Querying

Introduction to Information Retrieval
INF 141
Donald J. Patterson

Content adapted from Hinrich Schütze
http://www.informationretrieval.org

Parametric Search

- In these examples we select field values
  - Values could be hierarchical
    - USA -> California -> Orange County -> Newport Beach
- It is a paradigm for navigating through a corpus
  - e.g, “Aerospace companies in Brazil” can be found by combining “Geography” and “Industry”
    - (“Capulet”,”Romeo and Juliet) = 1
- Approach:
  - Filter for relevant documents
  - Run text searches on subset

Parametric Search

- Index support for parametric search
  - Must be able to support queries of the form:
    - Find pdf documents that contain “UCI”
    - Field selection and text query
- Field selection approach
  - Use inverted index of field values
    - (field value, docID)
    - organized by field name
    - Using same compression and sorting techniques

Parametric Search

- Now, we crawl the corpus
- We parse the document keeping track of terms, fields and docIDs
- Instead of building just a (term, docID) pair
- We build (term, field, docID) triples
- These can then be combined into postings like this:

William.author	2	4	8	16	32	64
William.title	1	2	3	5	8	13
William.abstract	1	3	5	7	9	11

Building up our query technology

- "Matching" search
  - Linear on-demand retrieval (aka grep)
  - 0/1 Vector-Based Boolean Queries
  - Posting-Based Boolean Queries
- Ranked search
  - Parametric Search
  - Zones

Zones

- A zone is an extension of a field
- A zone is an identified region of a document
  - e.g., title, abstract, bibliography
  - Generally identified by mark-up in a document
    - <title>Romeo and Juliet</title>
- Contents of zone are free text
  - Not a finite vocabulary
- Indices required for each zone to enable queries like:
  - (instant in TITLE) AND (oatmeal in BODY)
- Doesn't cover "all papers whose authors cite themselves"
  - Why?

Parametric Search

- So are we just creating a database?
  - Not really.
  - Databases have more functionality
    - Transactions
    - Recovery
      - Our index can be recreated. Not so with database.
    - Text is never stored outside of indices
- We are focusing on optimized indices for text-oriented queries not a full SQL engine

Building up our query technology

- "Matching" search
  - Linear on-demand retrieval (aka grep)
  - 0/1 Vector-Based Boolean Queries
  - Posting-Based Boolean Queries
- Ranked search
  - Parametric Search
  - Zones
  - Scoring

Scoring

- Boolean queries “match” or “don’t match”
- Good for experts with needs for precision and coverage
  - knowledge of corpus
  - need 1000’s of results
- Not good with non-expert users
  - who don’t understand boolean operators
  - or how they apply to search
  - or who don’t want 1000’s of results

Scoring

- Boolean queries require careful crafting to get the right number of results (Ferrari example)
- Ranked lists eliminate this concern
  - Doesn't matter how big the list is
- Scoring is the basis for ranking or sorting document that are returned from a query.
  - Ideally the score is high when the document is relevant
  - WLOG we will assume scores are between 0 and 1 for each doc.

Scoring

- First generation of scoring used a linear combination of Booleans

$$\begin{aligned} Score \quad = \quad & 0.6(oatmeal \in TITLE) + \\ & 0.3(oatmeal \in BODY) + \\ & 0.1(oatmeal \in ABSTRACT) \end{aligned}$$

- Explicit decision about importance of zone
- Each subquery is 0 or 1
- This example has a finite number of possible values
  - What are they?

Scoring

$$\begin{aligned} Score \quad = \quad & 0.6(oatmeal \in TITLE) + \\ & 0.3(oatmeal \in BODY) + \\ & 0.1(oatmeal \in ABSTRACT) \end{aligned}$$

- Subqueries could be *any* Boolean query
- Where do we get the weights? (e.g., 0.6,0.3,0.1)
  - Rarely from the user
  - Usually built into the query engine
    - Where does the query engine get them from?
      - Machine learning

Scoring Exercise

- Calculate the score for each document based on the weightings (0.1 author), (0.3 body), (0.6 title)
- For the query
  - "bill" or "rights"

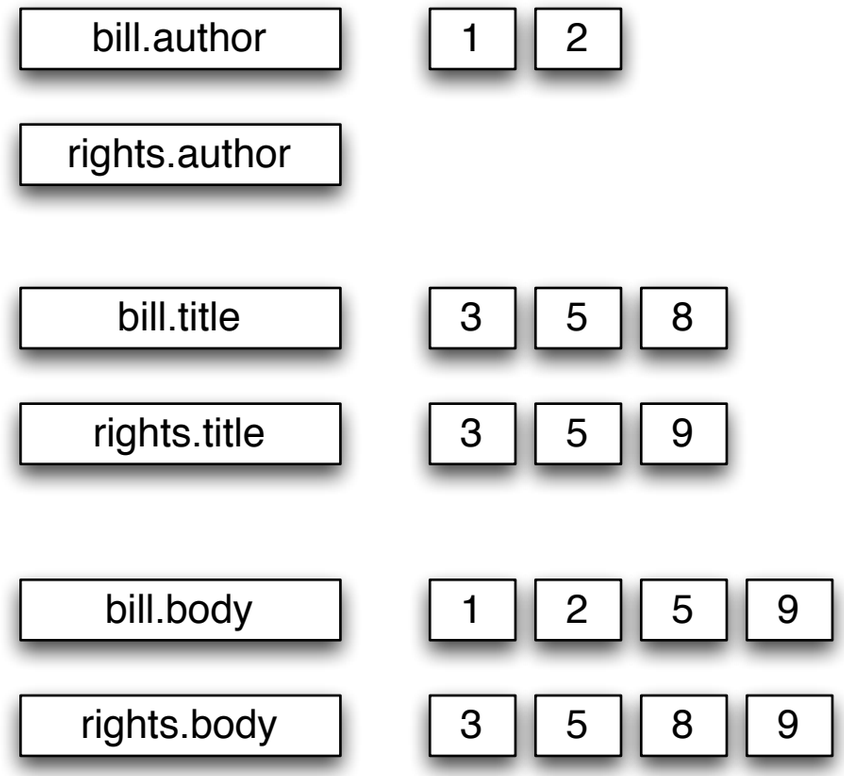
bill.author	1	2		
rights.author				
bill.title	3	5	8	
rights.title	3	5	9	
bill.body	1	2	5	9
rights.body	3	5	8	9

Building up our query technology

- "Matching" search
  - Linear on-demand retrieval (aka grep)
  - 0/1 Vector-Based Boolean Queries
  - Posting-Based Boolean Queries
- Ranked search
  - Parametric Search
  - Zones
  - Scoring

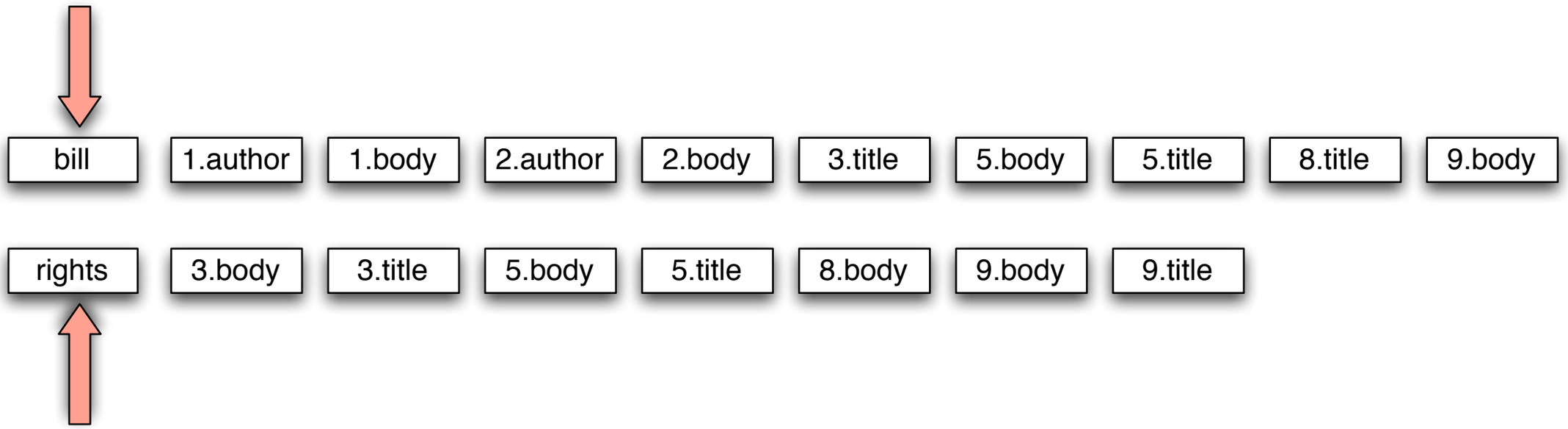
Zones combination index

- Encode the zone in the posting
- At query time accumulate the contributions to the total score from the various postings



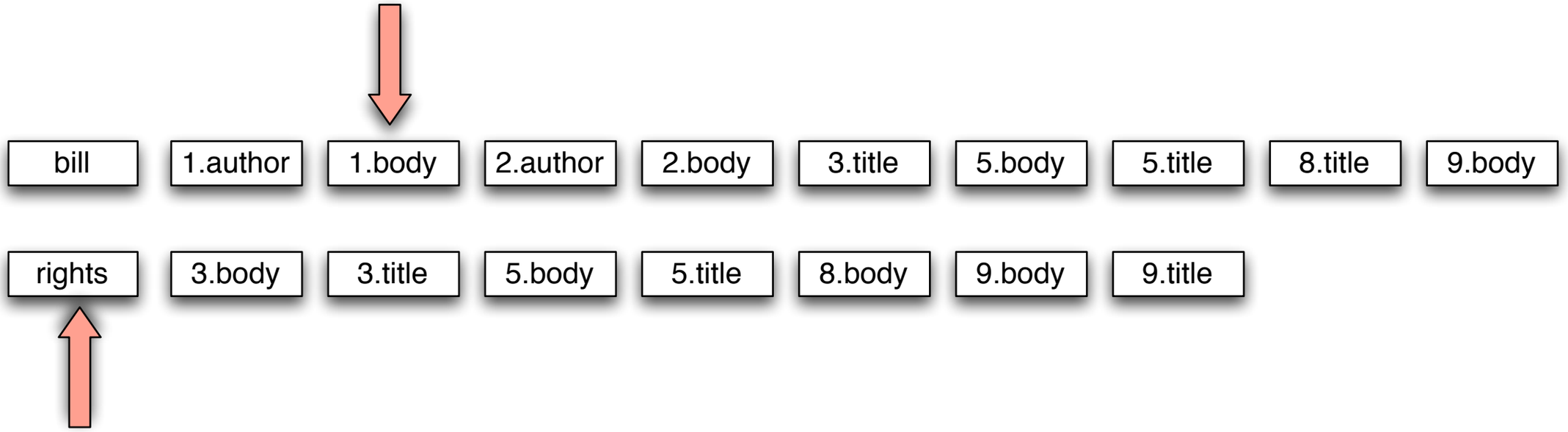
Zone scoring with zones combination index

"bill OR rights" (0.1 author), (0.3 body), (0.6 title)



Zone scoring with zones combination index

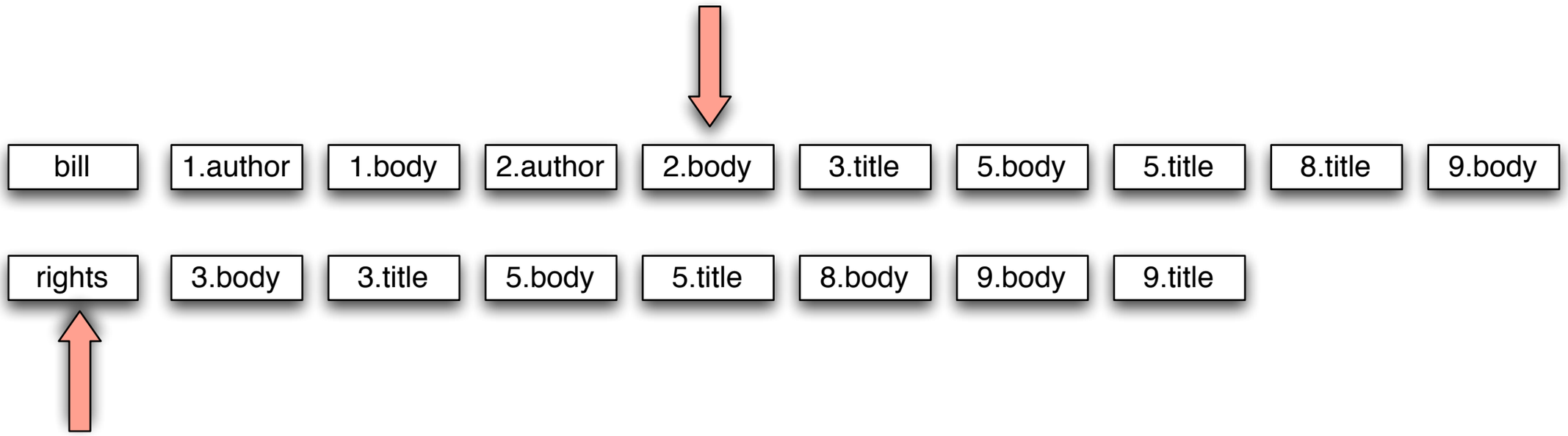
“bill OR rights” (0.1 author), (0.3 body), (0.6 title)



1: 0.4

Zone scoring with zones combination index

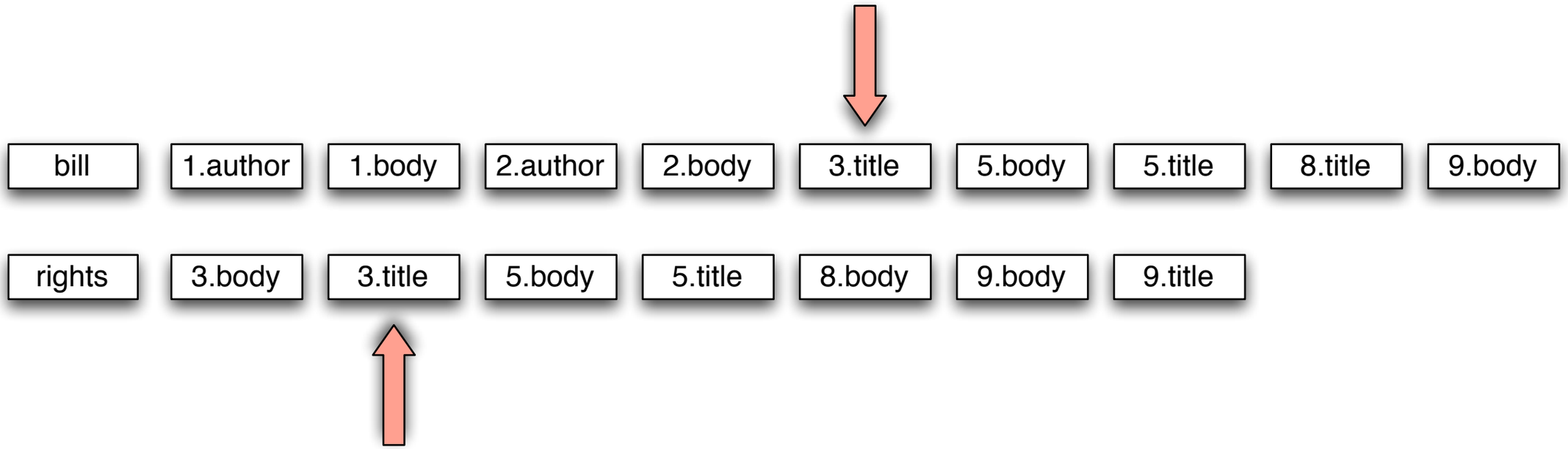
“bill OR rights” (0.1 author), (0.3 body), (0.6 title)



1: 0.4
2: 0.4

Zone scoring with zones combination index

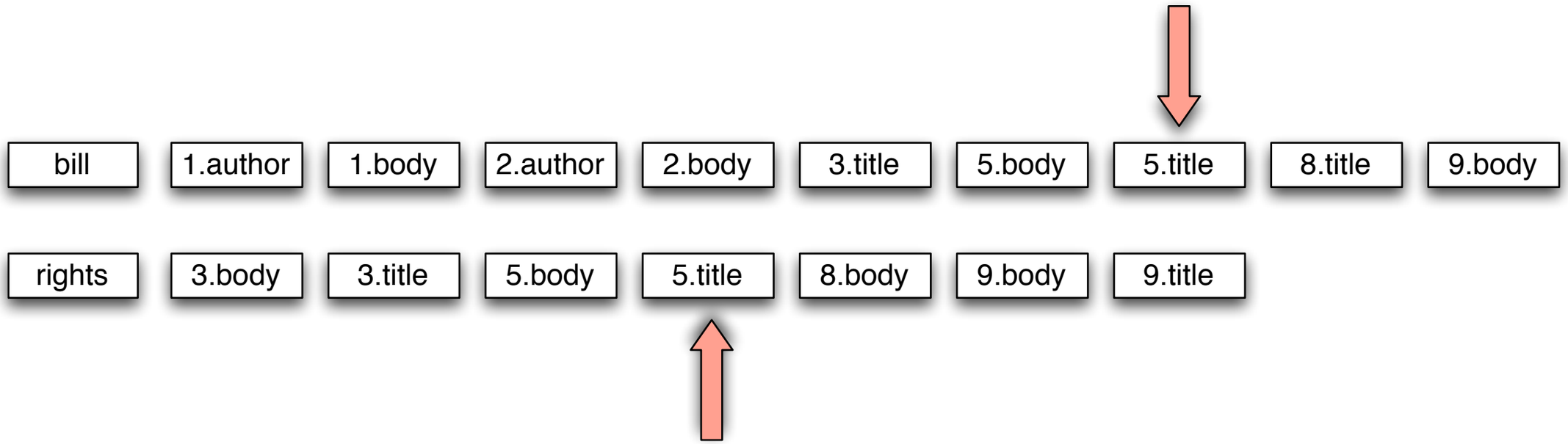
"bill OR rights" (0.1 author), (0.3 body), (0.6 title)



1: 0.4
2: 0.4
3: 0.9

Zone scoring with zones combination index

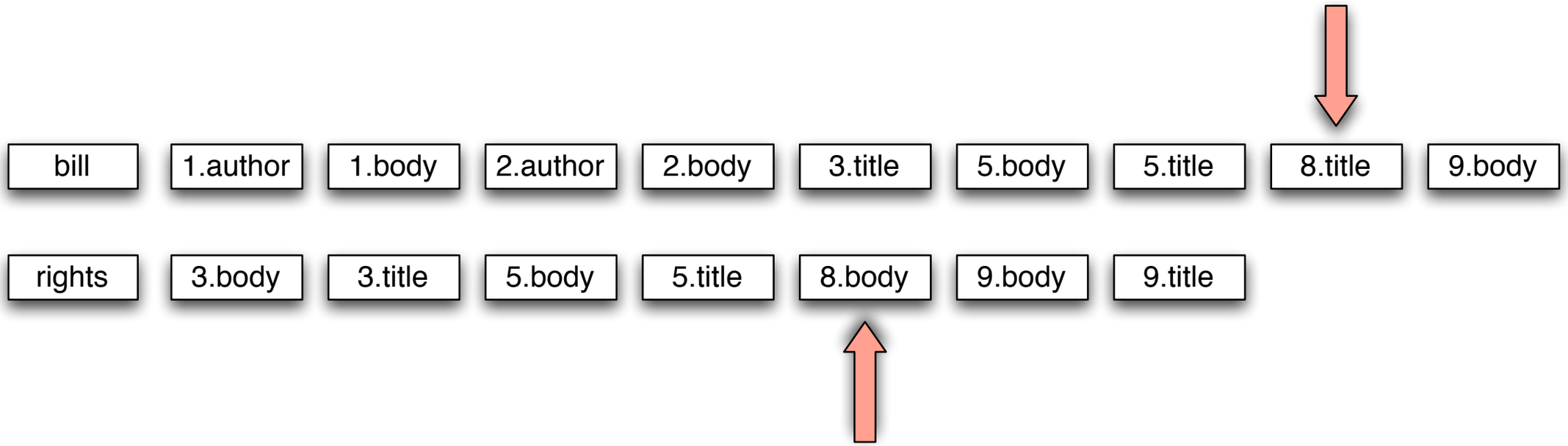
"bill OR rights" (0.1 author), (0.3 body), (0.6 title)



1: 0.4 5: 0.9
2: 0.4
3: 0.9

Zone scoring with zones combination index

“bill OR rights” (0.1 author), (0.3 body), (0.6 title)



1: 0.4 5: 0.9
2: 0.4 8: 0.9
3: 0.9

Zone scoring with zones combination index

"bill OR rights" (0.1 author), (0.3 body), (0.6 title)

1: 0.4 5: 0.9
2: 0.4 8: 0.9
3: 0.9 9: 0.9

Zone scoring with zones combination index

"bill OR rights" (0.1 author), (0.3 body), (0.6 title)

1: 0.4 5: 0.9
2: 0.4 8: 0.9
3: 0.9 9: 0.9

Results:
9,8,5,3,2,1