5	thunder:1, witchcraft:1	2
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	2
37	first:10, in:10, thunder:5, witch:10, witchcraft:10	15
…	…	

BoW Dot product – Pros and Cons

Pros

- Fast to compute
- Similar to "OR"
  - Easy to understand
- Can rank results

Cons

- Sensitive to document length

Document Similarity

Cosine

Normalizing for document length

- Idea: document similarity "should not depend on the length of the documents"
- E.g., want similarity between query {first:1, witch:1} and
  - {first:1, witch:1}
  - {first:2, witch:2}
  - {first:5, witch:5}
  - …to all be the same.
- Divide by sqrt(sum$_i$ #term$_i^2$)
  - All become {first:0.707, witch:0.707}

Cosine similarity

- Let *d*[*term*] be count of *term* in document *d*, *q*[*term*] be count of *term* in query *q*.

- Let
  - $\|d\| = \mathrm{sqrt}(\mathrm{sum}_i\ (d[\mathrm{term}_i]^2))$
  - $\|q\| = \mathrm{sqrt}(\mathrm{sum}_i\ (q[\mathrm{term}_i]^2))$

- Cosine similarity of *d* and *q* is:
- $(d[term_1]*q[term_1] + d[term_2]*q[term_2] + \ldots)/(\|d\|*\|q\|)$

Cosine similarity

- Cosine similarity of *d* and *q* is:
- $(d[term_1]*q[term_1] + d[term_2]*q[term_2] + \ldots)/(\|d\|*\|q\|)$

- Example:

$\vec{x} = [x_1\ x_2\ x_3]$ $\vec{y} = [y_1\ y_2\ y_3]$

Cosine similarity of x and *y* is:

$$\frac{x_1y_1+x_2y_2+x_3y_3}{||x||||y||} \qquad ||x|| = \sqrt{x_1^2 + x_2^2 + x_3^2} \quad ||y|| = \sqrt{y_1^2 + y_2^2 + y_3^2}$$

$$\frac{x_1y_1+x_2y_2+x_3y_3}{\sqrt{x_1^2+x_2^2+x_3^2}\ \sqrt{y_1^2+y_2^2+y_3^2}}$$

Normalizing for document length

- Divide by sqrt(sum$_i$ #term$_i^2$)
  - All become {first:0.707, witch:0.707}
- Define similarity to be dot product of normalized document vectors
- Minimum similarity is 0, max similarity is 1 (assuming #term$_i$ all positive)
  - Think – what is the similarity between a document and itself?
- This is the **cosine of the angle between the vectors that represent the documents**

Similarity – BoW Cosine measure

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	0.514
4	witches:2	0
5	thunder:1, witchcraft:1	1.0
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	0.229
37	first:1, in:1, thunder:1, witch:1, witchcraft:1	0.632
…	…	

Similarity – BoW Cosine measure

DocID	Words	Similarity to {witchcraft:1, thunder:1}
1	first:2, in:2, thunder:1, witch:2, witchcraft:2	0.514
4	witches:2	0
5	thunder:1, witchcraft:1	1.0
8	witching:1	0
9	hurlyburly:1	0
22	first:3, hurlyburly:1, in:2, thunder:2, witch:4, witches:2	0.229
37	first:5, in:5, thunder:5, witch:5, witchcraft:5	0.632
…	…	

Multiplying either vector by a positive constant does not change cosine similarity.

Similarity – BoW Cosine measure

DocID	Words	Similarity to {baseball:1, season:1, opener:1}
1	baseball:10, season:1, opener:1	0.686
2	baseball:10, season:5	0.775
6	season:1	0.577
7	baseball:10	0.577
10	baseball:10, season:3	0.719
35	baseball:10, season:2	0.679
…	…	

BoW Cosine similarity

Pros

- Fast to compute
- Similar to "OR"
  - Easy to understand
- Can rank results
- Invariant to document length. (Multiplicative scaling of vectors.)

Cons

- Treats all words equally