

# Efficient Document Retrieval in Main Memory

---

Trevor Strohman • W. Bruce Croft  
University of Massachusetts Amherst

**What if we put  
everything in memory?**

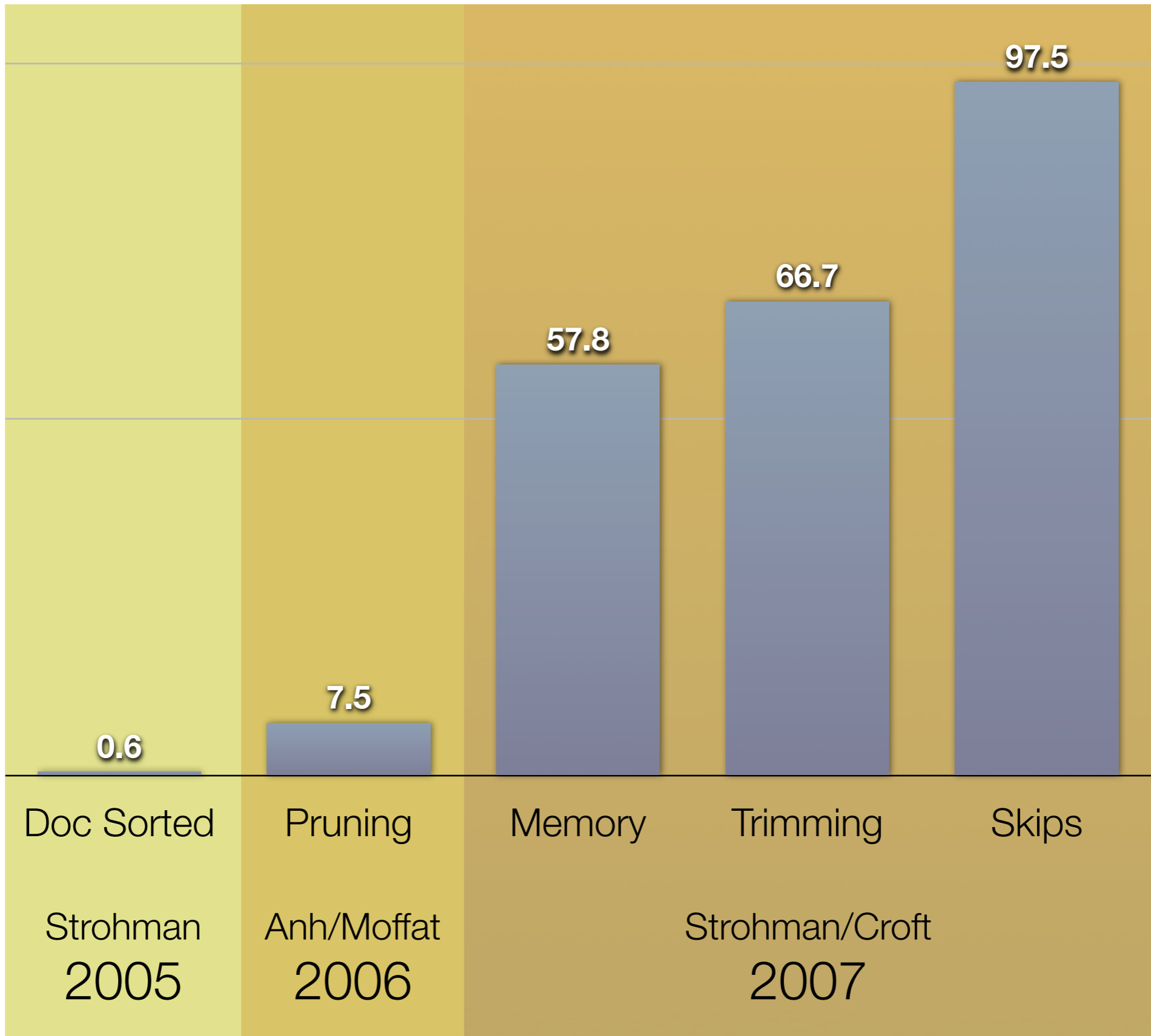
**What if we put  
everything in memory?**

**Does in-memory processing  
allow for new efficient  
retrieval techniques?**

Results

GOV2 queries/second

100  
50  
0



Doc Sorted

Pruning

Memory

Trimming

Skips

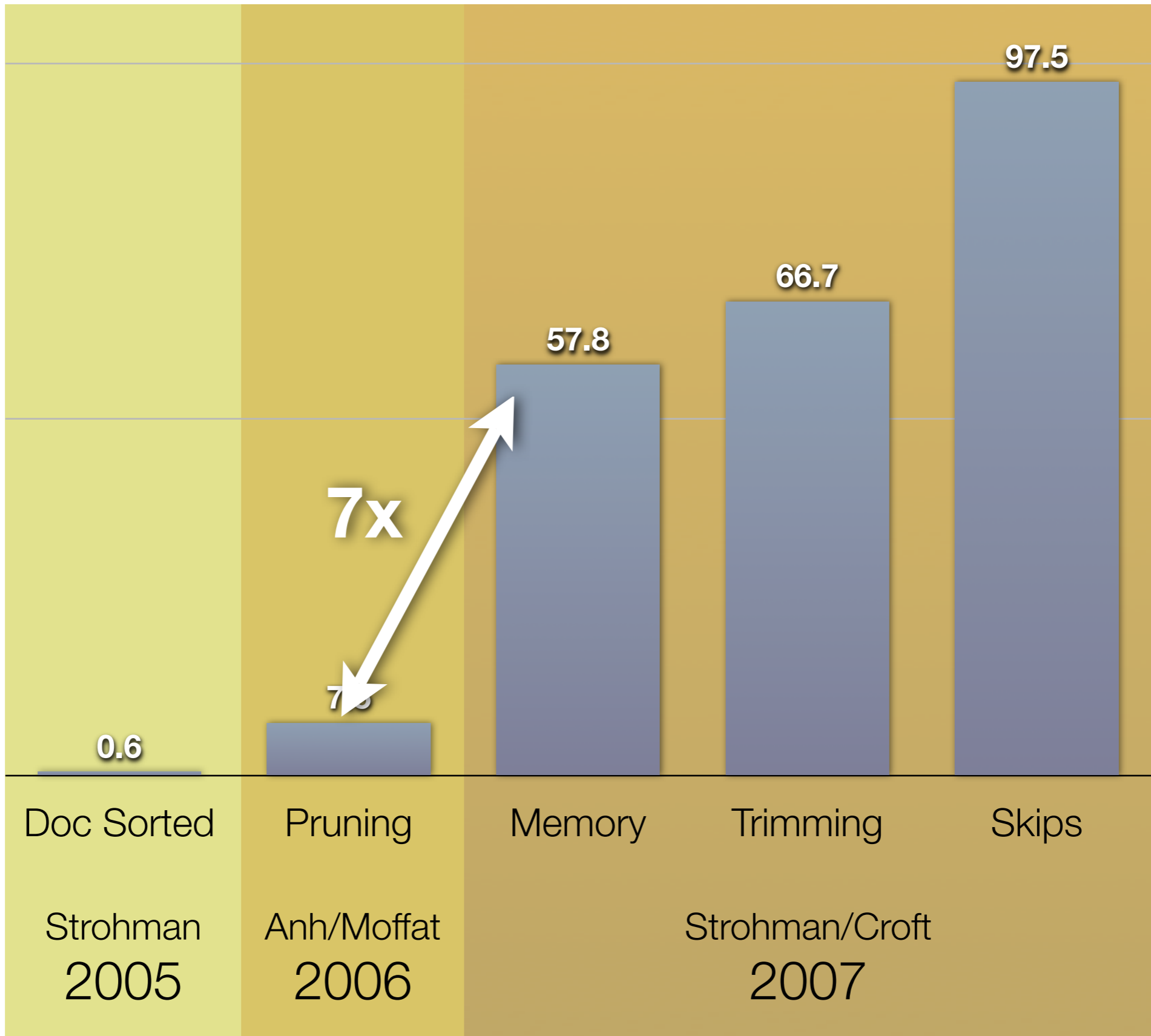
Strohman  
2005

Anh/Moffat  
2006

Strohman/Croft  
2007

GOV2 queries/second

100  
50  
0



Doc Sorted

Pruning

Memory

Trimming

Skips

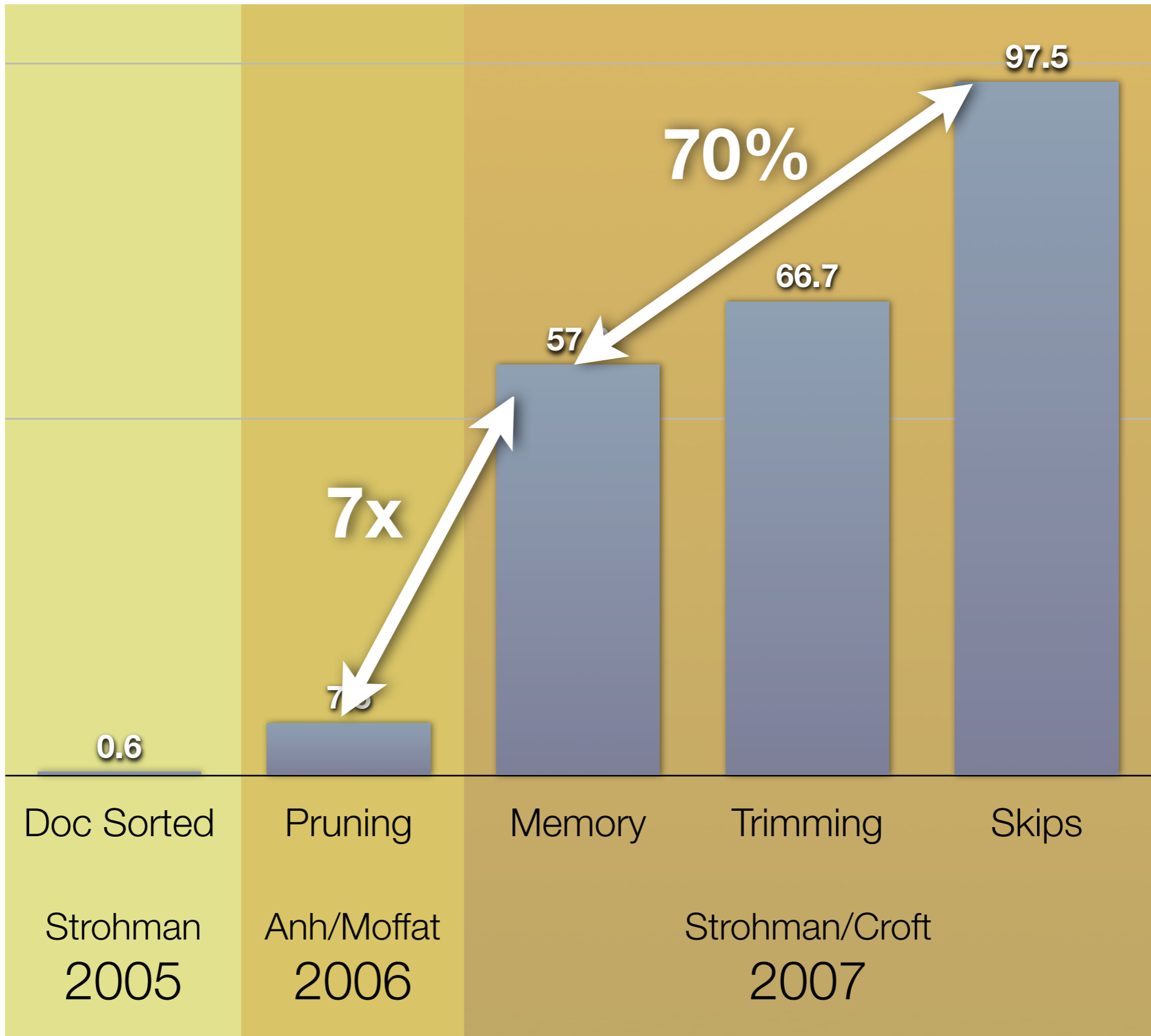
Strohman  
2005

Anh/Moffat  
2006

Strohman/Croft  
2007

GOV2 queries/second

100  
50  
0



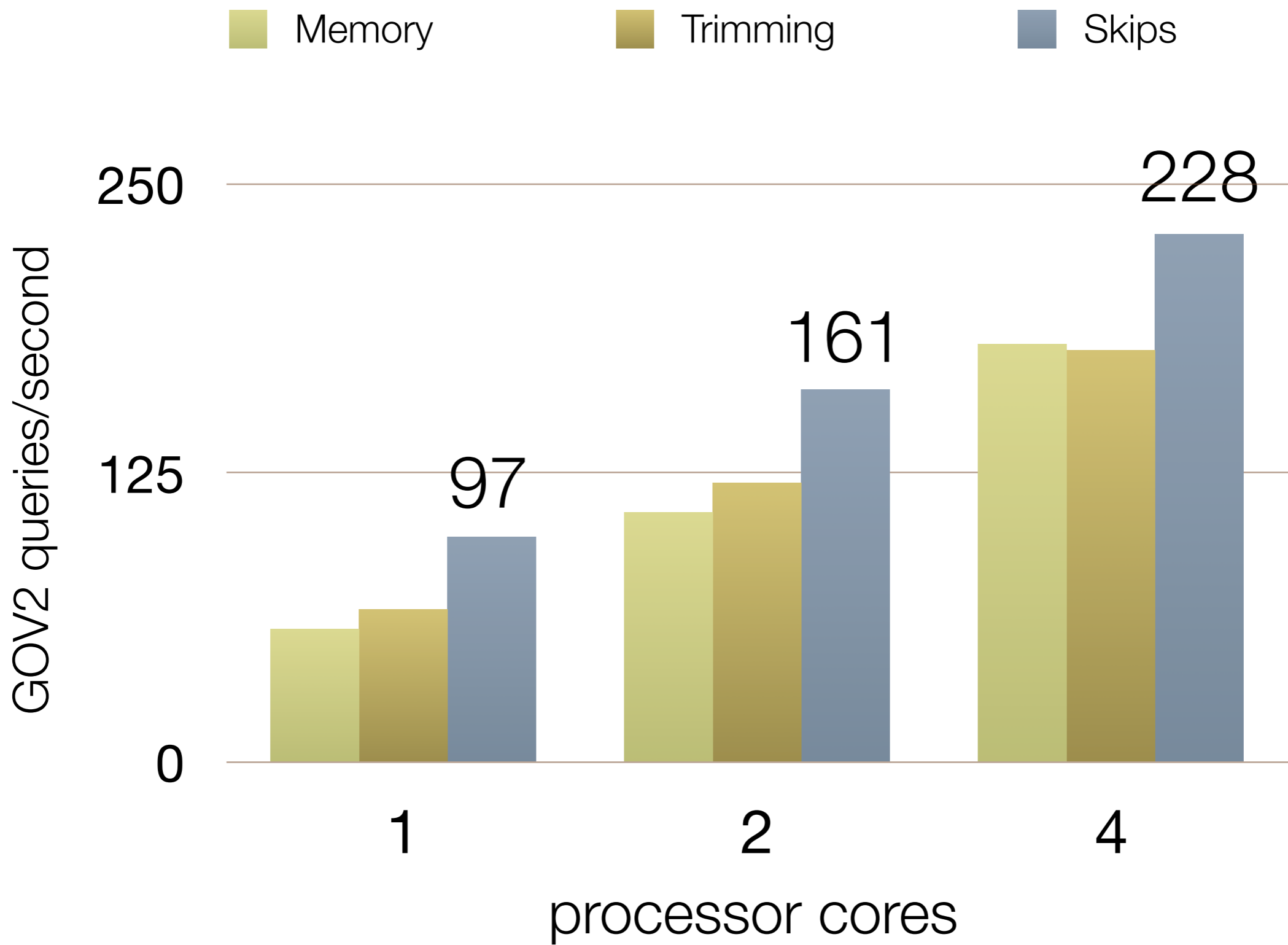
Doc Sorted  
Strohman  
2005

Pruning  
Anh/Moffat  
2006

Memory

Trimming  
Strohman/Croft  
2007

Skips



# Exact results only

no static pruning

Büettcher, Carmel

no approximate pruning

Lester: Adaptive Pruning

Moffat/Zobel: Quit and Continue

Components

**Everything in memory**

**Efficient dynamic pruning**

# Everything in memory

index is memory mapped

8GB of RAM, 7GB index

# Efficient dynamic pruning

## Everything in memory

index is memory mapped

8GB of RAM, 7GB index

## Efficient dynamic pruning

accumulator trimming

inverted list skipping

# Dynamic Pruning

weights

**8**

1

4

6

**7**

5

7

**6**

2

33

**5**

20

29

32

**4**

9

11

**3**

15

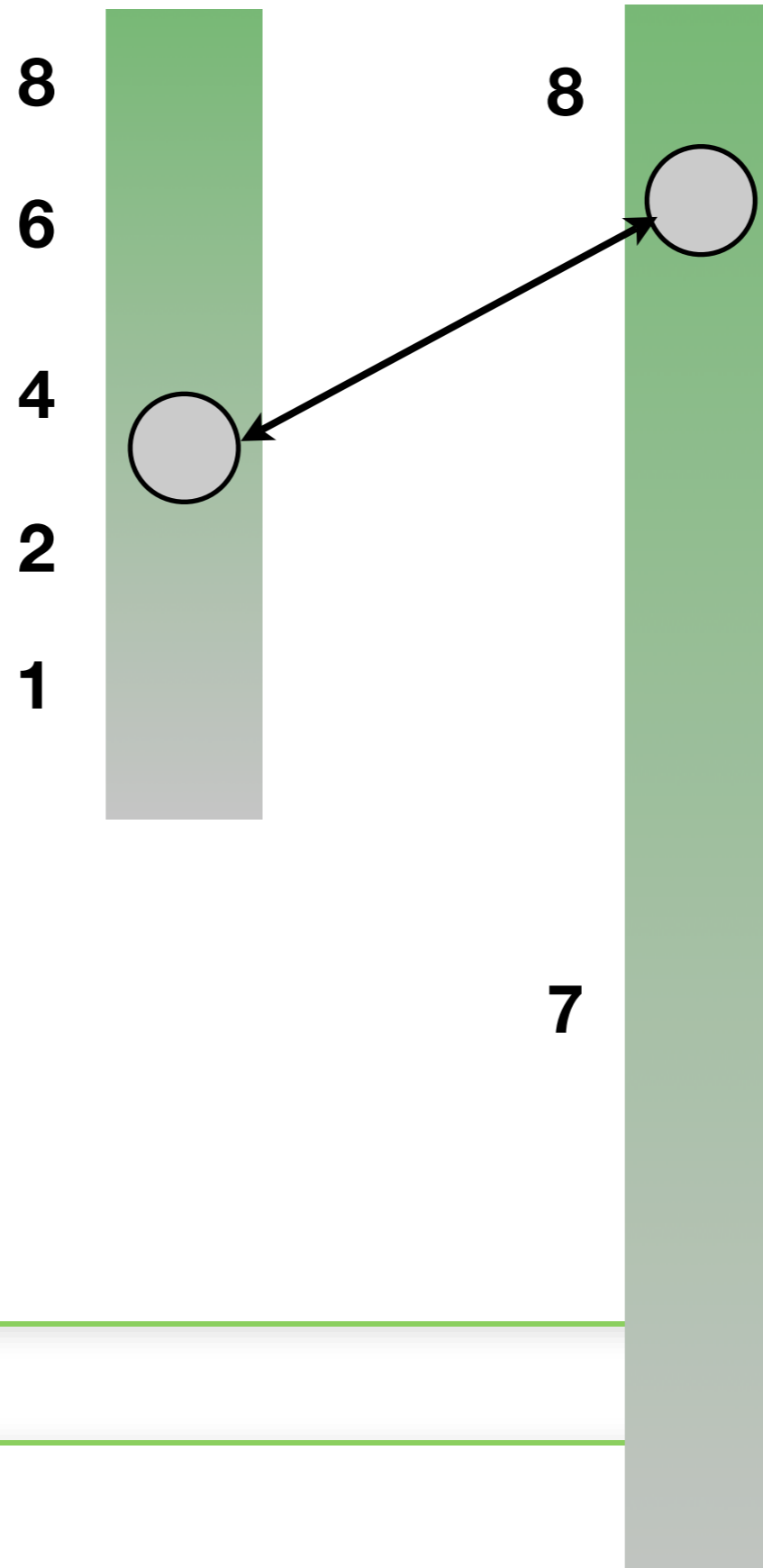
19


21

documents

stellwagen  
345,000

bank  
45.8 million



stellwagen bank 

# accumulator



maximum possible score  
for **any** document

maximum possible score  
for **this** document

**uncertain region**

minimum possible score  
for **this** document

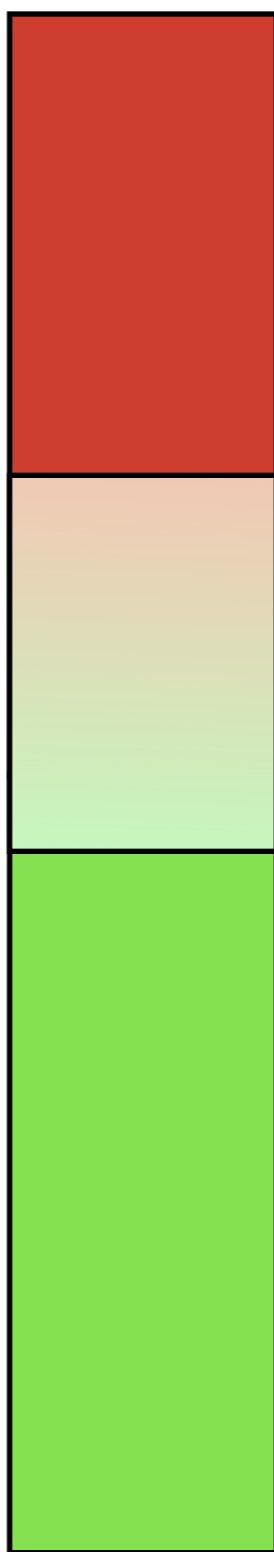
minimum possible score  
for **any** document

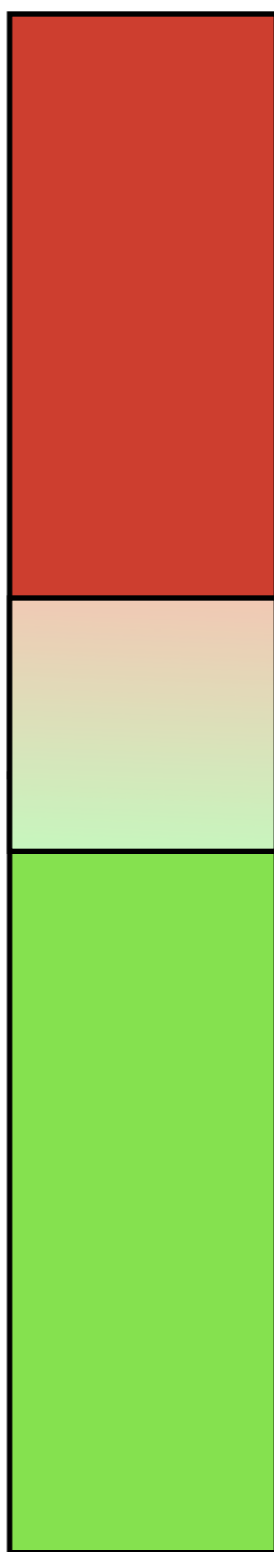




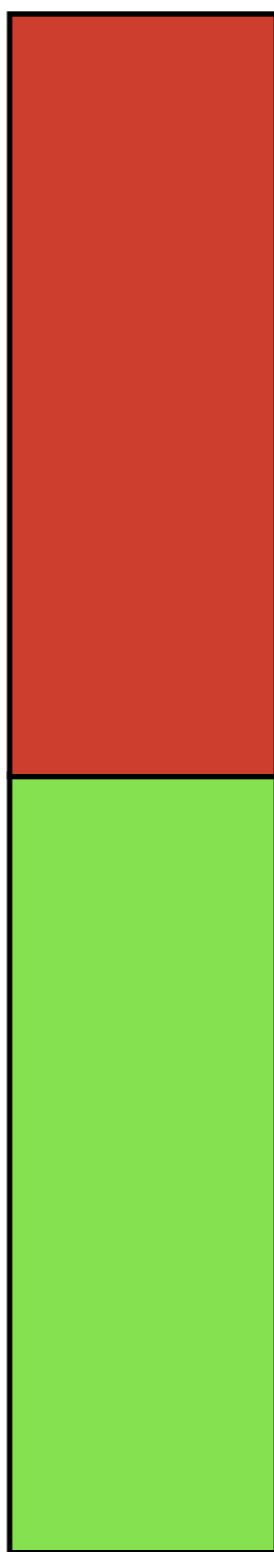


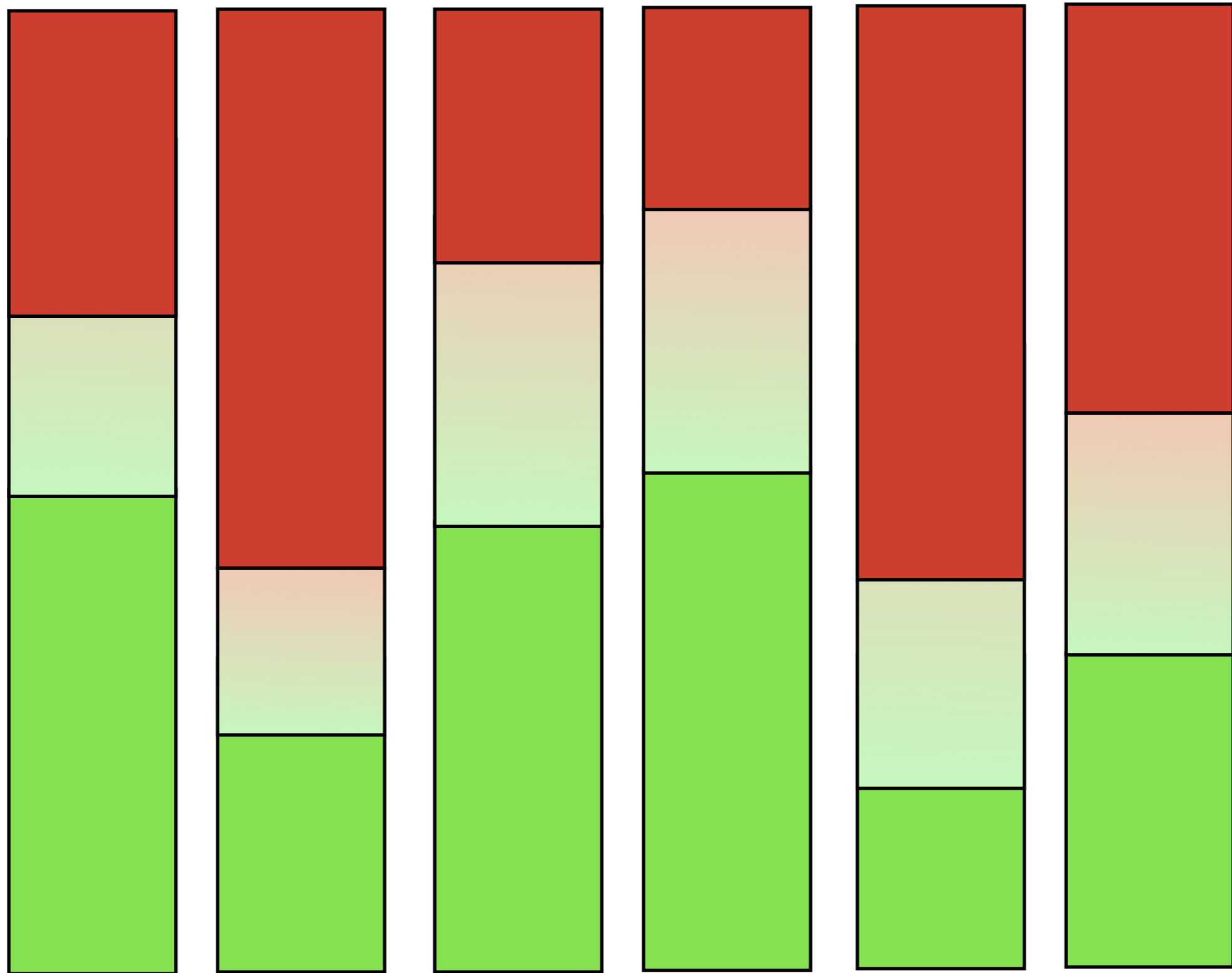




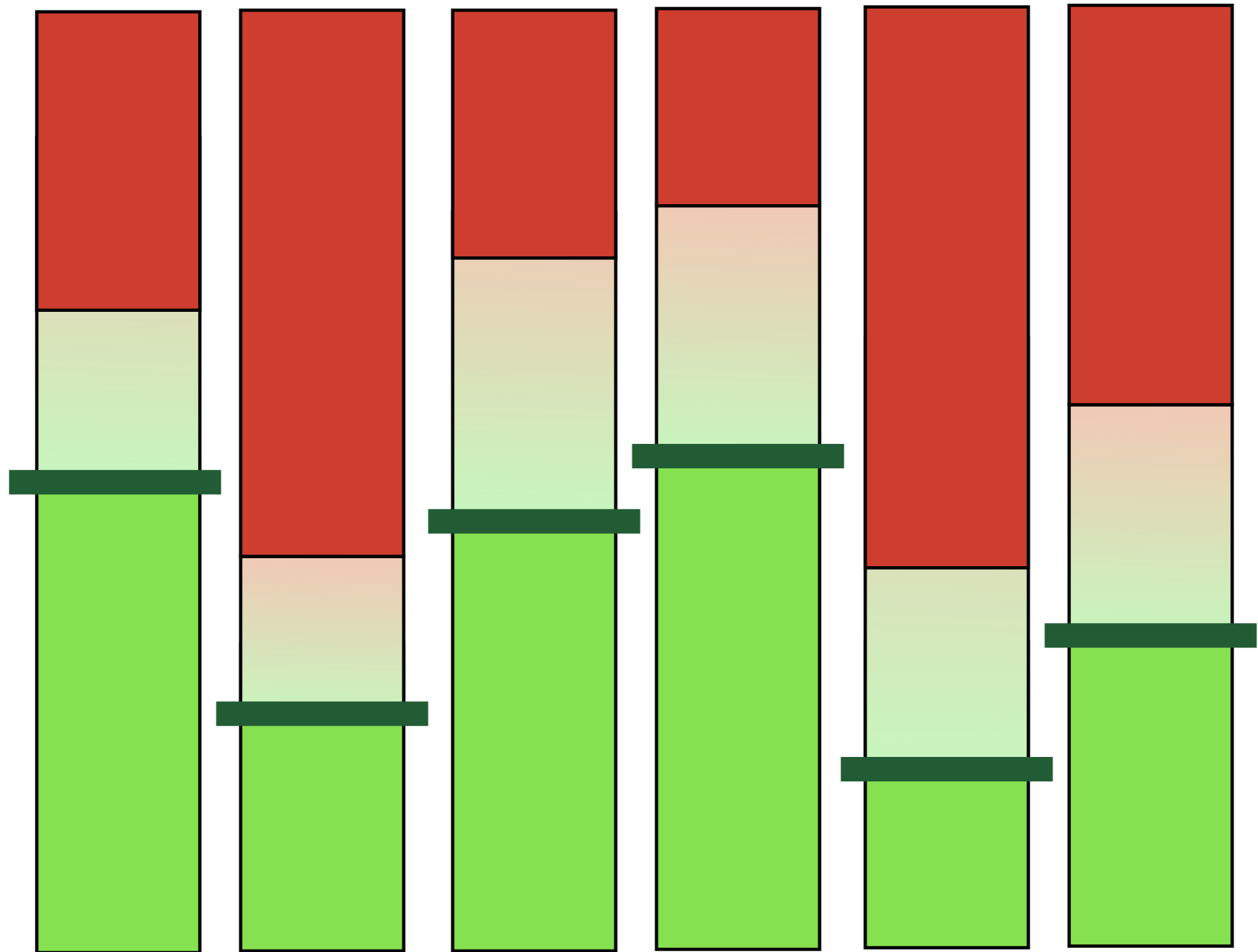




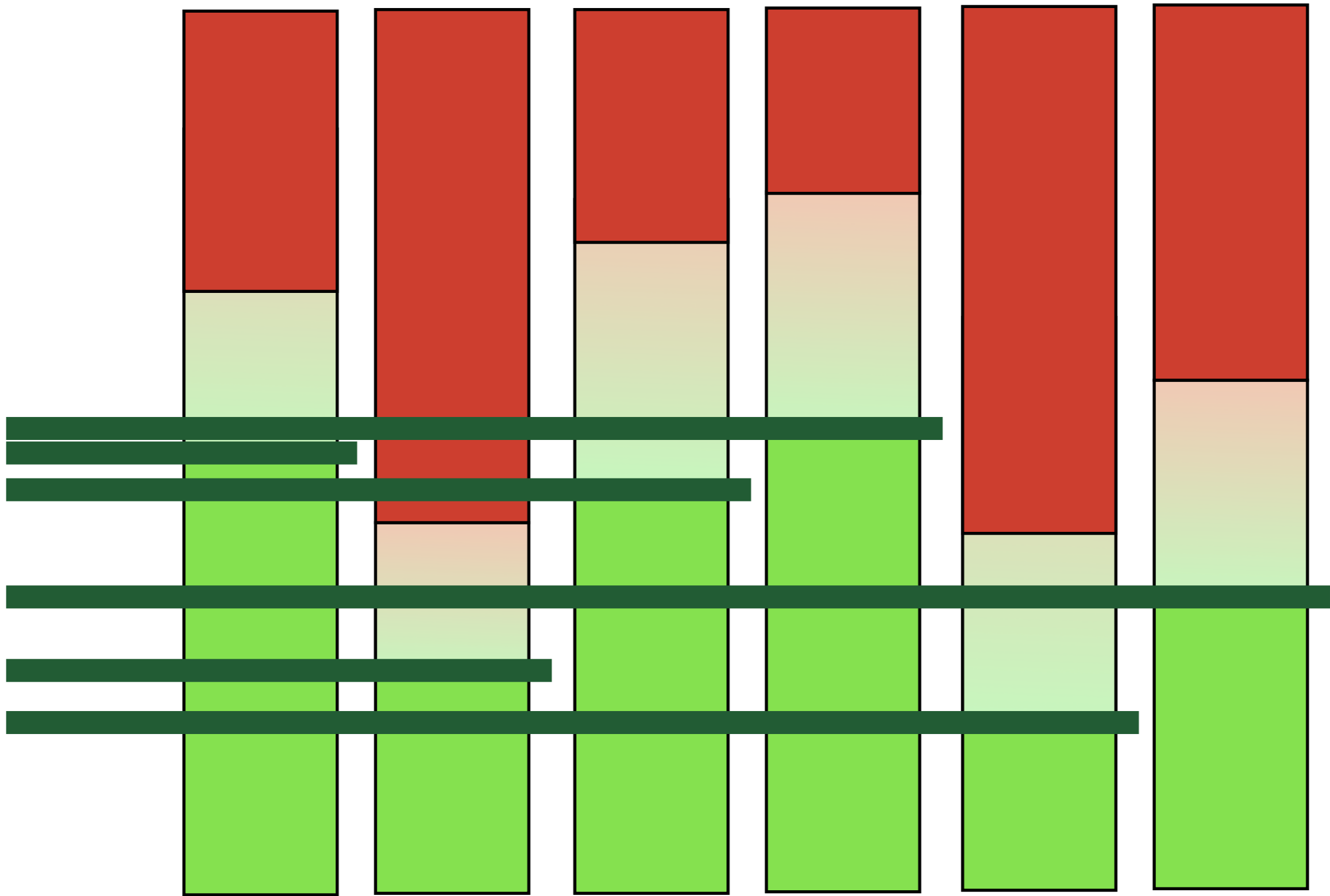




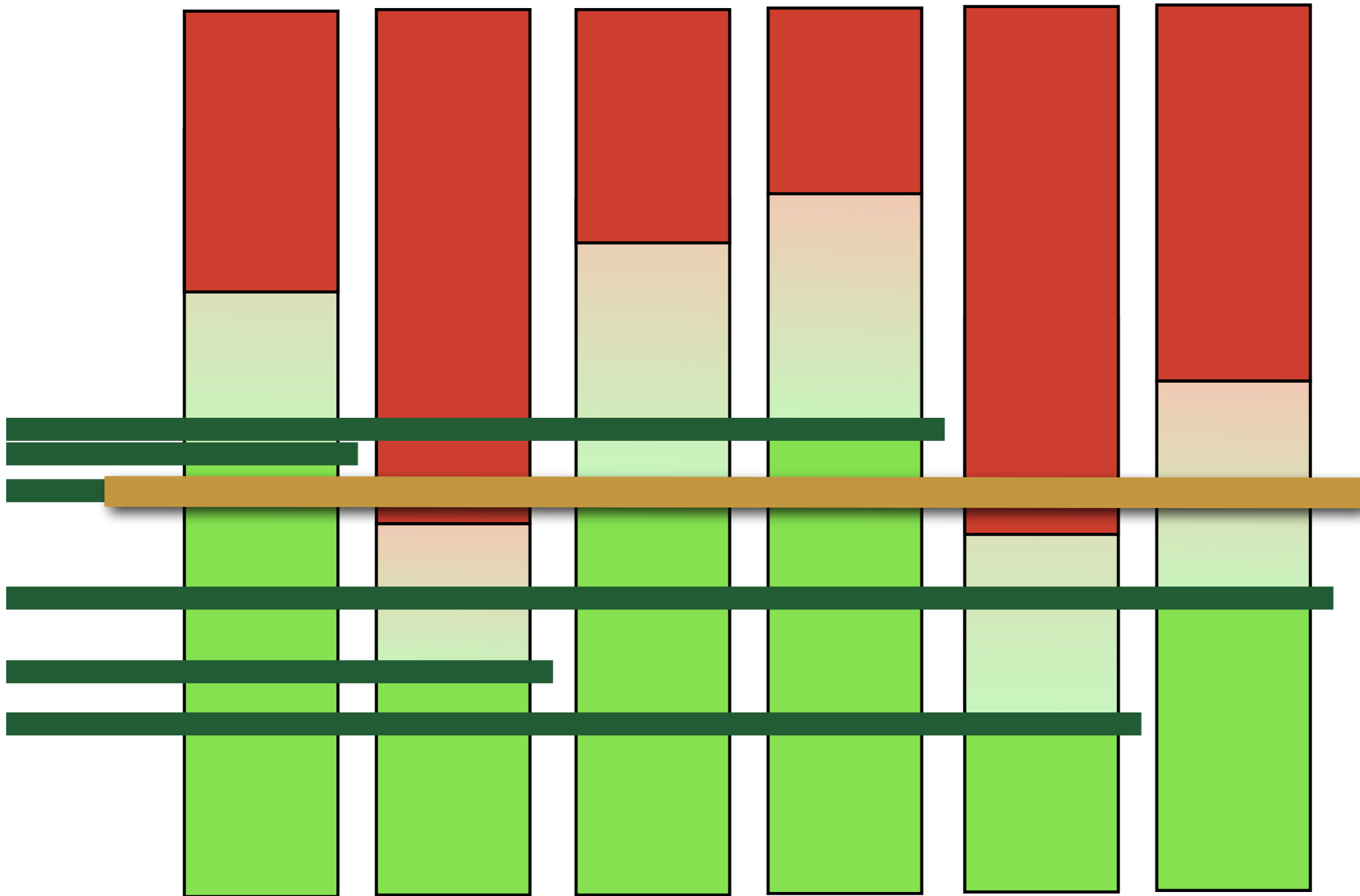
6 accumulators, top-3 retrieval



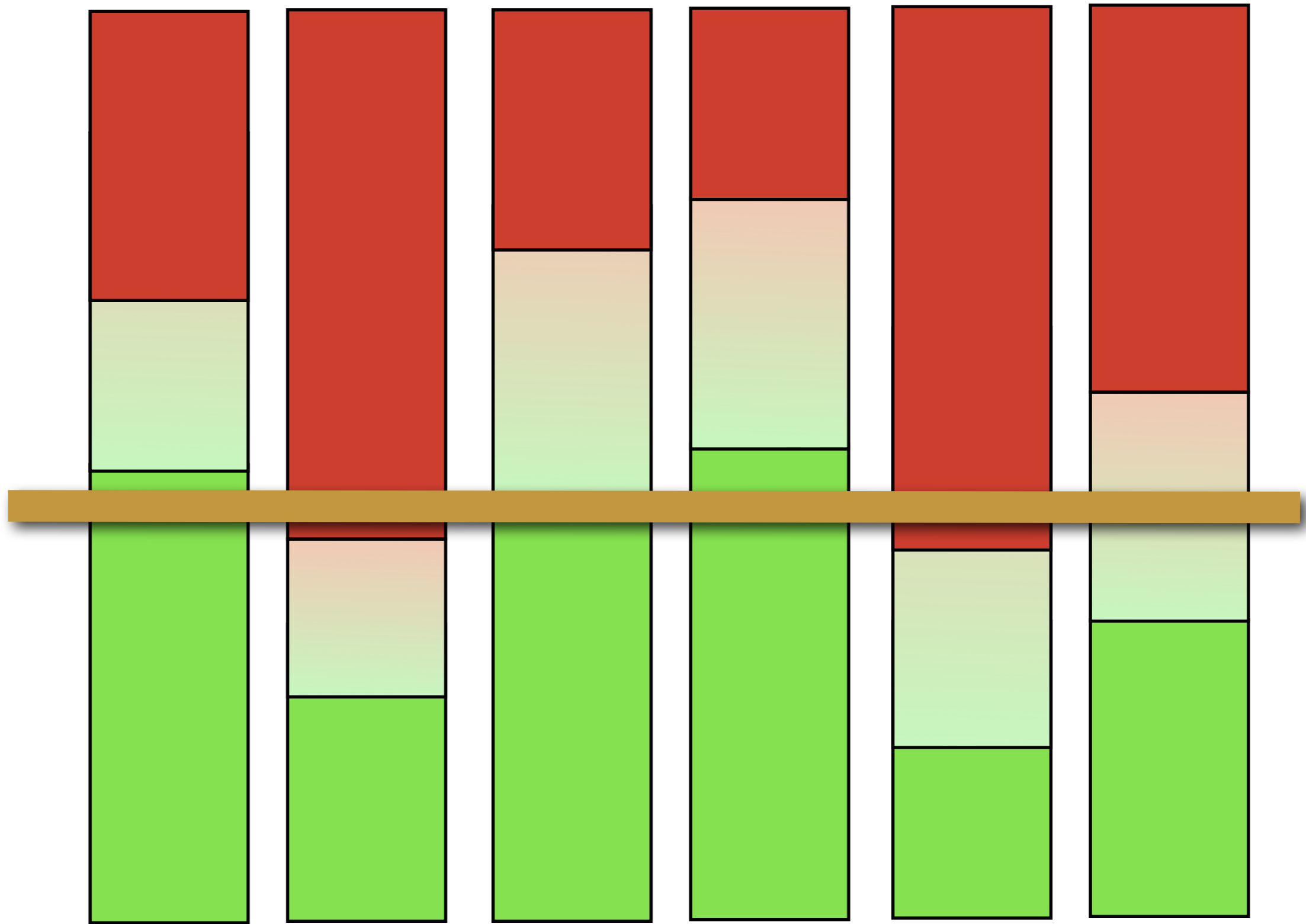
6 accumulators, top-3 retrieval



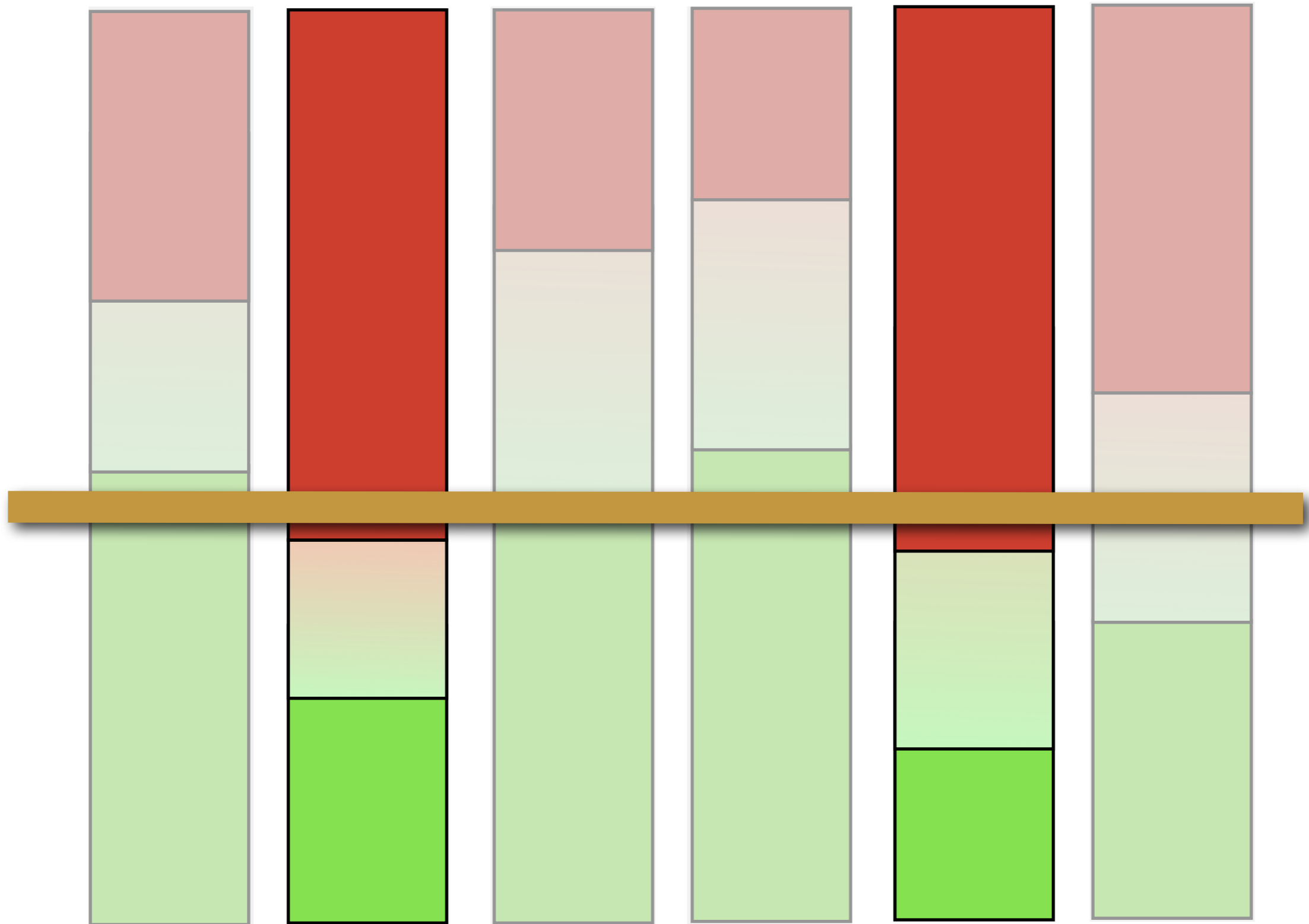
6 accumulators, top-3 retrieval



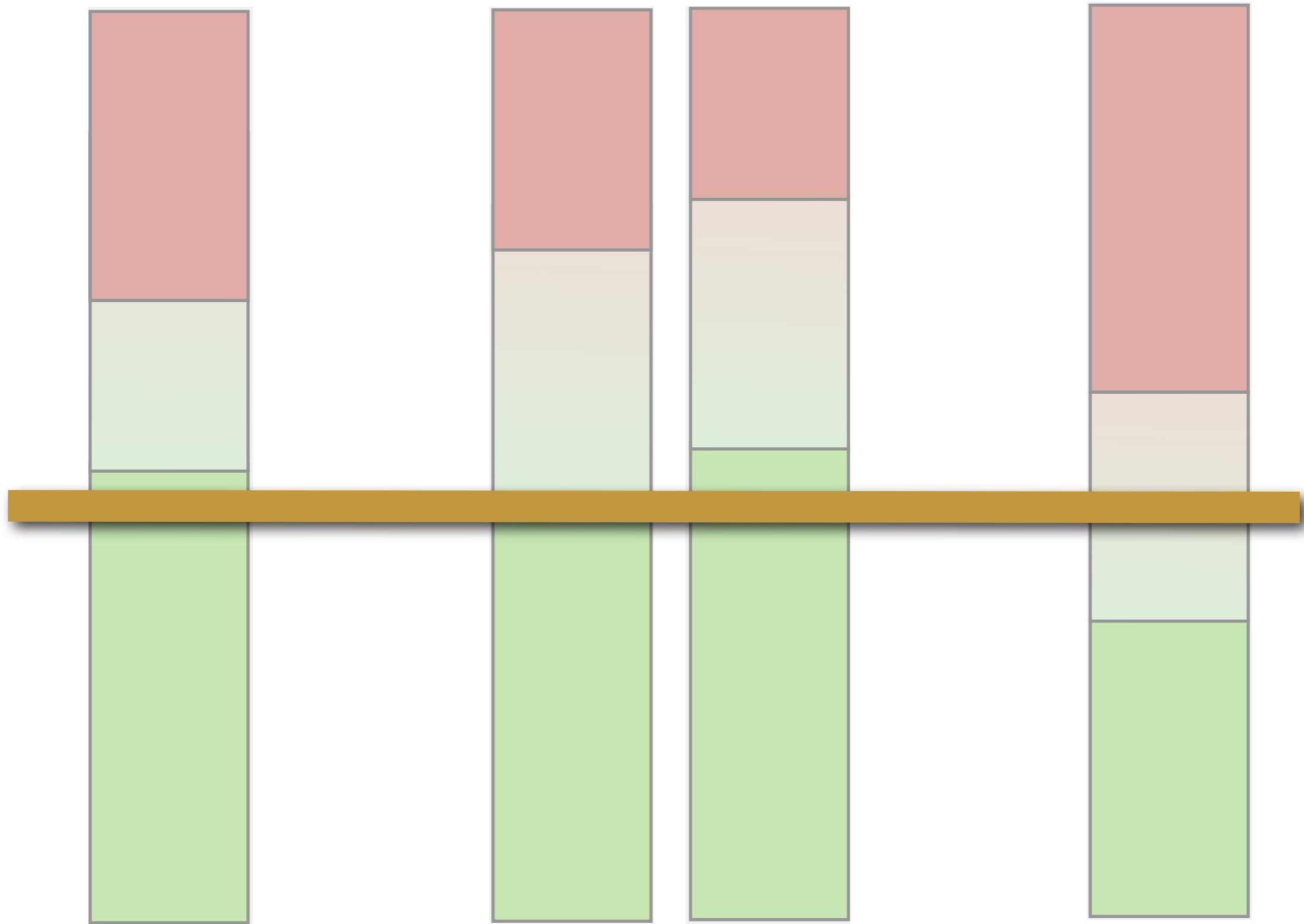
6 accumulators, top-3 retrieval



6 accumulators, top-3 retrieval



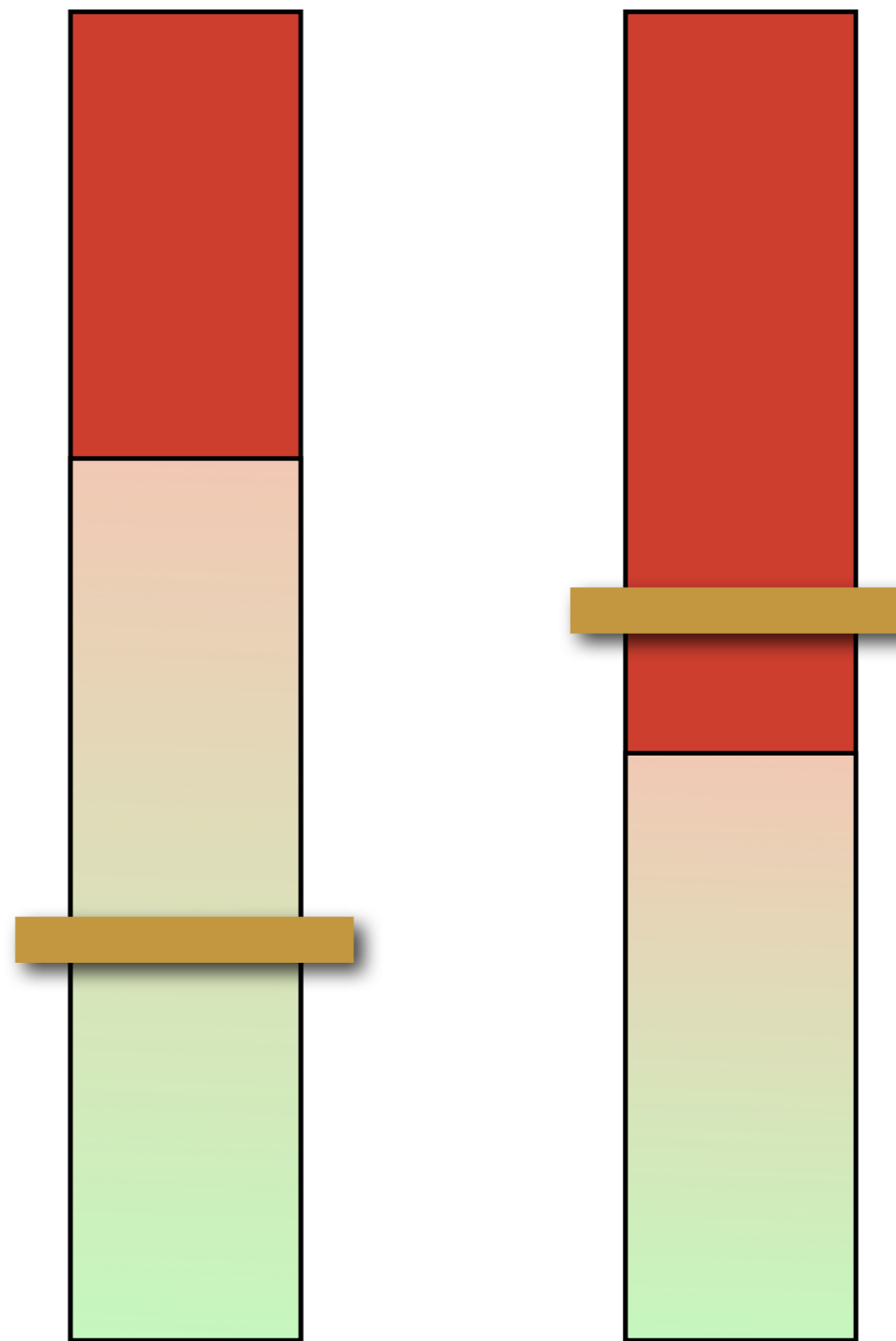
6 accumulators, top-3 retrieval



6 accumulators, top-3 retrieval

## unseen documents

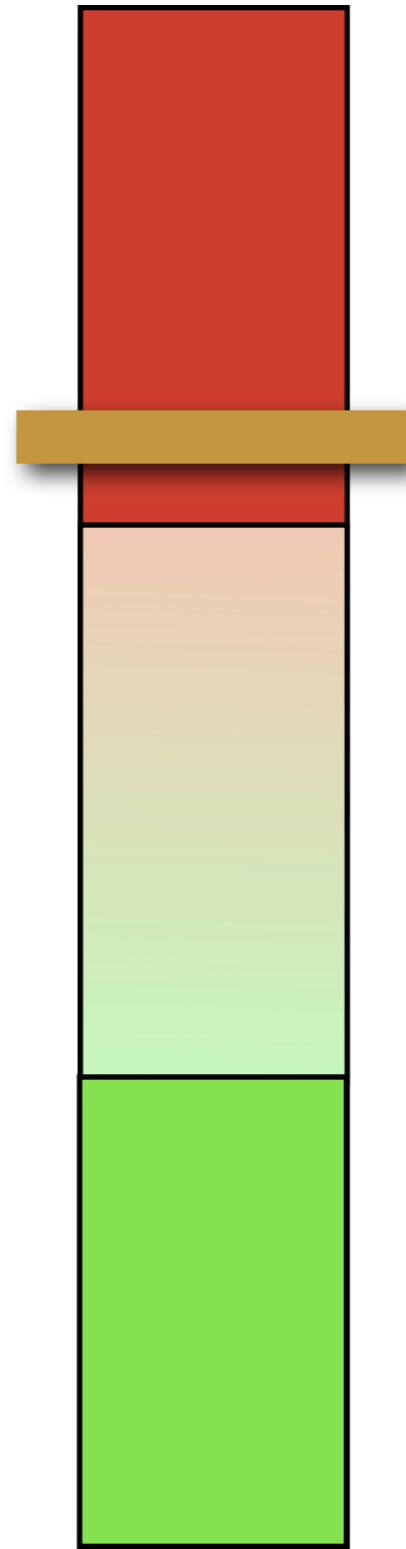
documents that have not yet been seen in inverted lists

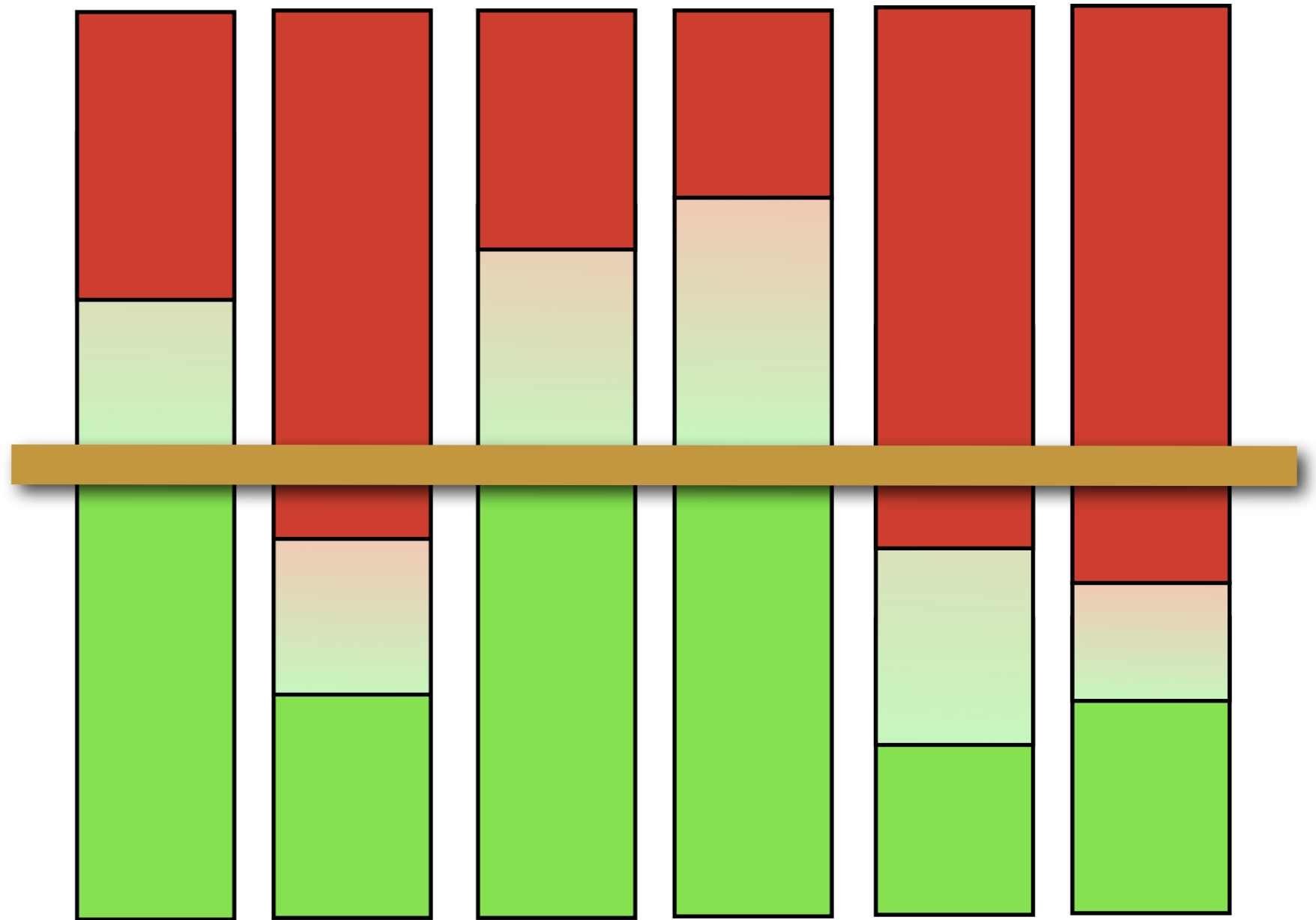


**OR**

**AND**

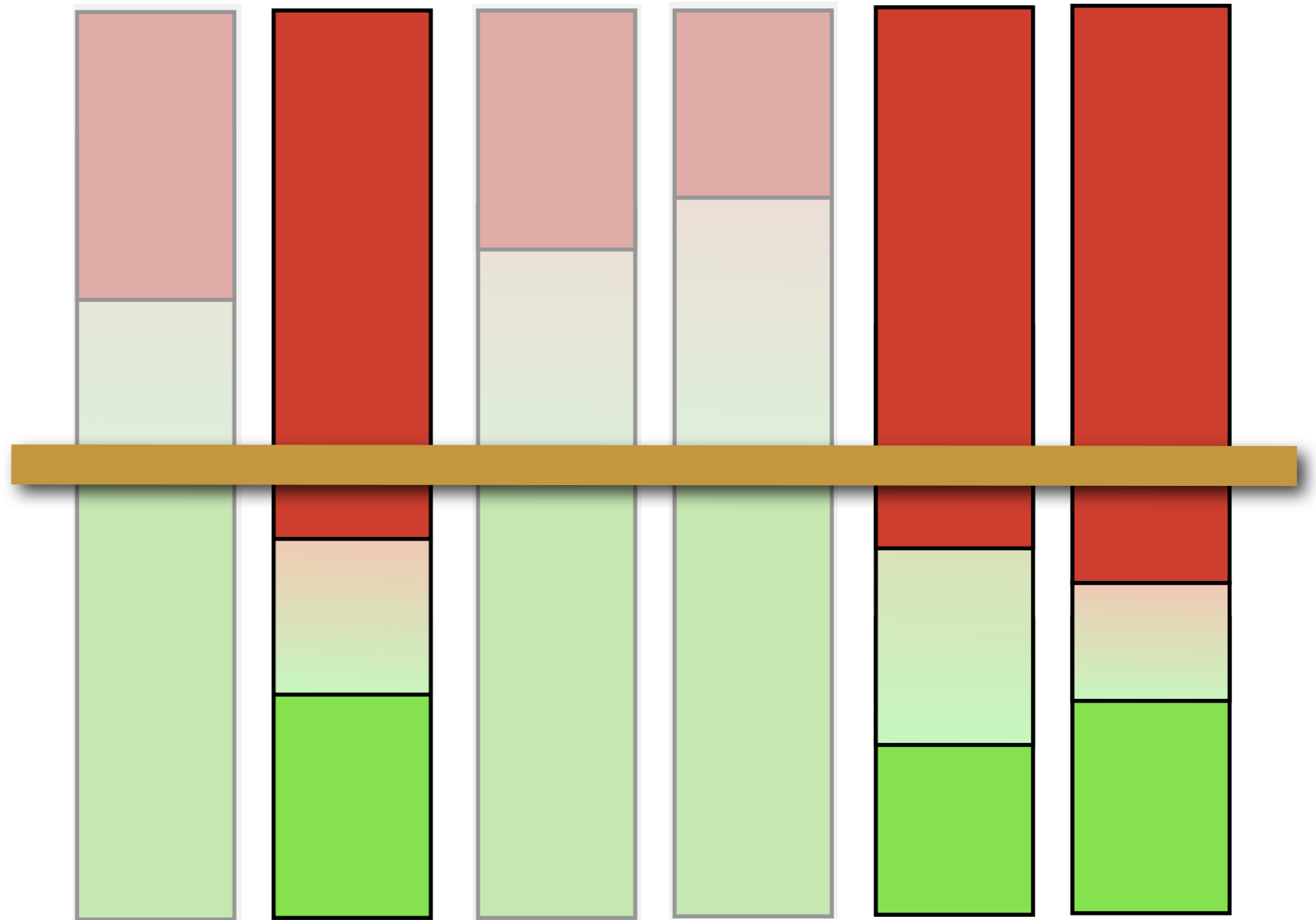
**trimming**





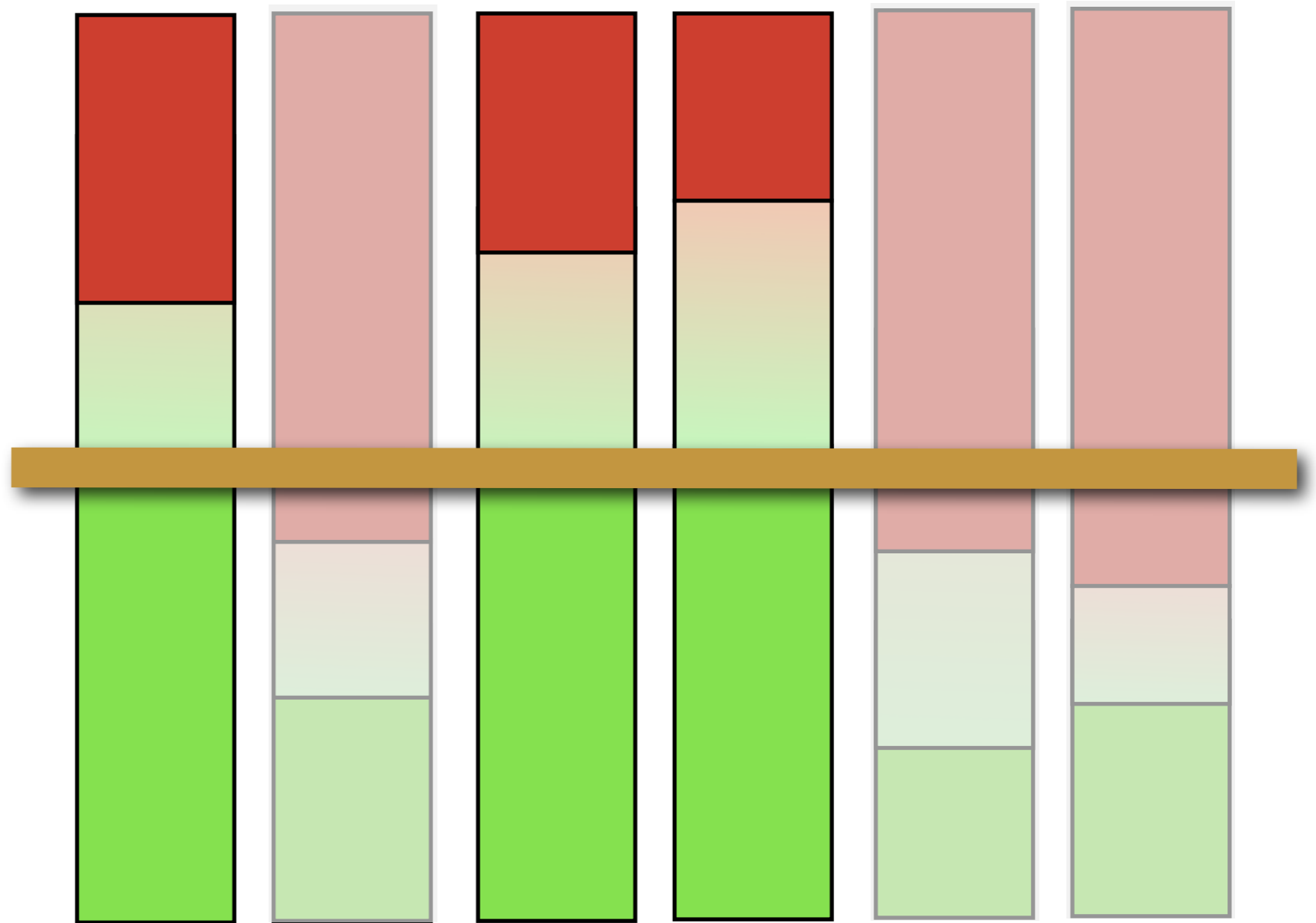
## top documents determined

top documents are known,  
although order is not



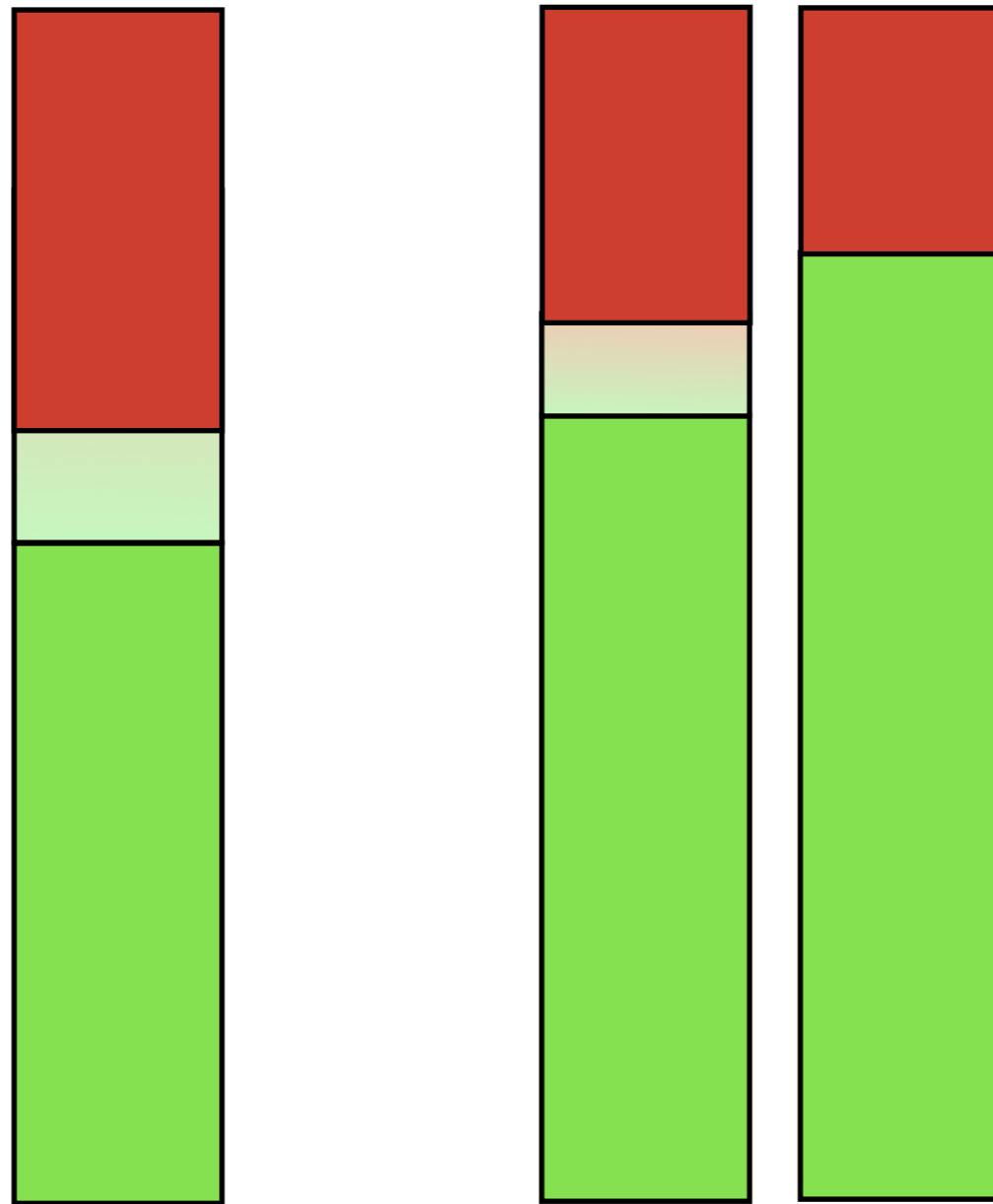
## top documents determined

top documents are known,  
although order is not



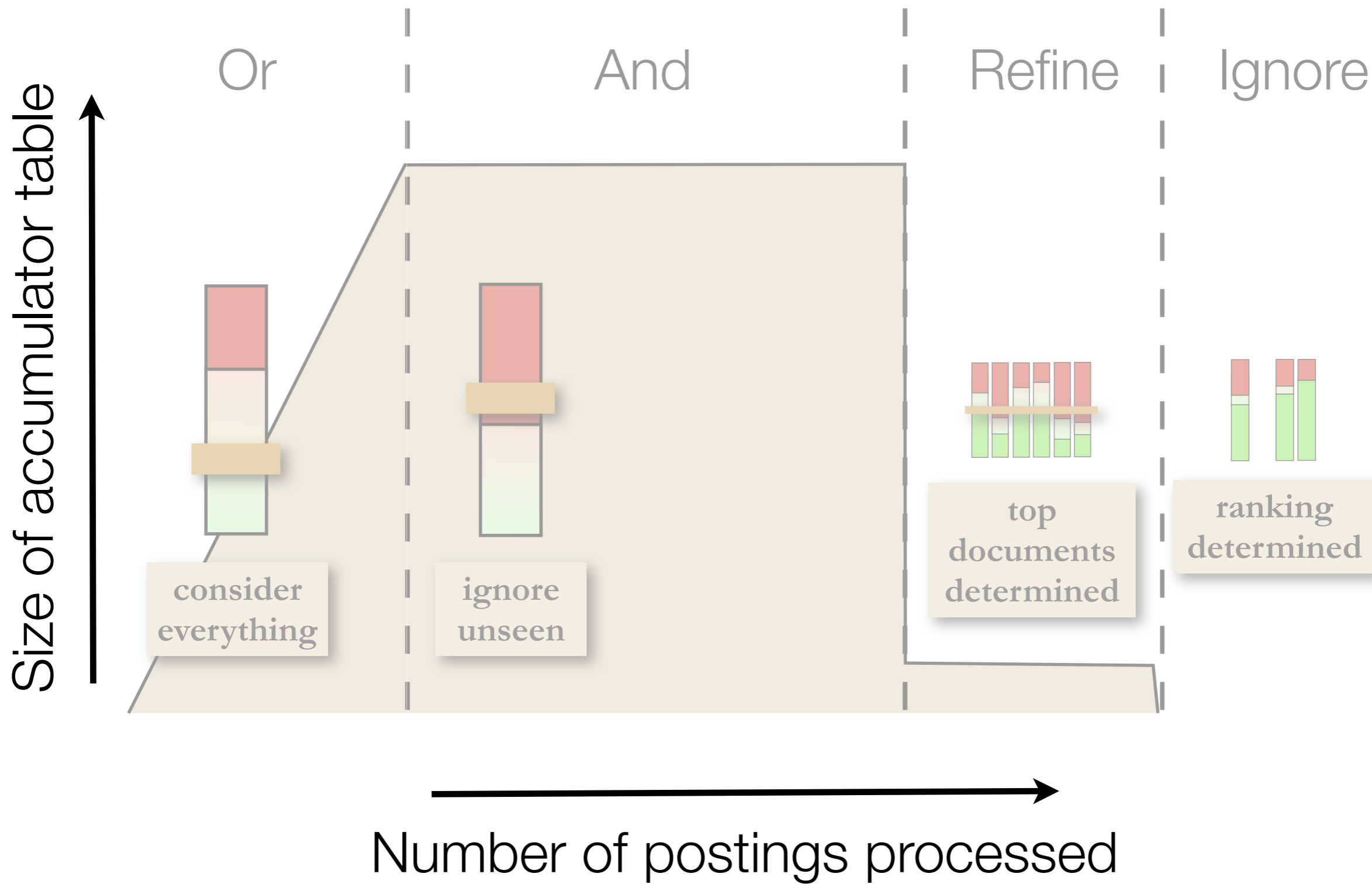
## top documents determined

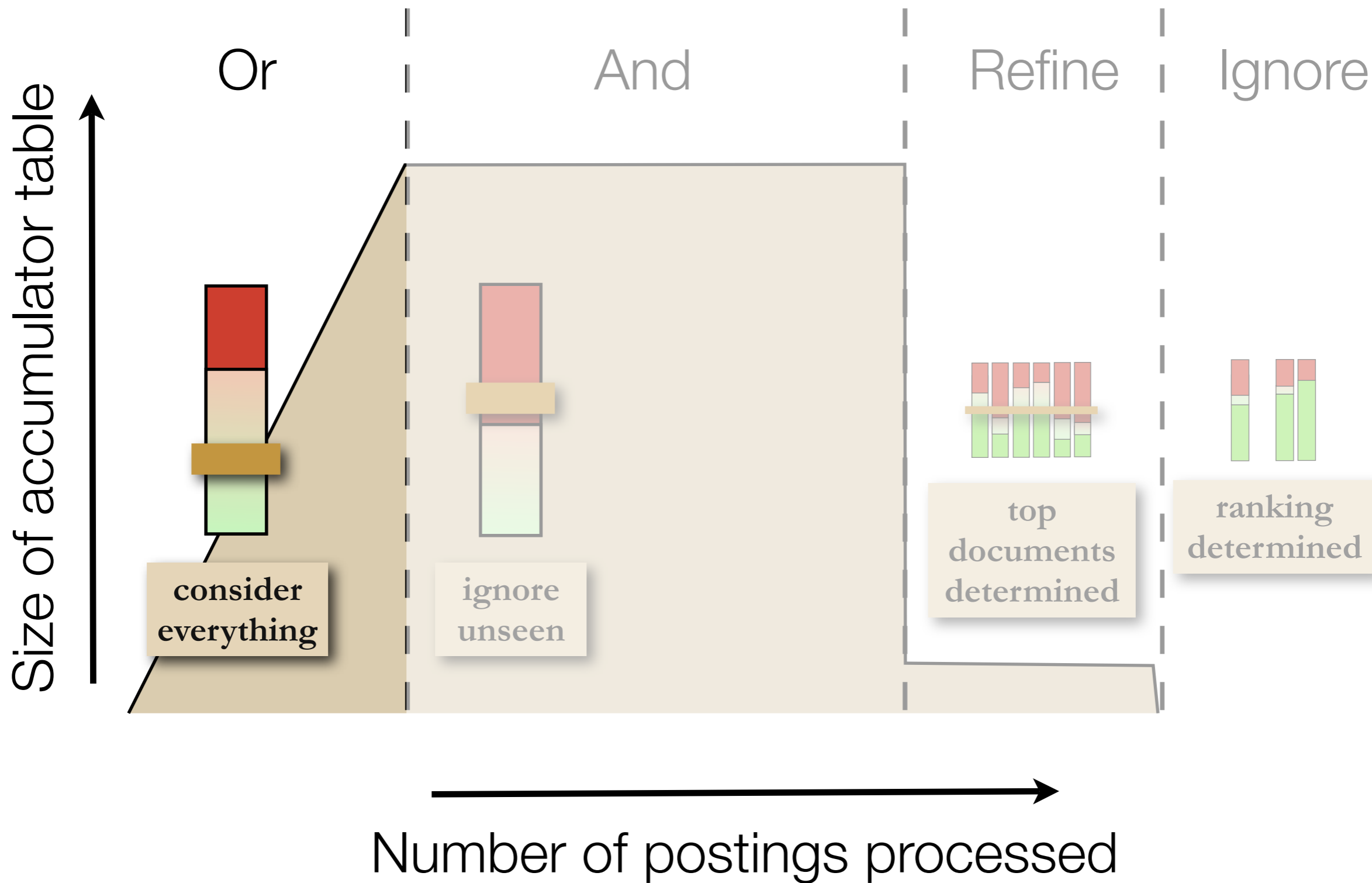
top documents are known,  
although order is not

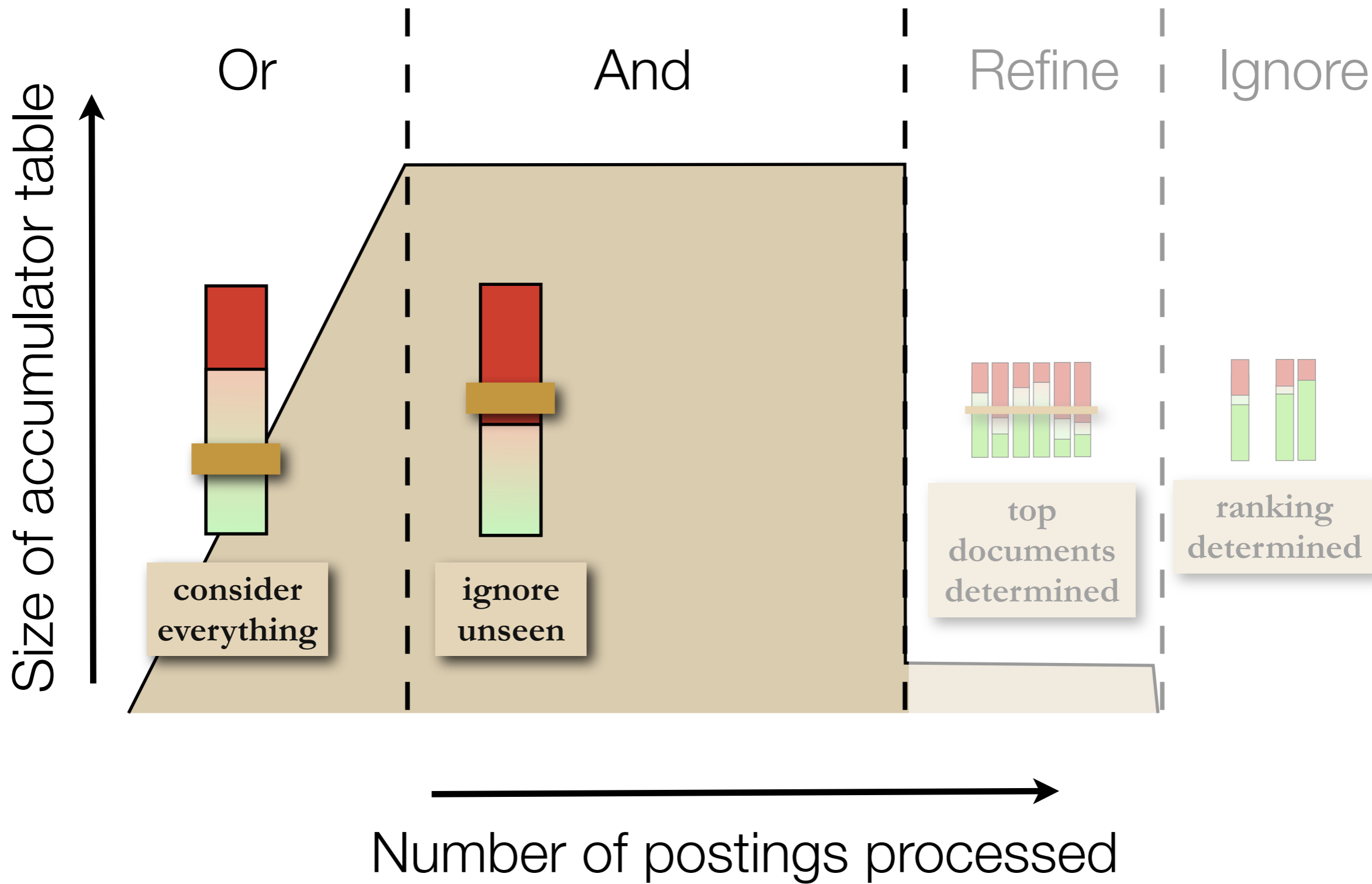


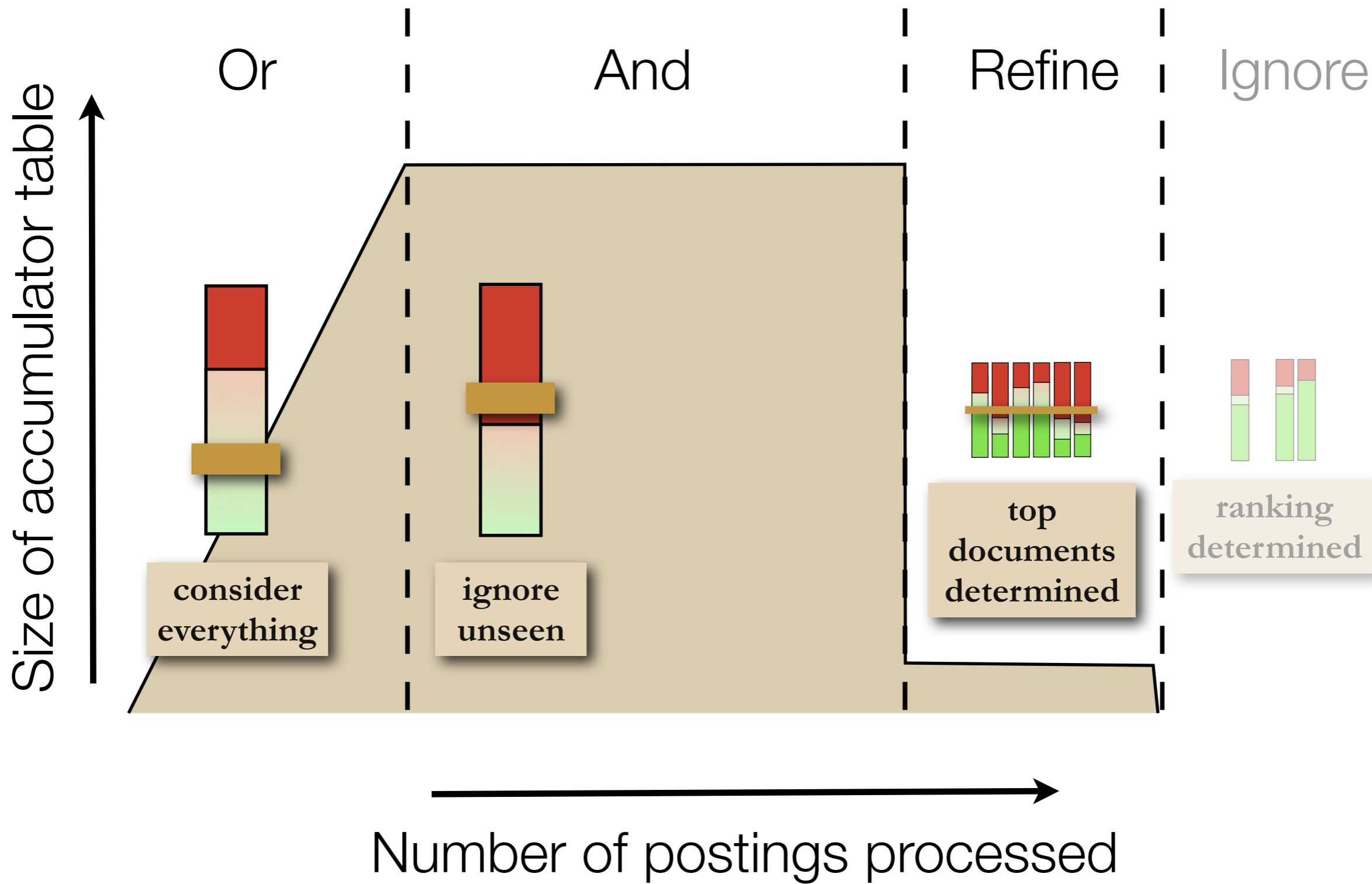
## ranking determined

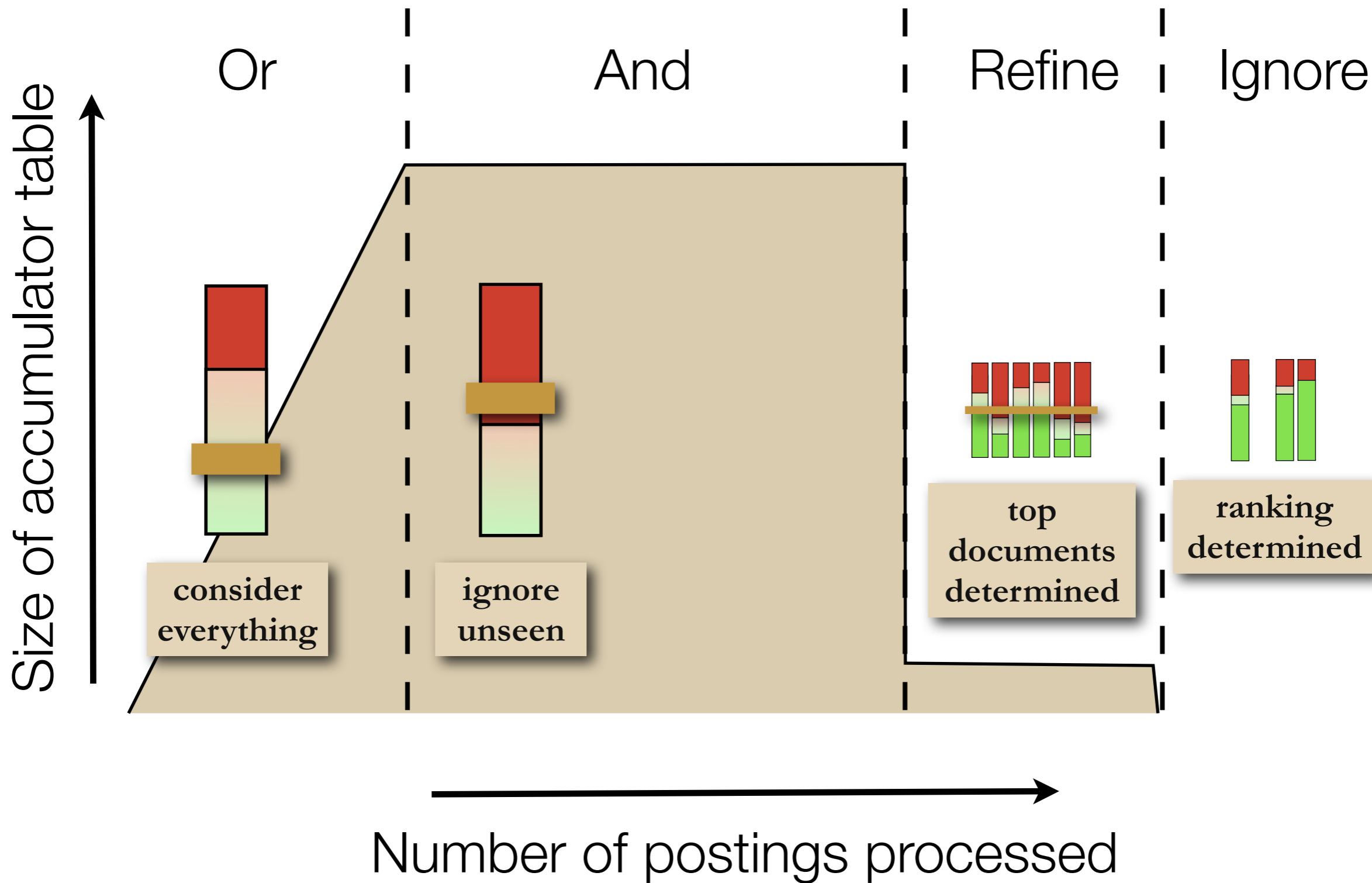
ranking is known, although  
exact scores are not

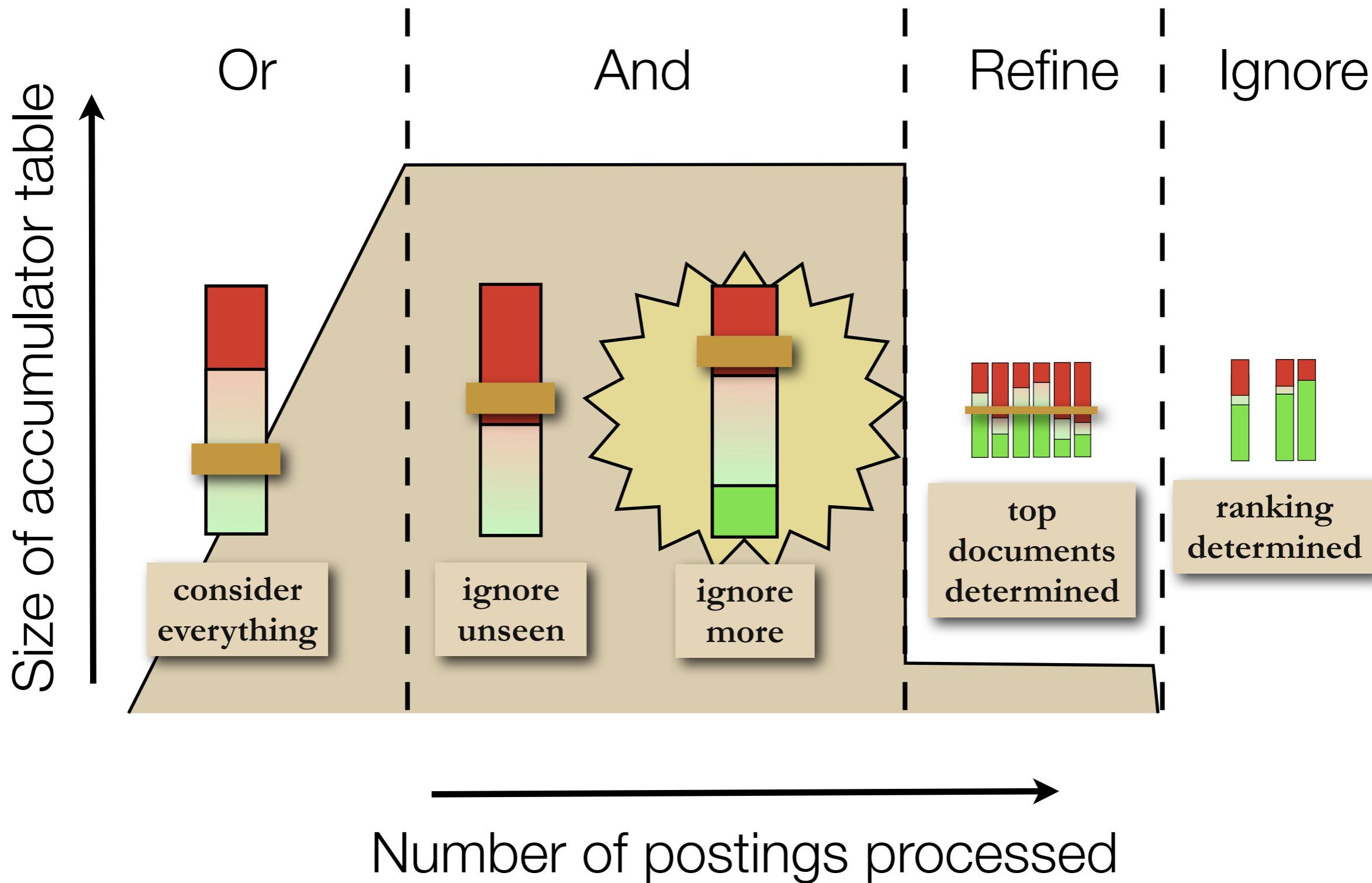


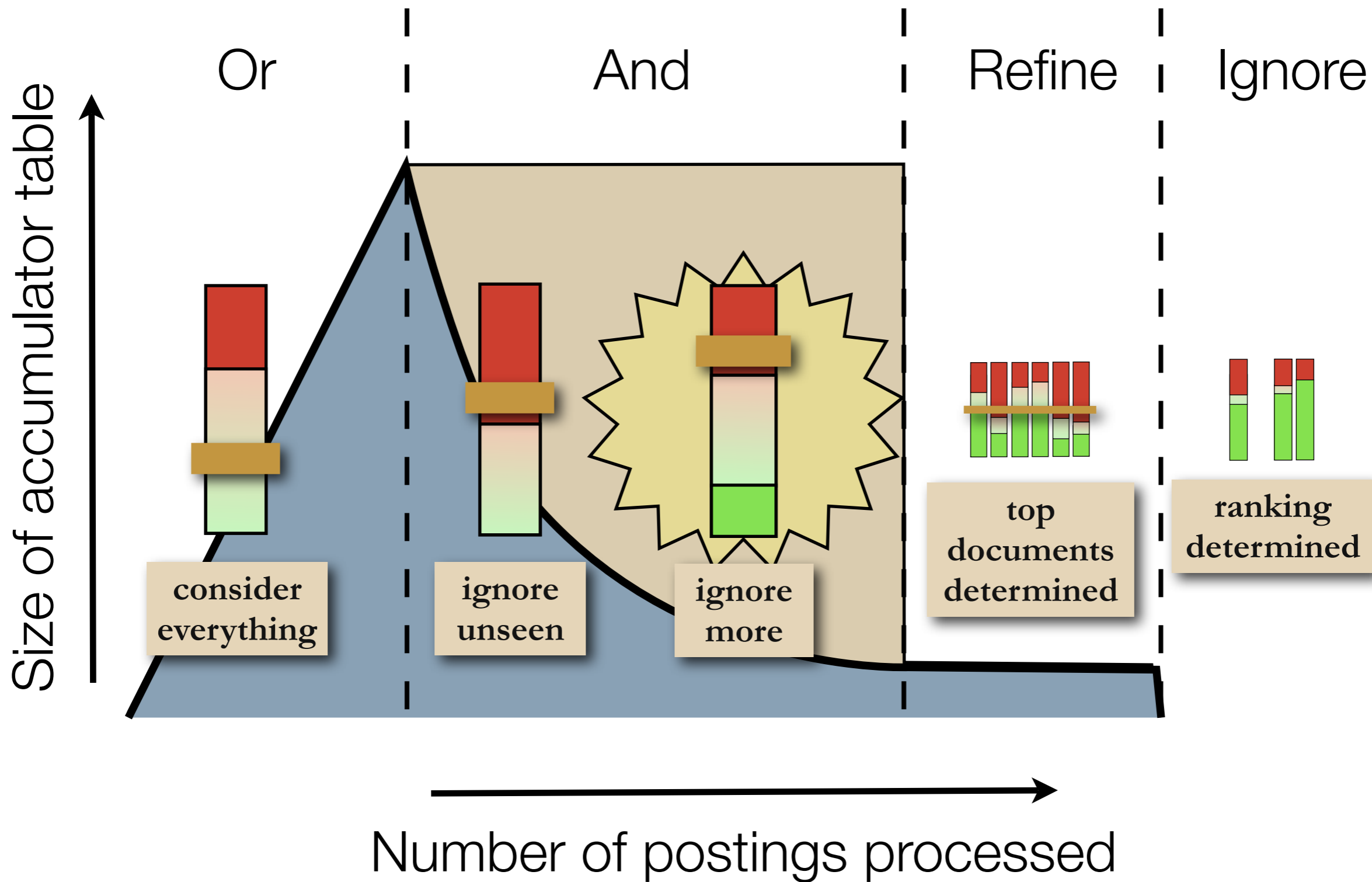






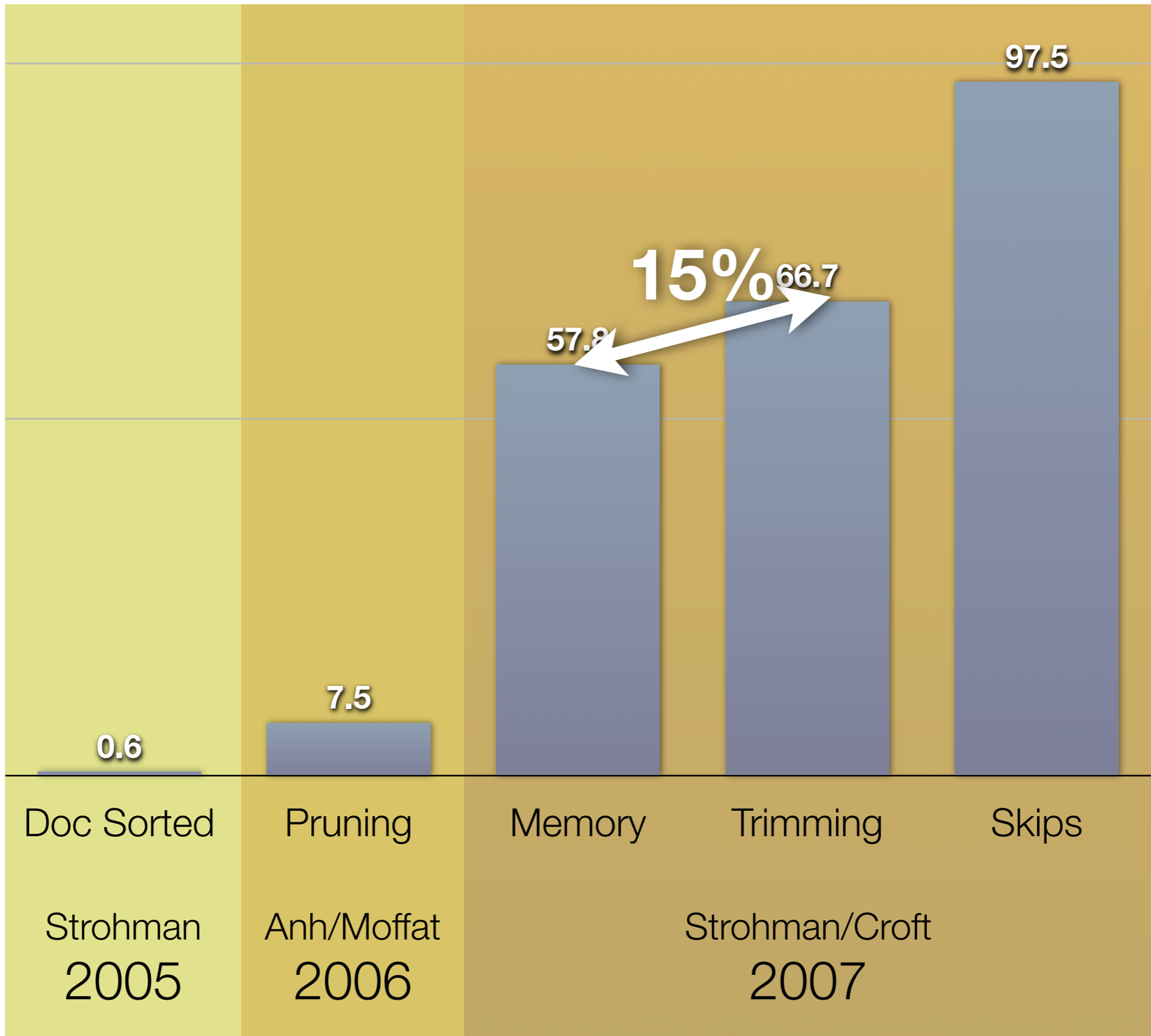






GOV2 queries/second

100  
50  
0



Doc Sorted

Pruning

Memory

Trimming

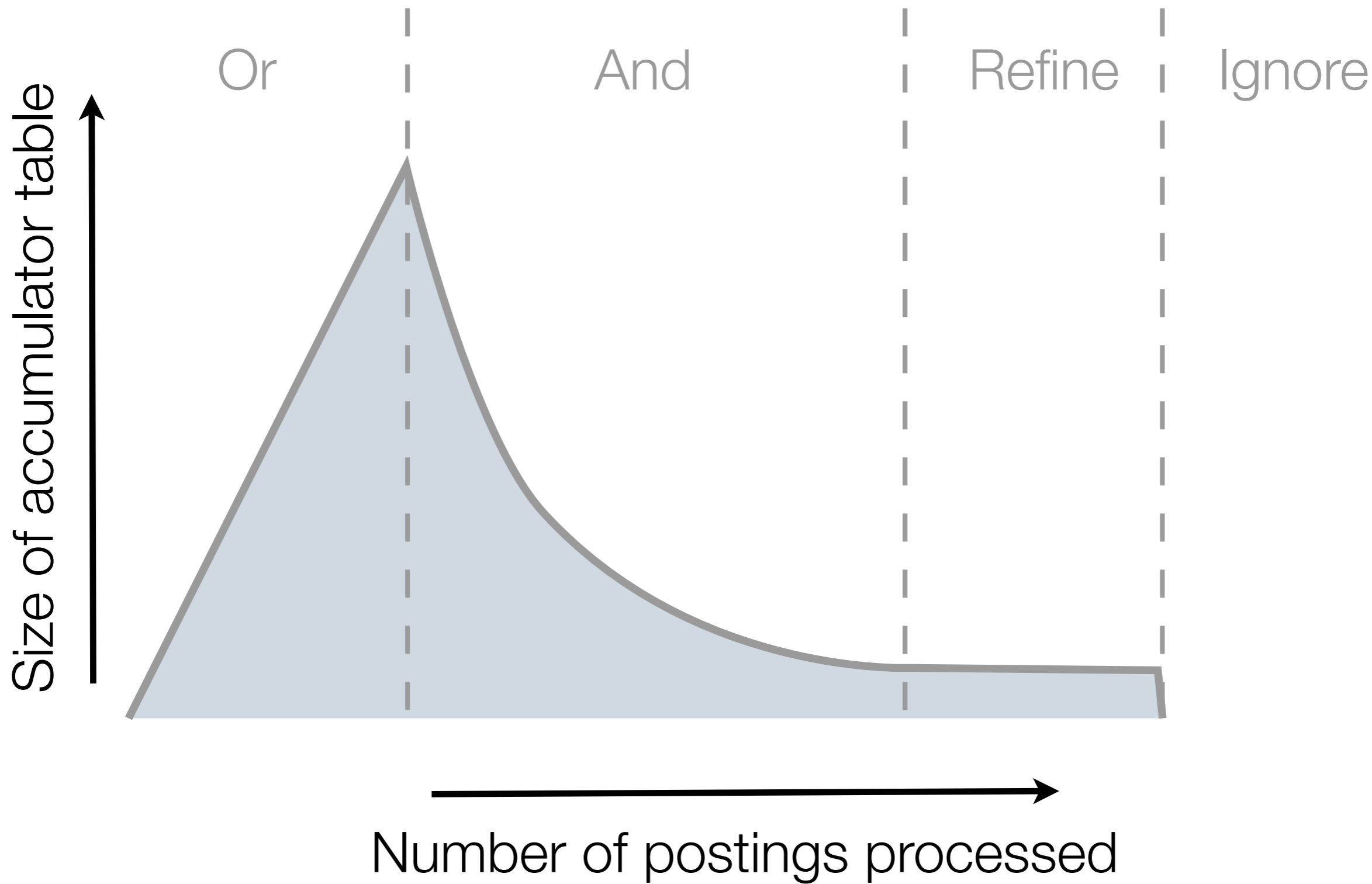
Skips

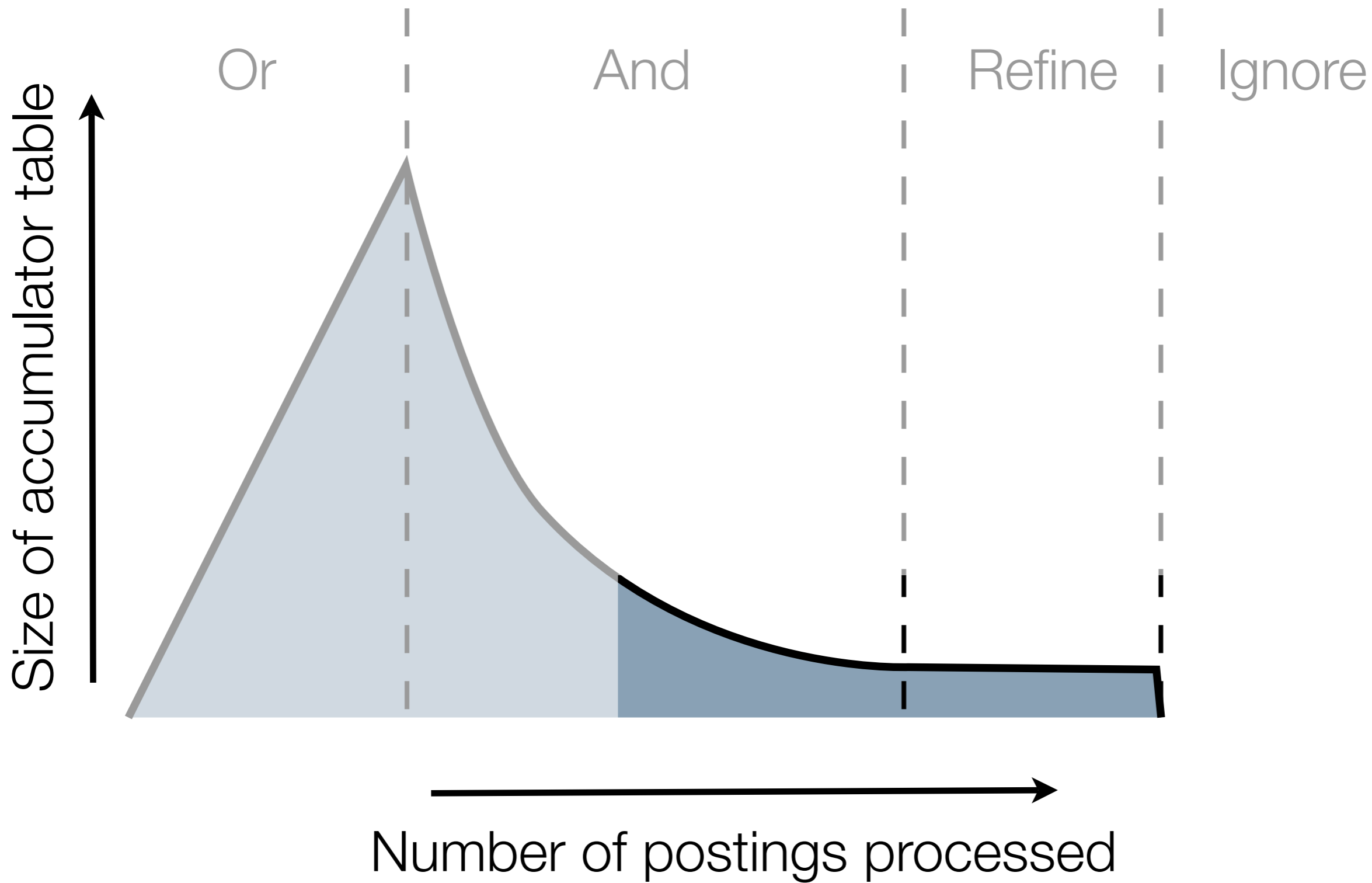
Strohman  
2005

Anh/Moffat  
2006

Strohman/Croft  
2007

Skipping





**8**

1

4

6

**7**

5

**7**

**6**

2

33

**5**

20

**29**

32

**4**

9

**11**

**3**

15

19

21

**8**

1

4

6

**7**

5

**7**

**6**

2

33

**5**

20

**29**

**4**

32

9

**11**

**3**

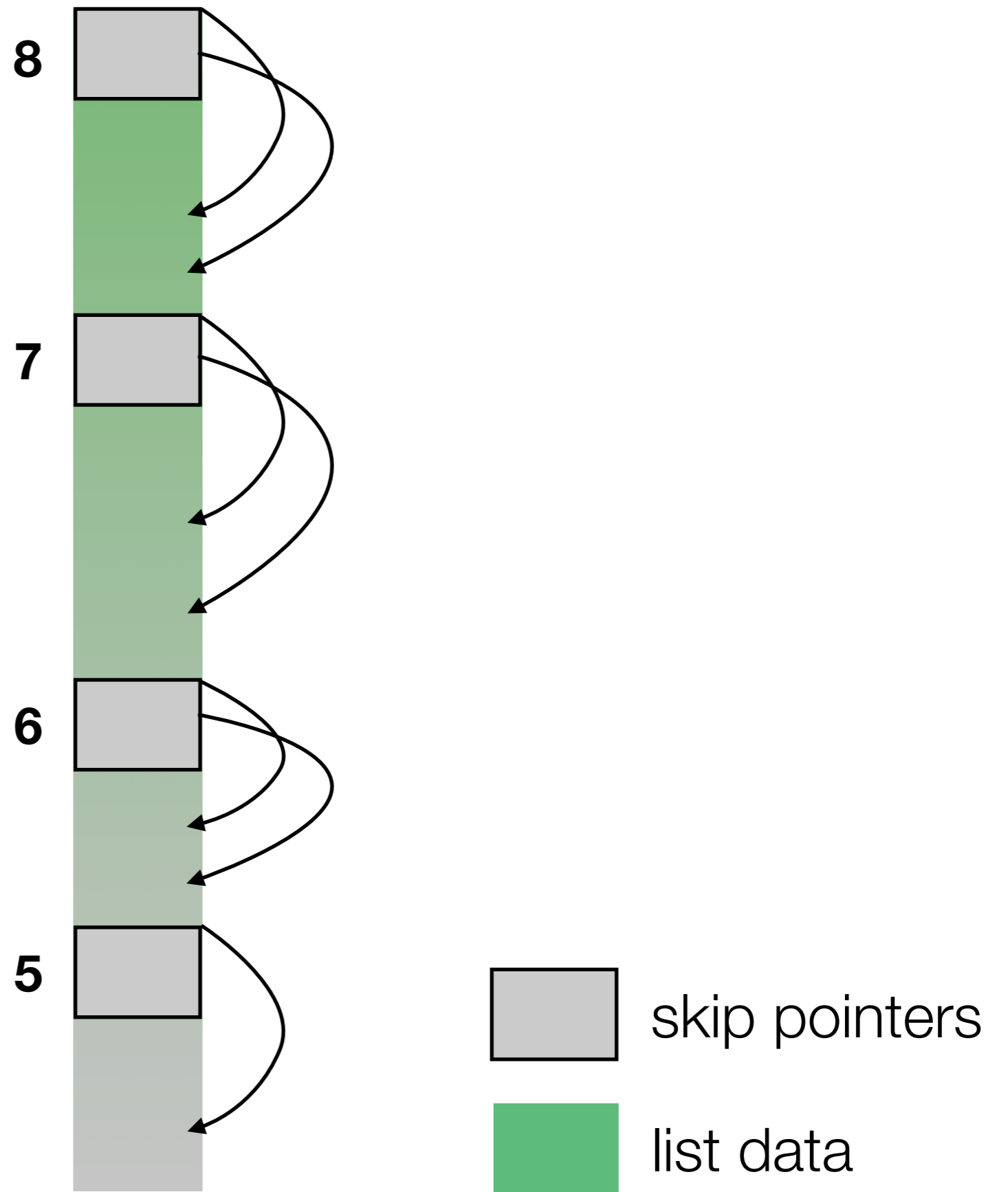
15

19

21

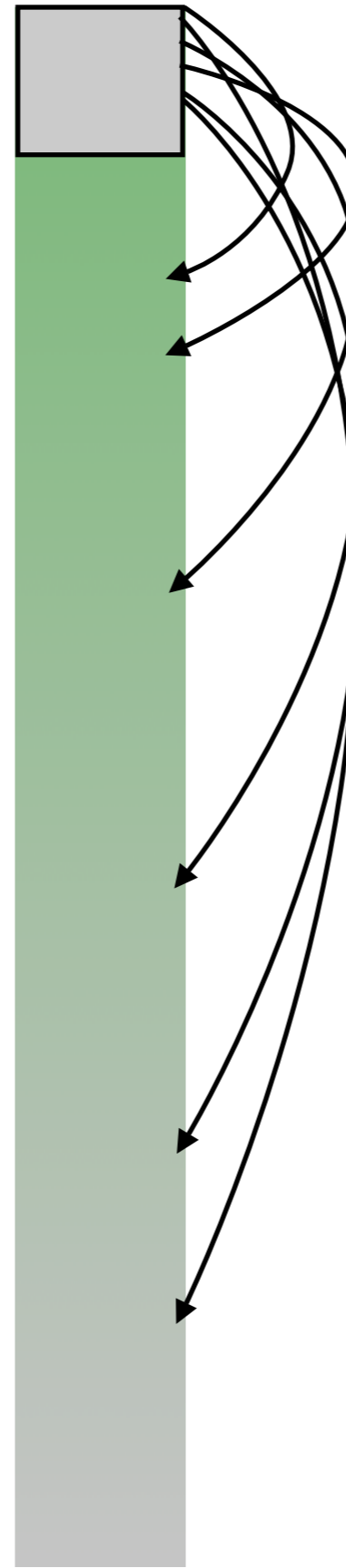
# skipping

lookup table allows fast access to segments of the compressed list



# skipping

lookup table allows fast access to segments of the compressed list



skip pointers



list data

# Skipping disadvantages

extra code complexity

more data to read and process

## Solution

dynamically choose when to skip

if in OR mode:

**don't** skip

if in AND mode with *many* accumulators:

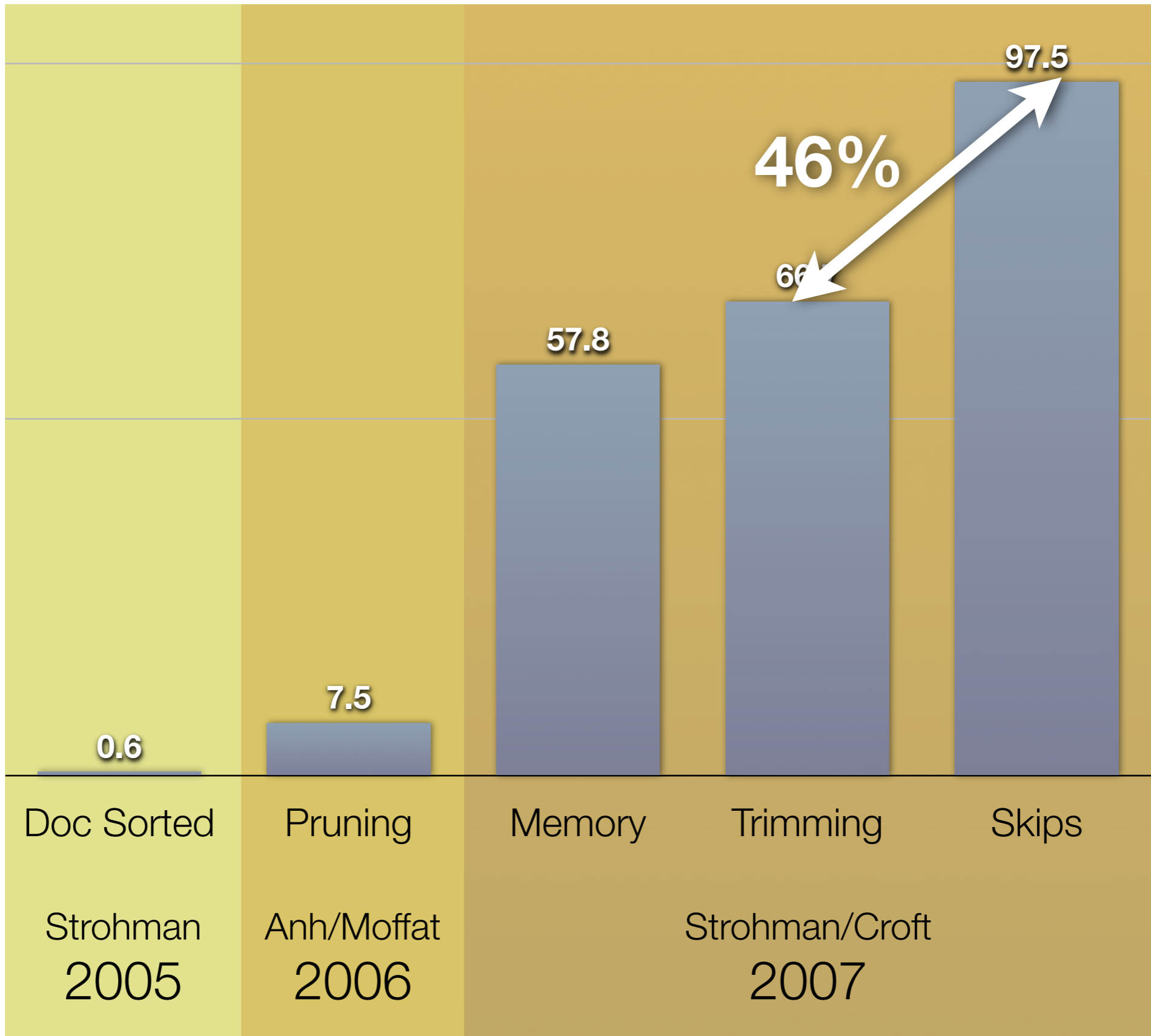
**don't** skip

if in AND mode with *few* accumulators:

**do** skip

GOV2 queries/second

100  
50  
0



Doc Sorted  
Strohman  
2005

Pruning  
Anh/Moffat  
2006

Memory

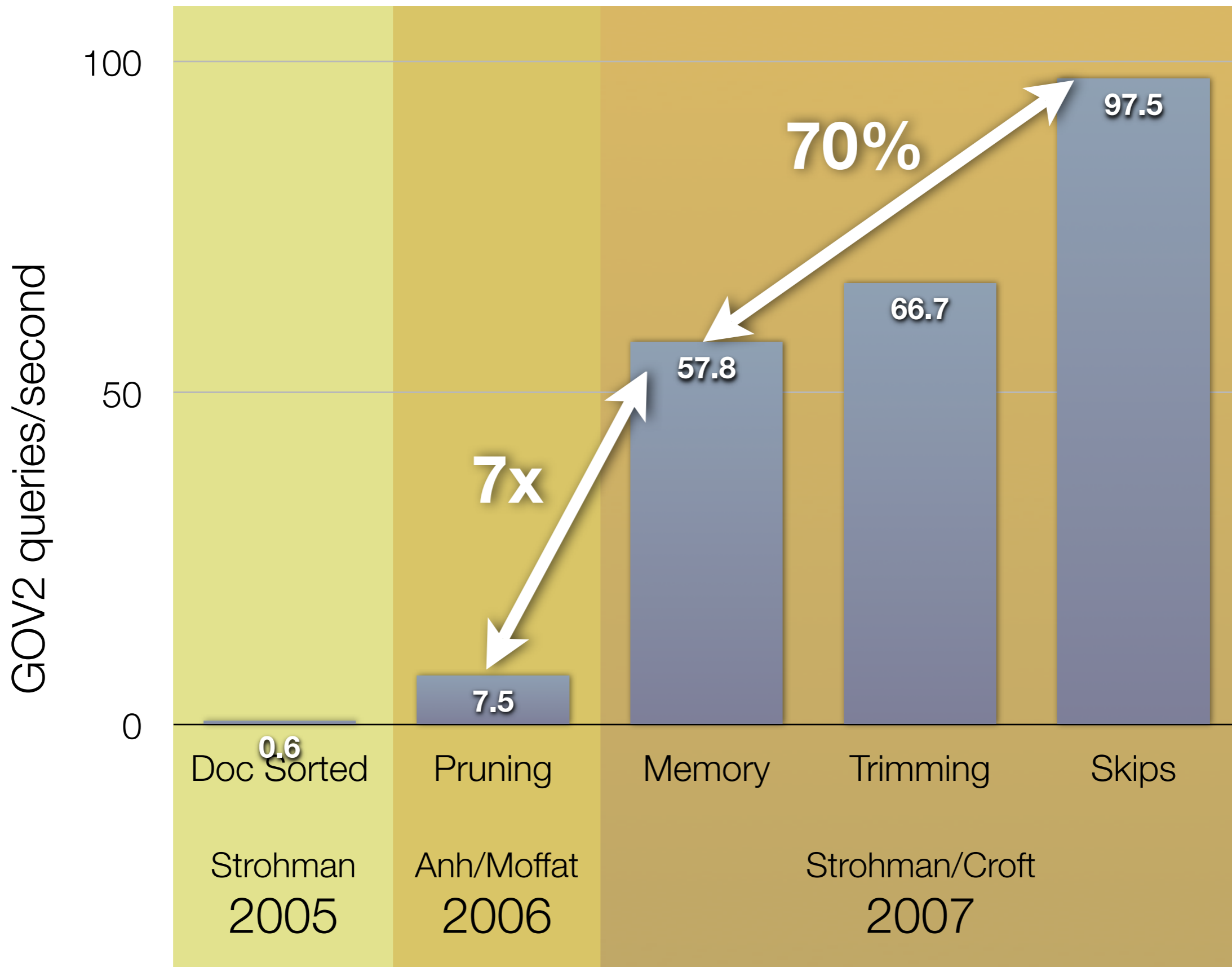
Trimming  
Strohman/Croft  
2007

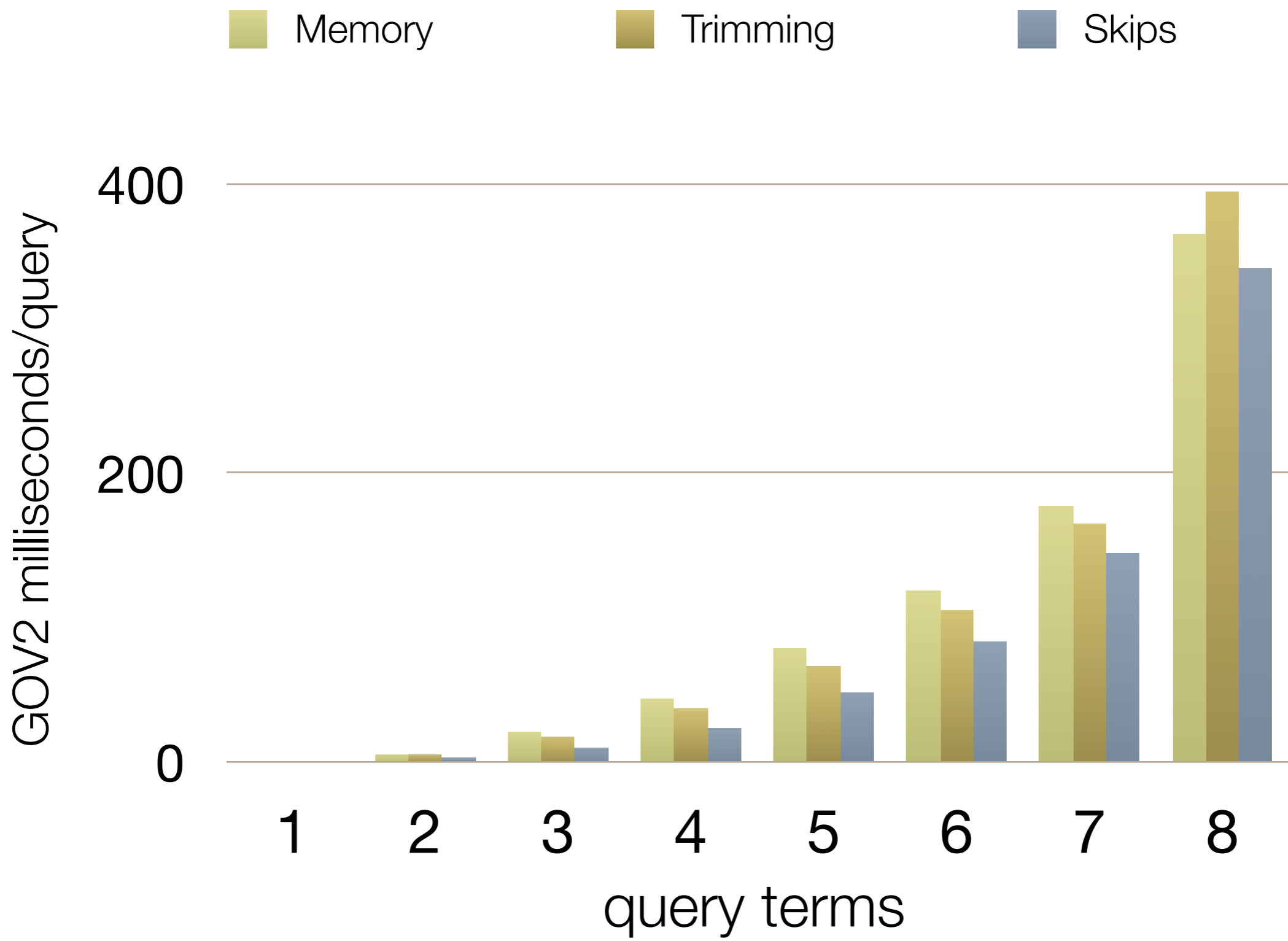
Skips

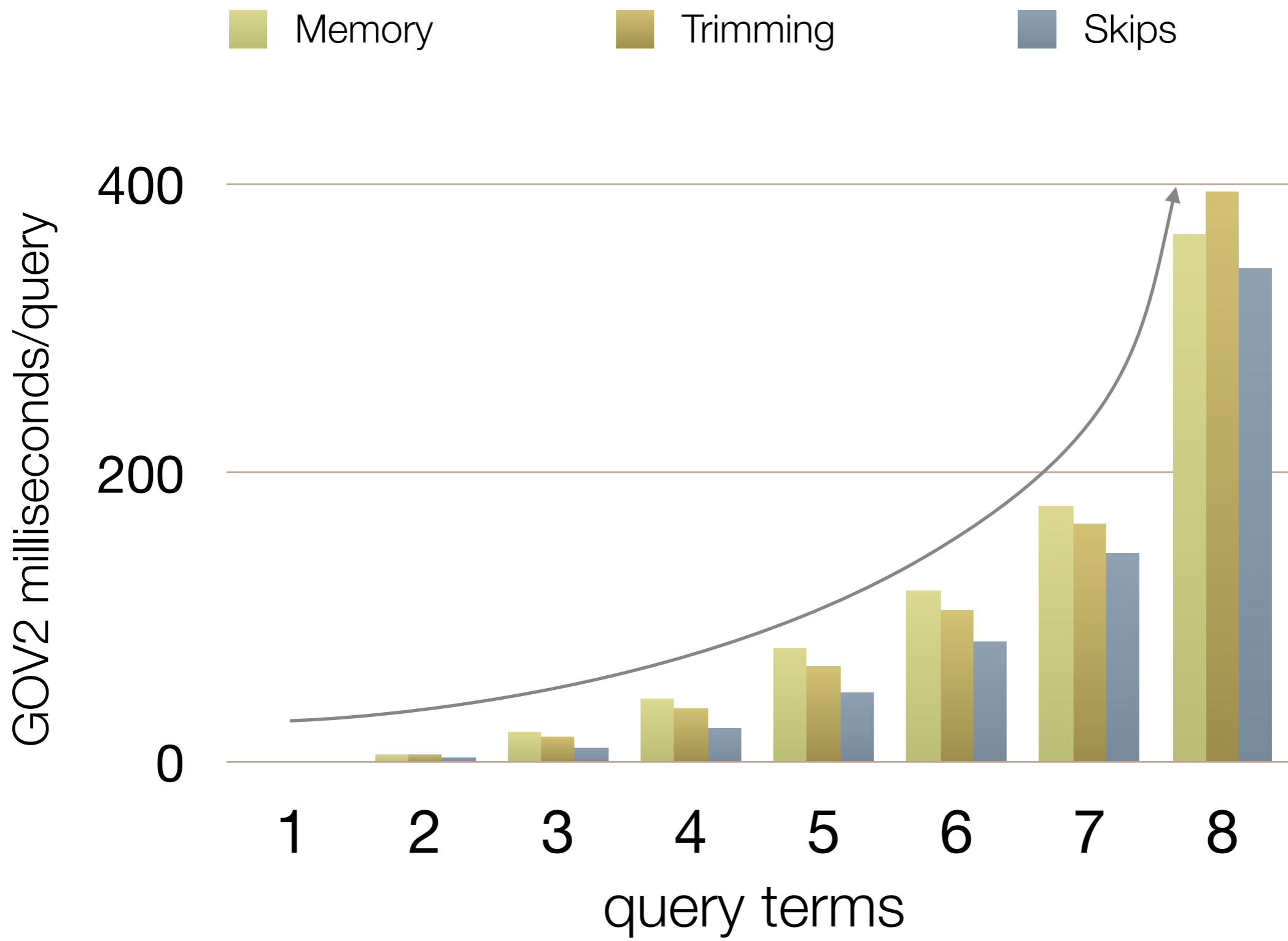
## Results, Part II

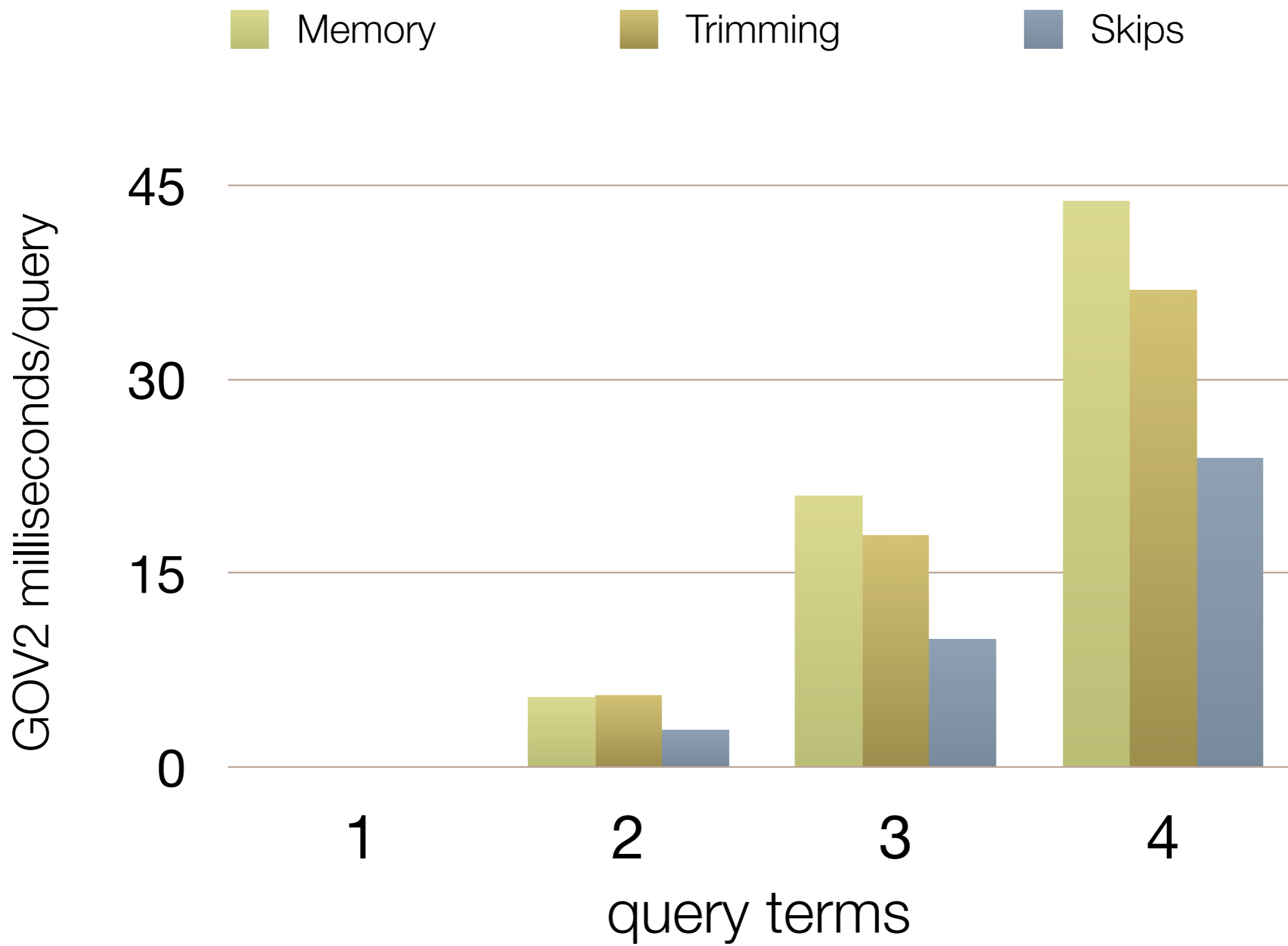
**everything in memory**

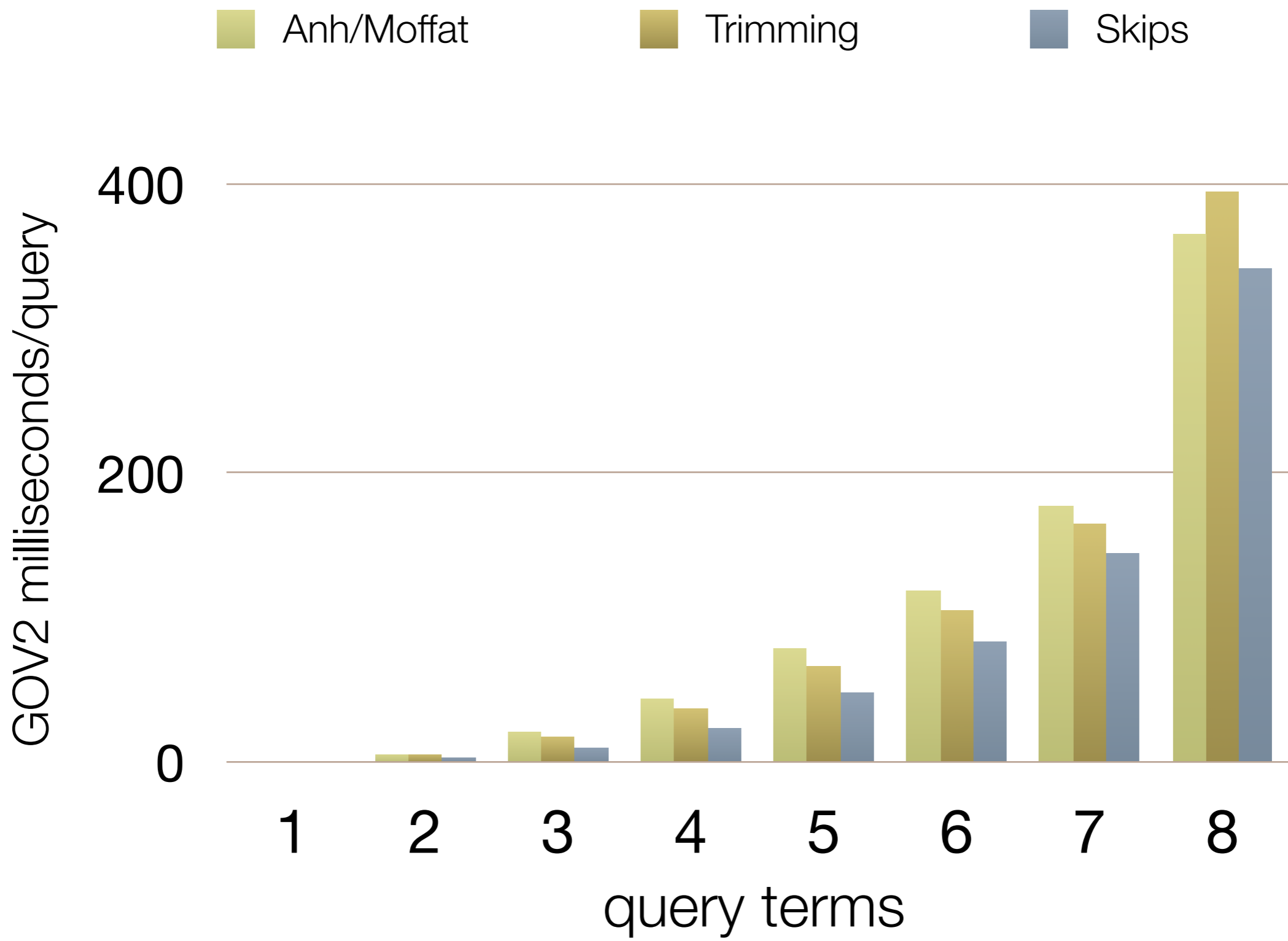
**efficient dynamic pruning**

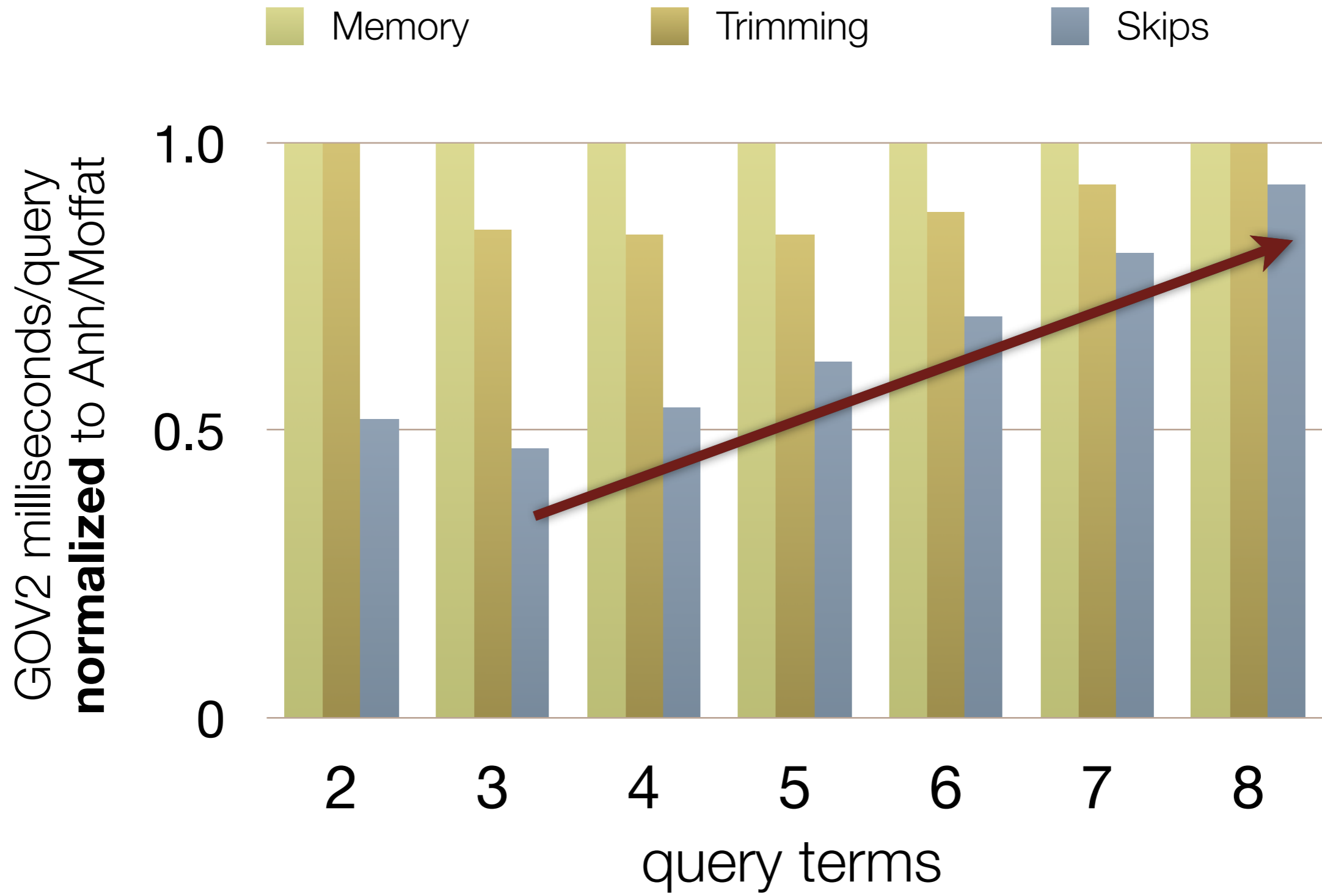


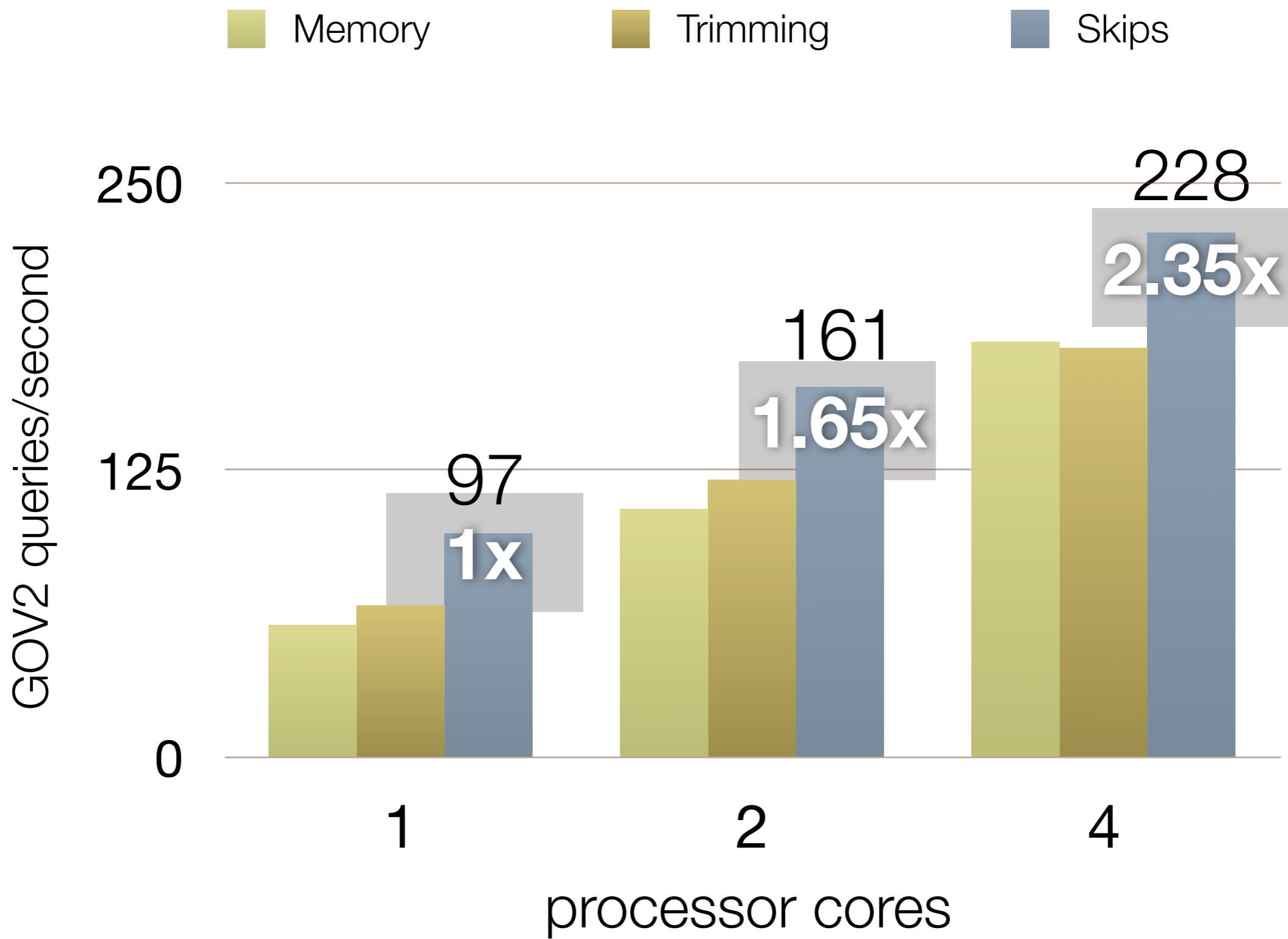












## **In the paper**

model for skip length performance

skip length training

accumulator structures

cache effects in skipping

TREC 2006 results

query length distributions

Conclusion

**Everything in memory**

**Efficient dynamic pruning**

# Everything in memory

it may be worth spending extra for more memory

7x performance improvement

memory bandwidth is important

# Efficient dynamic pruning

## Everything in memory

it may be worth spending extra for more memory

7x performance improvement

memory bandwidth is important

## Efficient dynamic pruning

like static pruning without the risk

70% faster than previous approaches

skipping and trimming work together

important to study the memory-only case

# GALAGO

[WWW.GALAGOSEARCH.ORG](http://WWW.GALAGOSEARCH.ORG)

Java-based distributed indexer  
C++ and Java retrieval engines

*Fall 2007*