11-442 / 11-642 / 11-742:
Search Engines

# Best-Match Retrieval: HW2 Implementation

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

## Outline

**HW2 implementation**

- Indri default beliefs
- Window operator

## Indri Implementation

**Indri operators calculate scores <u>only for documents that contain a query term</u>**

- Use inverted or score lists
  - Similar to HW1
- Use document length, ctf, and corpus length for smoothing
  - Lookup from the index – see the HW2 web page

**But, one aspect is a little tricky to get right…**

$$
\begin{array}{llll}
\text{AND, COMBINE:} & p_{and}(q \mid d) & = & \prod_{q_i \in q} p(q_i \mid d)^{\frac{1}{|q|}} \\
\text{WAND, WEIGHT:} & p_{wand}(q \mid d) & = & \prod_{q_i \in q} p(q_i \mid d)^{\frac{w_i}{w}}, \quad w = \sum w_i \\
\text{OR:} & p_{or}(q \mid d) & = & 1 - \prod_{q_i \in q} \left(1 - p(q_i \mid d)\right) \\
\text{WSUM:} & p_{wsum}(q \mid d) & = & \sum_{q_i \in q} \frac{w_i}{w} p(q_i \mid d), \quad w = \sum w_i \\
\text{NOT:} & p_{not}(q \mid d) & = & 1 - p(q \mid d)
\end{array}
$$

3

## Indri Review

**Your HW2 system will use two-stage smoothing to compute term weights**

$$p(q_i|d) = (1-\lambda)\frac{tf_{q_i,d} + \mu\, p_{MLE}(q_i|C)}{length_d + \mu} + \lambda\, p_{MLE}(q_i|C)$$

$$p_{MLE}(q_i|C) = \frac{ctf_{q_i}}{length_C}$$

This is the #SCORE operator for the Indri retrieval model

4

## Indri Implementation

**Query:** #OR (a #AND (b c) ) **Document:** a

**Query terms b and c do not appear in this document**

**… what is the score of the #AND operator?**

- $tf_b = 0$ $tf_c = 0$
- Do the usual Indri score calculation
  - – MLE p(t | d) scores are 0 for b and c, so only <u>smoothing</u> scores for b and c

$$\boxed{p(q_i|d) = (1-\lambda)\frac{tf_{q_i,d} + \mu\, p_{MLE}(q_i|C)}{length_d + \mu} + \lambda\, p_{MLE}(q_i|C)}$$

**This is simple conceptually, but <u>how is it implemented</u>?**

- You don't want to calculate #AND scores for <u>every</u> document (<u>way</u> too slow)

  … just the documents that have at least one query term

5 © 2021 Jamie Callan

## Indri Implementation

**Query:** #OR (a #AND (b c) ) **Document:** a

**Add a <u>new method</u> to <u>all</u> QrySop operators**

double getDefaultScore (RetrievalModel r, long docid)

**When <u>any</u> QrySop operator calculates scores for the <u>Indri retrieval model</u>**

```
if $q_i$ has a match for document d
then
   call $q_i$.getScore
else
   call $q_i$.getDefaultScore
```

6 © 2021 Jamie Callan

## Indri Implementation

**Query:** #OR (a #AND (b c) )  **Document:** a

**QrySopScore.getDefaultScore (RetrievalModel r, long docid)**

- The standard Indri SCORE calculation done with tf=0

  If r == RetrievalModel.Indri

$$p_{scoreDefault}(t \mid \text{docid}) = (1-\lambda)\frac{\boxed{0} + \mu\, p_{MLE}(t \mid C)}{length(\text{docid}) + \mu} + \lambda\, p_{MLE}(t \mid C)$$

**This is the main difference.**
**Do the usual calculation, but with tf=0.**

7 © 2021 Jamie Callan

## Indri Implementation

**Query:** #AND (a #NEAR/3 (b c) )

**QrySopScore.getDefaultScore (RetrievalModel r, long docid)**

**What happens if #NEAR/3 (b c) doesn't occur in the collection?**

- Its ctf == 0
  - … so $p_{MLE}(t|C) == 0$ (i.e., no smoothing weight)
  - … so #AND returns 0 for all documents

$$p_{and}(q \mid d) = \prod_{q_i \in q} p(q_i \mid d)^{\frac{1}{|q|}}$$

- This behavior is exact match, not best match
  - Indri is a best match model

8 © 2021 Jamie Callan

## Indri Implementation

**Query:** #AND (a #NEAR/3 (b c) )

**QrySop<u>Score</u>.getDefaultScore (RetrievalModel r, long docid)**

**What happens if #near/3 (b c) doesn't occur in the collection?**

**Solution:** <u>Extra smoothing</u> for terms that have ctf = 0

If r == RetrievalModel.Indri

If ctf ($t$) = 0

calculate $p_{MLE}(t|C)$ using ctf ($t$) = 0.5 } **Undocumented behavior (ctf < 1)**

$$p_{scoreDefault}(t \mid \text{docid}) = (1-\lambda)\frac{0 + \mu\, p_{MLE}(t \mid C)}{length(\text{docid}) + \mu} + \lambda\, p_{MLE}(t \mid C)$$

## Indri Implementation

**Query:** #OR (a #AND (b c) ) **Document:** a

**QrySop<u>And</u>.getDefaultScore (RetrievalModel r, long docid)**

- The standard Indri <u>AND</u> calculation done on the <u>default score</u> of each argument

If r == RetrievalModel.Indri

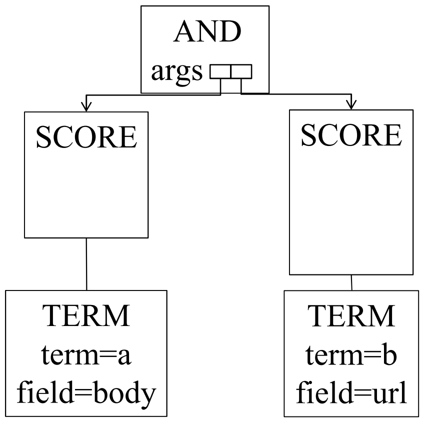
$$p_{andDefault}(q \mid d) = \prod_{q_i \in q} \boxed{p_{q_i default}}(q_i \mid d)^{\frac{1}{|q|}}$$

**This is the only difference.**
**Call the ith query argument's getDefaultScore method.**

## QryEval Example

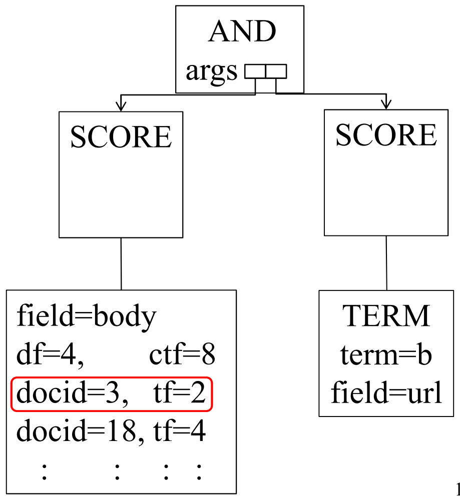
11

## The initial query: #AND (a.body b.url)



12

## Query Initialization



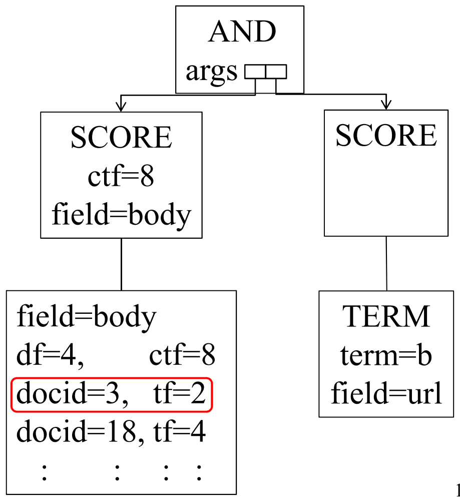
**AND operator initialization**

AND initializes its first arg.

SCORE initializes its arg.

The result is an inverted list.

## Query Initialization



**AND operator initialization**

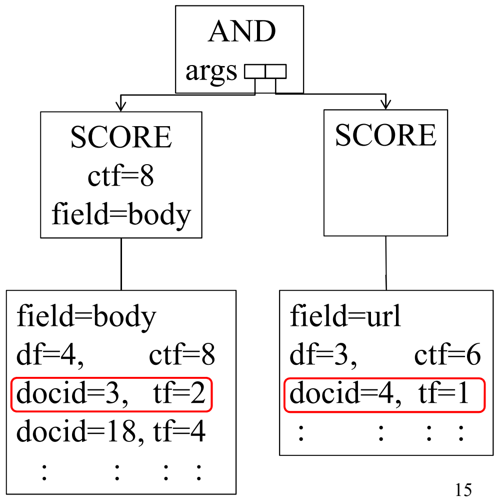
AND initializes its first arg.

SCORE initializes its arg.

The result is an inverted list.

The SCORE operator caches information that it will need later.

## Query Initialization



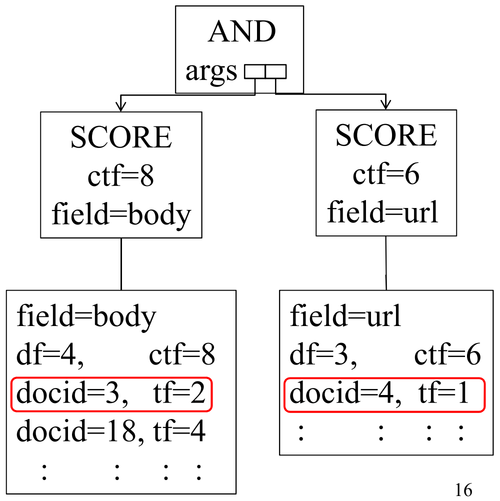
**AND operator initialization**

AND initializes its second arg.

SCORE initializes its arg.

The result is an inverted list.

## Query Initialization



**AND operator initialization**

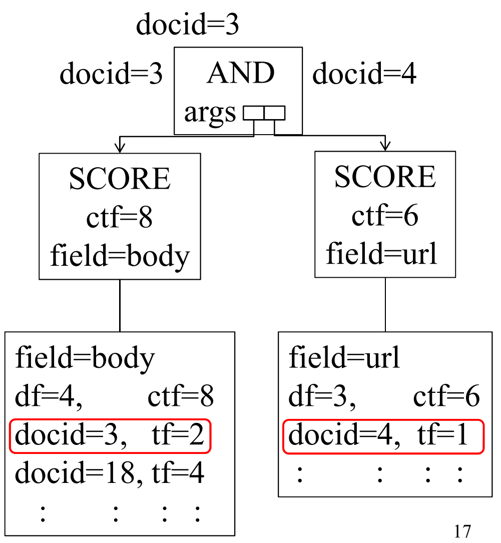
AND initializes its second arg.

SCORE initializes its arg.

The result is an inverted list.

The SCORE operator caches information that it will need later.

## Call to docIteratorHasMatch & getScore (First Time)

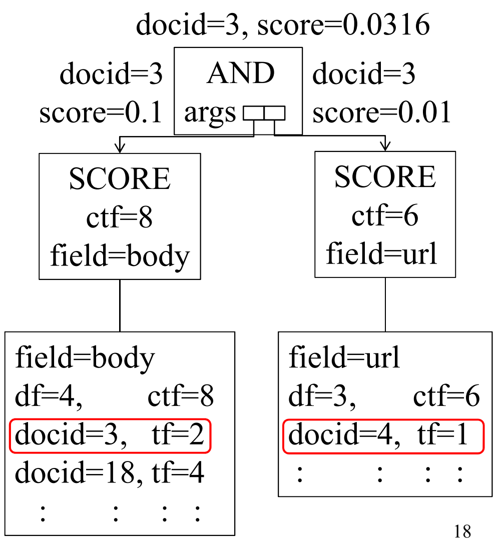


**AND Operator Evaluation**

Min document is 3.

HasMatch caches the docid (3) and returns True

## Call to docIteratorHasMatch & getScore (First Time)



**AND Operator Evaluation**

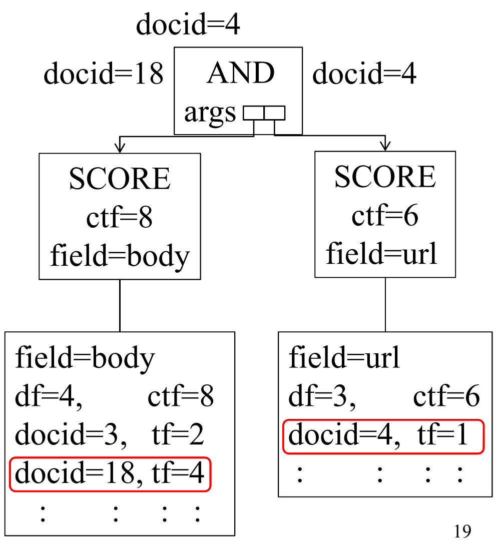
getScore uses the cached docid (3).

args[0] has match, so call
  args[0].getScore ().
Suppose the result is 0.1.

args[1] does not match, so call
  args[1].getDefaultScore(3).
Suppose the result is 0.01.

$\text{Score}_{\text{AND}}(3) = (0.1^{0.5} \times 0.01^{0.5})$

## Call to docIteratorHasMatch & getScore (Second Time)

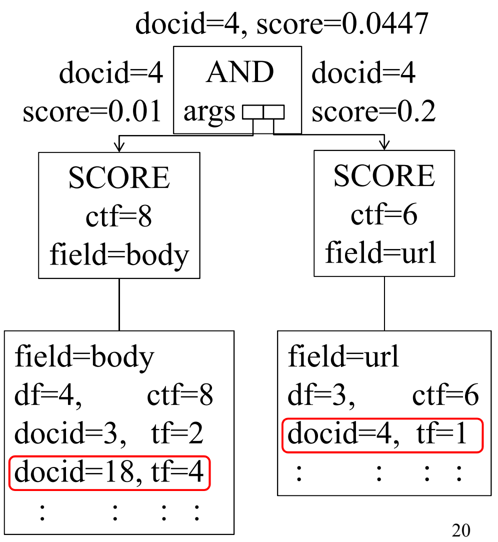


**AND Operator Evaluation**

Min document is 4.

HasMatch caches the docid (4) and returns True

## Call to docIteratorHasMatch & getScore (Second Time)



**AND Operator Evaluation**

getScore uses the cached docid (4).

args[0] does not match, so call args[0].getDefaultScore (4). Suppose the result is 0.01.

args[1] has match, so call args[1].getScore(). Suppose the result is 0.2.

$\text{Score}_{\text{AND}}(4) = (0.01^{0.5} \times 0.2^{0.5})$

## Default Belief Scores Are Only a Small Complication

**When evaluating a query argument**

- If it has a match for the current document
  - Ask the query argument to calculate the document score for the current document
  - Else ask the query argument to calculate a default score for the current document
    - » E.g., a SCORE operator (the example given)
    - » E.g., an OR operator (similar logic)

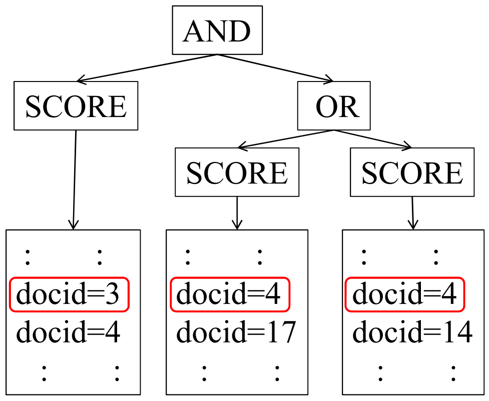
## Default Belief Scores Are Only a Small Complication

**Which types of query operators calculate default scores?**

- If an operator calculates scores

  … it also calculates default scores
- QrySop operators calculate default scores
- QryIop operators do not calculate default scores

## Call to docIteratorHasMatch & getScore (First Time)



**AND Operator Evaluation**

Min document is 3.

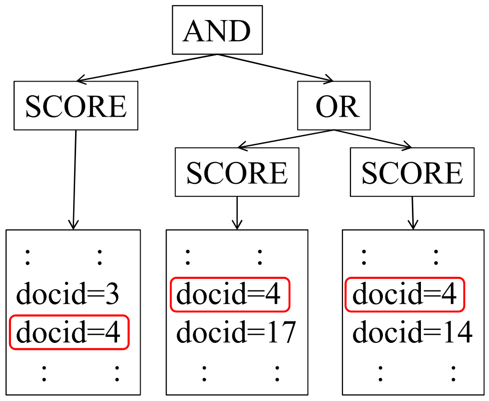
args[0] has match, so call args[0].getScore (). Suppose the result is 0.3.

args[1] does not match, so call args[1].getDefaultScore(3).

OR calls getDefaultScore(3) for all of its args and computes a score. Suppose it is 0.01.

$\text{Score}_{\text{AND}}(3) = (0.3^{0.5} \times 0.01^{0.5})$

## Call to docIteratorHasMatch & getScore (Second Time)



**AND Operator Evaluation**

Min document is 4.

args[0] matches, so call args[0].getScore (). Suppose the result is 0.3.

args[1] matches, so call args[1].getScore(). Suppose the score is 0.2.

$\text{Score}_{\text{AND}}(4) = (0.3^{0.5} \times 0.2^{0.5})$

## Using Default Belief Scores Properly Requires Two Components

**Add a new method to all QrySop operators**

double getDefaultScore (RetrievalModel r, long docid)

– QrySopScore.getDefaultScore calculates a score for a term

– QrySop <other>. getDefaultScore combines scores for *n* terms

**When any QrySop operator calculates scores**

If the ith query argument contains document d

then read its score from the ith score list

else call the ith query argument's getDefaultScore method

**It may sound complicated now, but actually it is very easy**

## Outline

**HW2 implementation**

- Indri default beliefs
- Window operator

## Proximity Operators: The Window (or Unordered Window) Operator

**The WINDOW/n operator is used to match related concepts**

- Arguments can be in any order
- n specifies the maximum distance between any pair of terms

**Examples**

- WINDOW/100 (obama merkel putin)
  - We don't care which order they occur in

**Typically proximity operators have complexity O(|C|)**

- A single pass down each inverted list, similar to NEAR/n

27

## Proximity Operators: The Window (or Unordered Window) Operator

| **a** | | **b** | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | … | |

Query: # WINDOW/20 (a b)

28

## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | … | |

Query: # WINDOW/20 (a b)

Initialize doc iterators

## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | … | |

Query: # WINDOW/20 (a b)

Advance all doc iterators until they point to the same document

- This is a simple nested loop

## Proximity Operators: The Window (or Unordered Window) Operator

| a | b |
|---|---|
| df: 47 | df: 95 |
| doc: 19 | doc: 23 |
| tf: 1 | tf: 1 |
| locs: 7 | locs: 99 |
| doc: 27 | doc: 27 |
| tf: 3 | tf: 4 |
| locs: 47 | locs: 48 |
| 98 | 49 |
| 132 | 133 |
| doc: 92 | 134 |
| … | doc: 148 |
| | ... |

Query: # WINDOW/20 (a b)

Same document
Initialize location iterators

## Proximity Operators: The Window (or Unordered Window) Operator

| a | b |
|---|---|
| df: 47 | df: 95 |
| doc: 19 | doc: 23 |
| tf: 1 | tf: 1 |
| locs: 7 | locs: 99 |
| doc: 27 | doc: 27 |
| tf: 3 | tf: 4 |
| locs: 47 | locs: 48 |
| 98 | 49 |
| 132 | 133 |
| doc: 92 | 134 |
| … | doc: 148 |
| | ... |

Query: # WINDOW/20 (a b)

Find the min (47) and
max (48) locations

## Proximity Operators: The Window (or Unordered Window) Operator

| **a** | | **b** | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Is (max – min) < window?

48 – 47 < 20 (match)

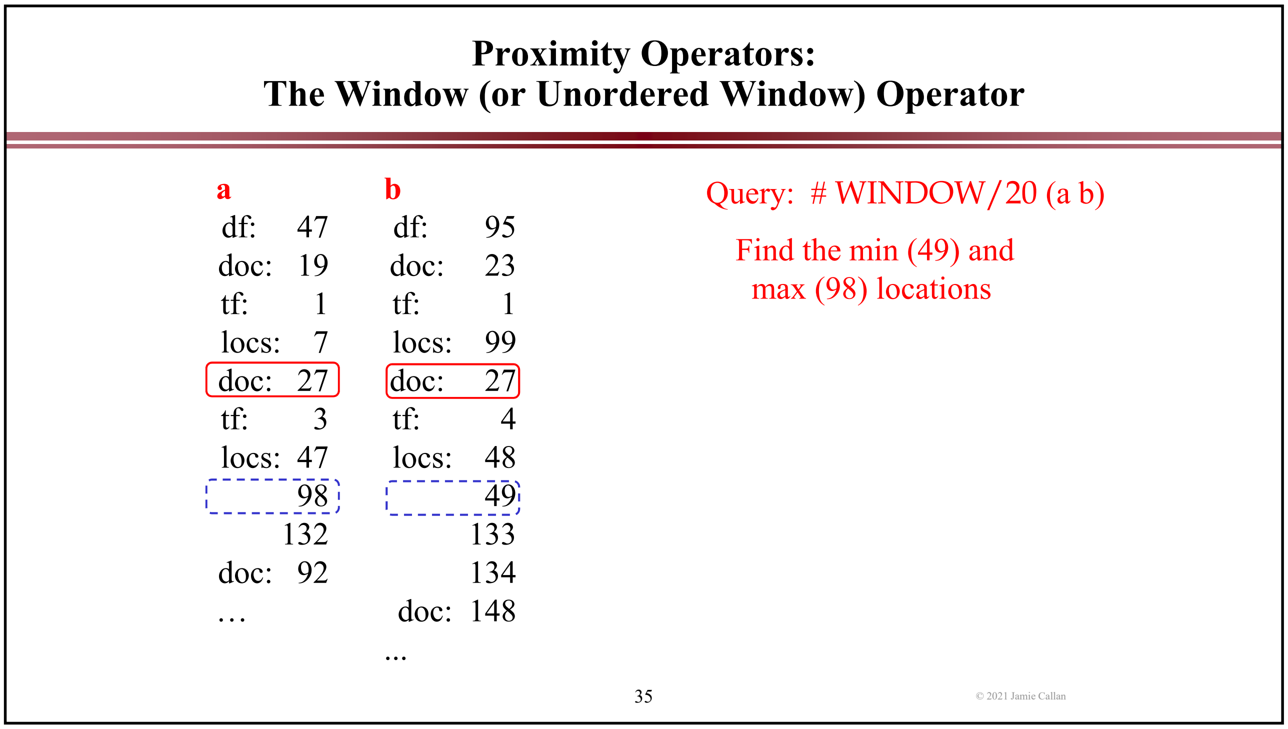
Record match
- max location (48)

## Proximity Operators: The Window (or Unordered Window) Operator

| **a** | | **b** | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Increment all loc iterators

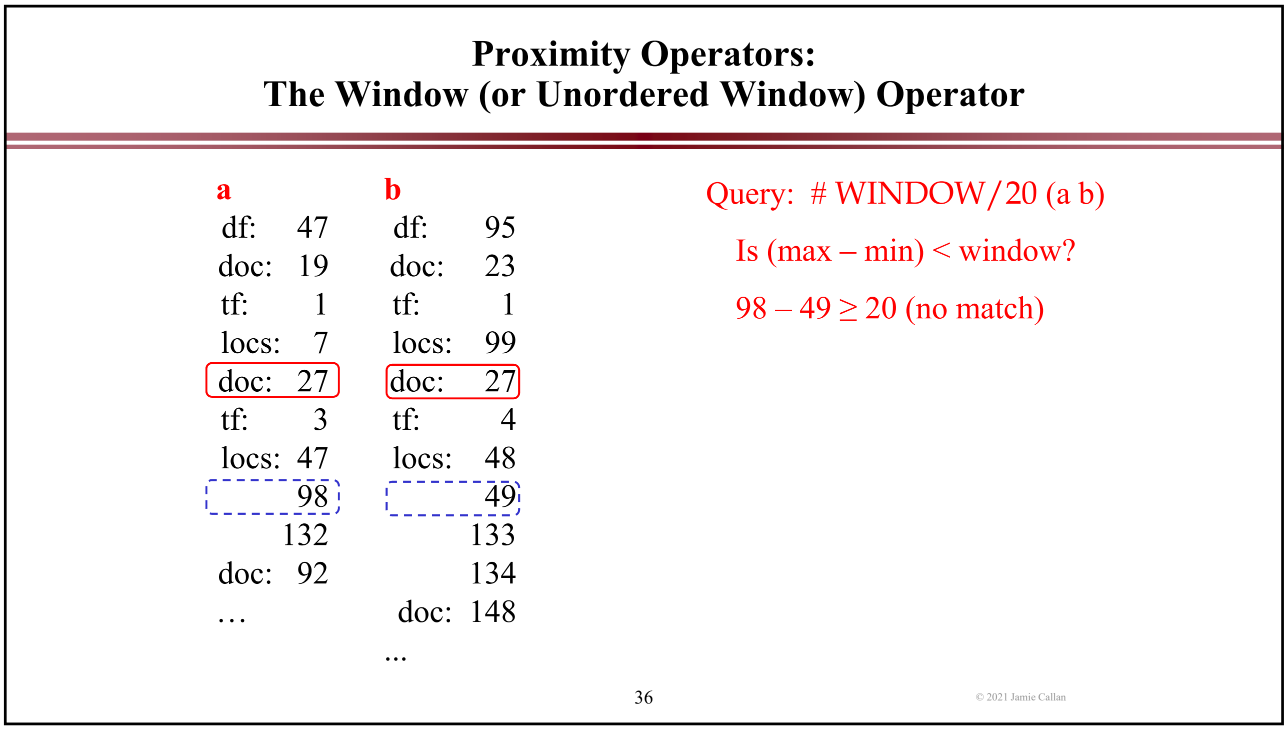


## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Find the min (49) and max (98) locations



## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Is (max – min) < window?

$98 - 49 \geq 20$ (no match)

## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

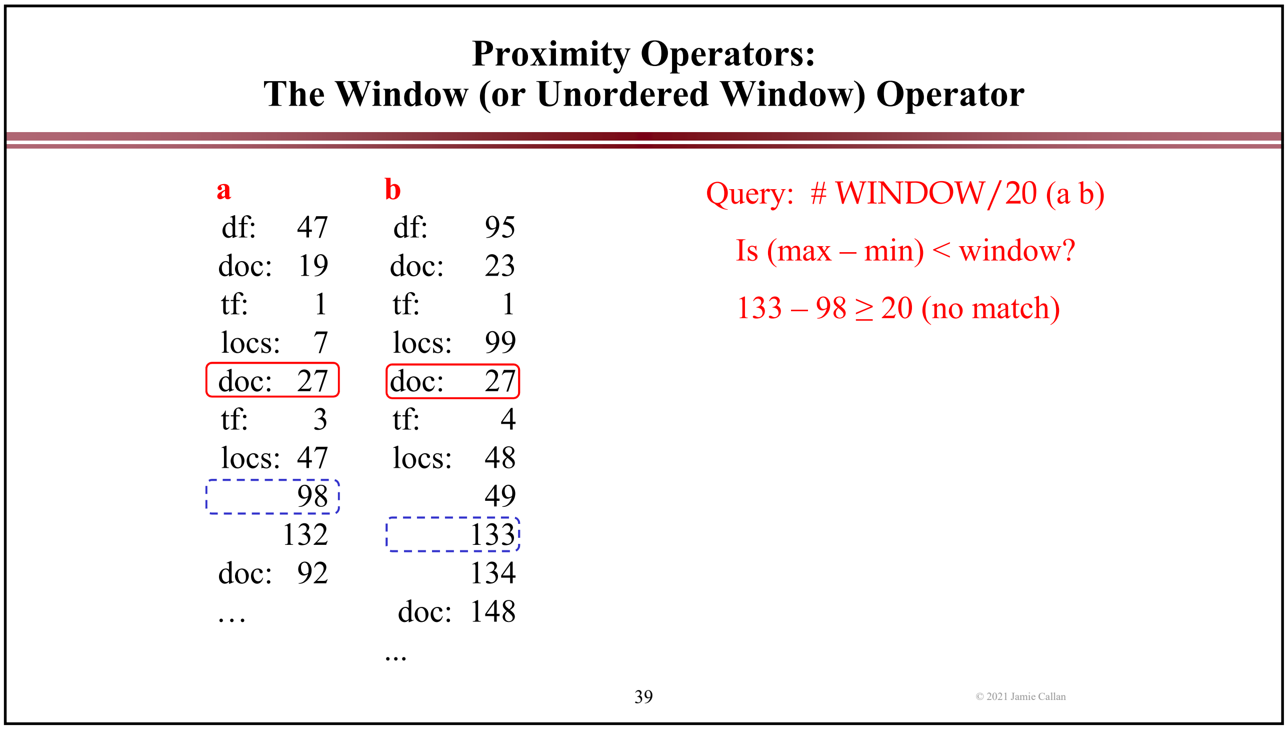
Increment the iterator for the min location

## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Find the min (98) and max (133) locations



## Proximity Operators: The Window (or Unordered Window) Operator

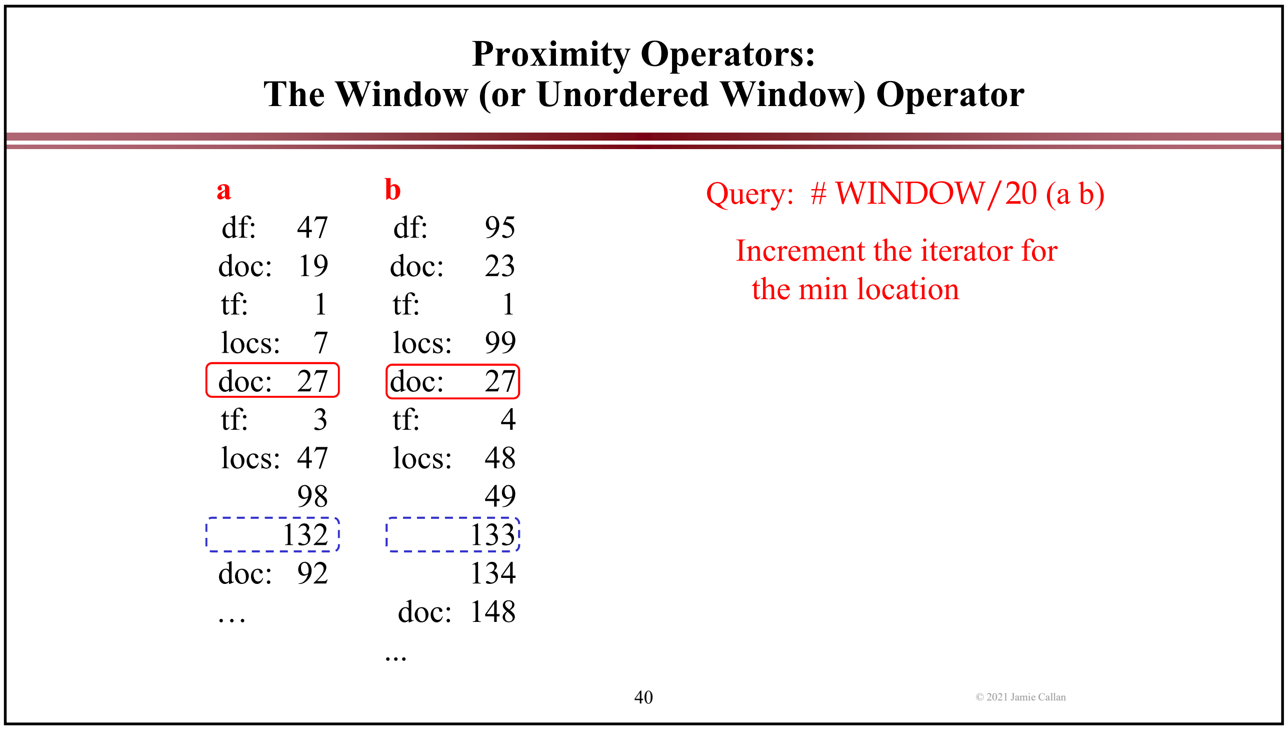
| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Is (max – min) < window?

$133 - 98 \geq 20$ (no match)

39



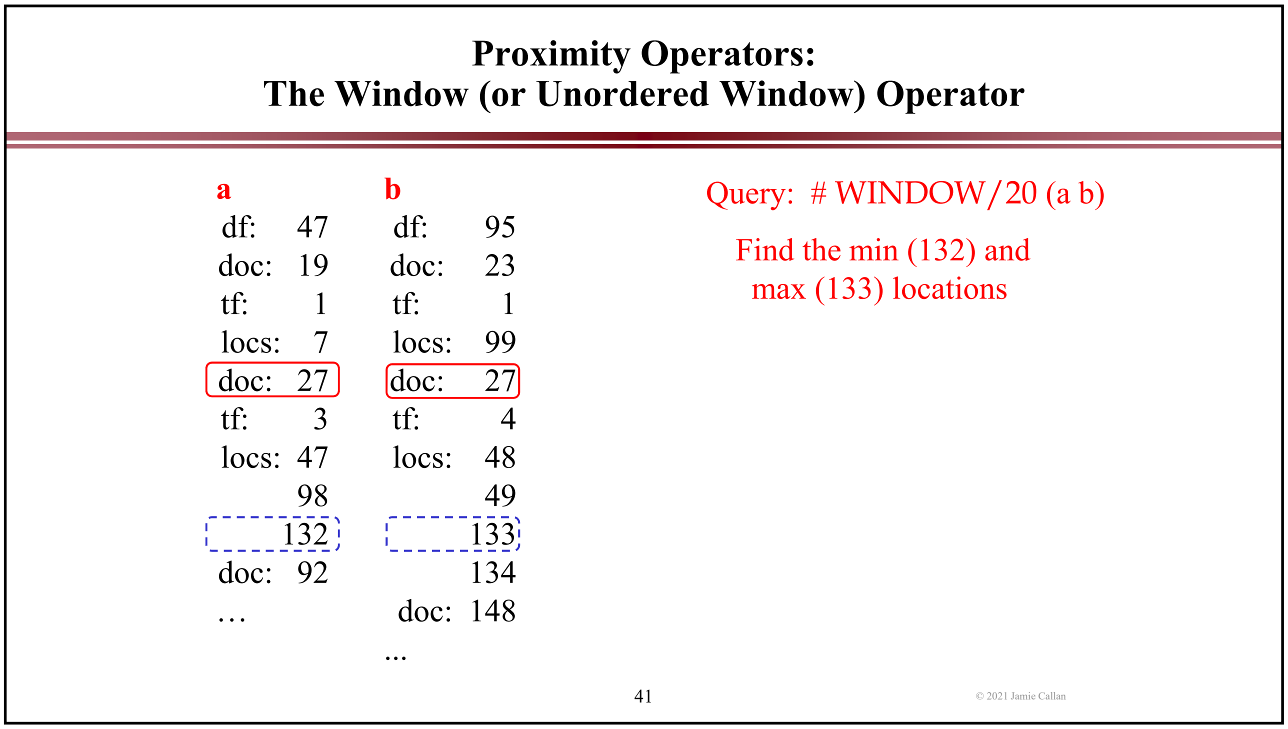
## Proximity Operators: The Window (or Unordered Window) Operator

| a | | b | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| doc: | 27 | doc: | 27 |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | 132 | | 133 |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Increment the iterator for the min location

40



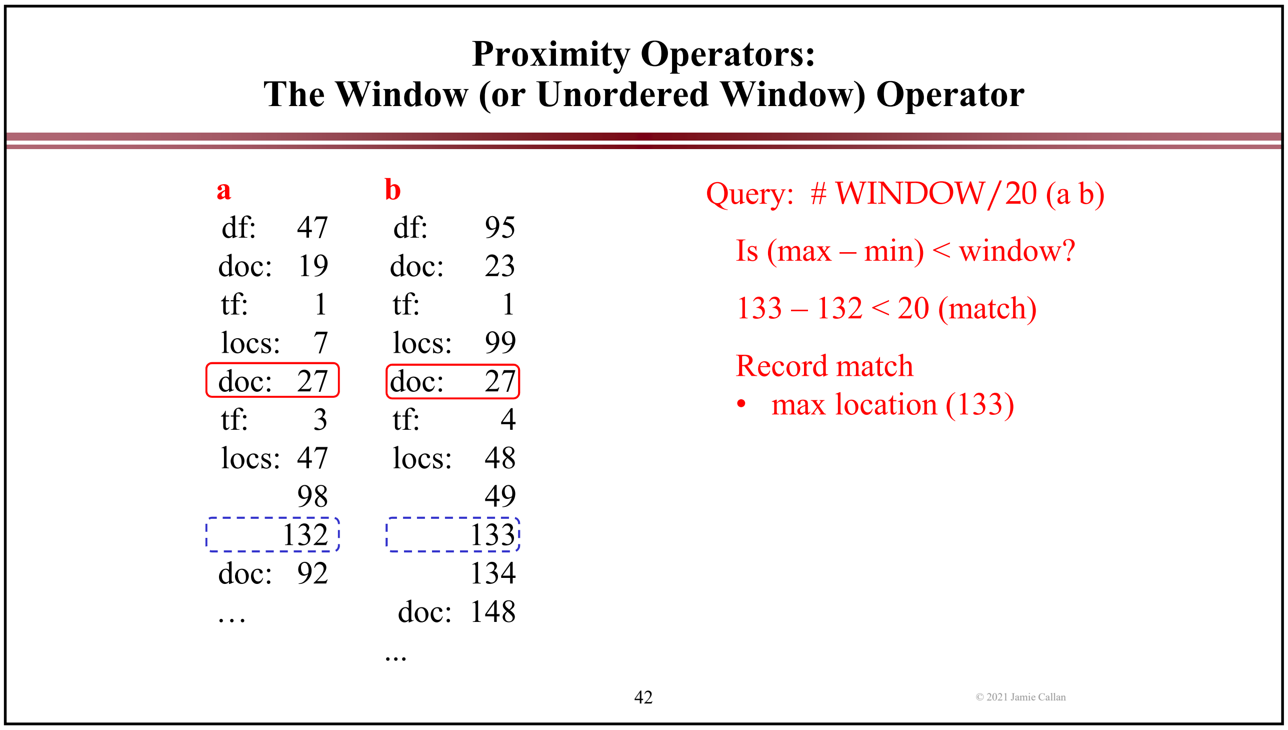
## Proximity Operators: The Window (or Unordered Window) Operator

| **a** | | **b** | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| $\boxed{\text{doc:}\ 27}$ | | $\boxed{\text{doc:}\ 27}$ | |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | $\boxed{132}$ | | $\boxed{133}$ |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

Query: # WINDOW/20 (a b)

Find the min (132) and max (133) locations

41 © 2021 Jamie Callan



## Proximity Operators: The Window (or Unordered Window) Operator

| **a** | | **b** | |
|---|---|---|---|
| df: | 47 | df: | 95 |
| doc: | 19 | doc: | 23 |
| tf: | 1 | tf: | 1 |
| locs: | 7 | locs: | 99 |
| $\boxed{\text{doc:}\ 27}$ | | $\boxed{\text{doc:}\ 27}$ | |
| tf: | 3 | tf: | 4 |
| locs: | 47 | locs: | 48 |
| | 98 | | 49 |
| | $\boxed{132}$ | | $\boxed{133}$ |
| doc: | 92 | | 134 |
| … | | doc: | 148 |
| | | ... | |

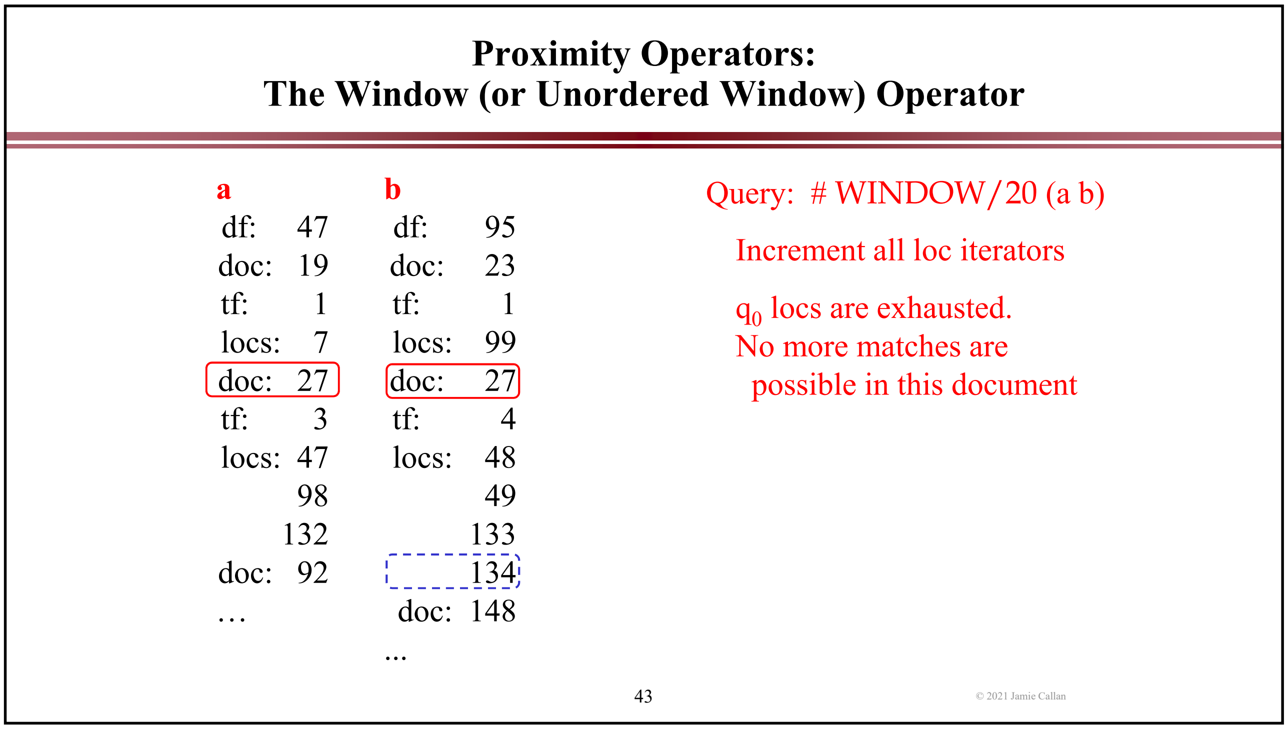
Query: # WINDOW/20 (a b)

Is (max – min) < window?

133 – 132 < 20 (match)

Record match
- max location (133)

42 © 2021 Jamie Callan



## Proximity Operators: The Window (or Unordered Window) Operator

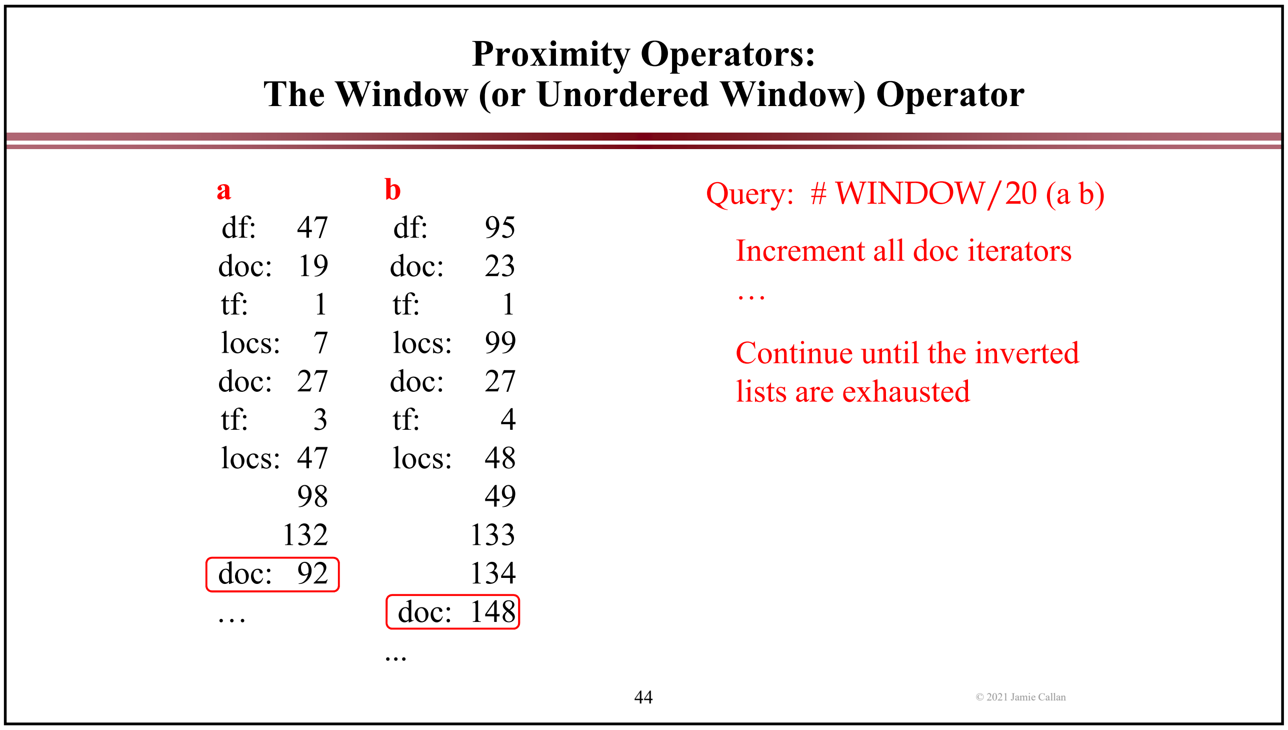
| a | b |
|---|---|
| df: 47 | df: 95 |
| doc: 19 | doc: 23 |
| tf: 1 | tf: 1 |
| locs: 7 | locs: 99 |
| $\boxed{\text{doc: 27}}$ | $\boxed{\text{doc: 27}}$ |
| tf: 3 | tf: 4 |
| locs: 47 | locs: 48 |
| 98 | 49 |
| 132 | 133 |
| doc: 92 | $\boxed{134}$ |
| … | doc: 148 |
| | ... |

Query: # WINDOW/20 (a b)

Increment all loc iterators

$q_0$ locs are exhausted.
No more matches are possible in this document

43



## Proximity Operators: The Window (or Unordered Window) Operator

| a | b |
|---|---|
| df: 47 | df: 95 |
| doc: 19 | doc: 23 |
| tf: 1 | tf: 1 |
| locs: 7 | locs: 99 |
| doc: 27 | doc: 27 |
| tf: 3 | tf: 4 |
| locs: 47 | locs: 48 |
| 98 | 49 |
| 132 | 133 |
| $\boxed{\text{doc: 92}}$ | 134 |
| … | $\boxed{\text{doc: 148}}$ |
| | ... |

Query: # WINDOW/20 (a b)

Increment all doc iterators
…

Continue until the inverted lists are exhausted

44

## Proximity Operators: The Window (or Unordered Window) Operator

**Implementation note**

- A document term can only match the query once
- **Query:** #WINDOW/100 (obama merkel putin)
- **Document**: obama … merkel … putin … merkel … obama
- There is just <u>one</u> match here

**One can imagine other implementations, but this is the norm**

- Usually more complicated matching doesn't improve accuracy