

**Thinking about performance**

# **Search: a case study**

Perf: speed/power/etc.

Perf: why do we care?

“Premature optimization is the  
root of all evil”

“We should forget about small efficiencies, say about 97% of the time”

Different designs:

100x - 1000x perf difference

Working code attracts people who want to code.  
Design documents attract people who want to talk.

RETWEETS

**538**

LIKES

**561**





“Coding feels like real work”

Whiteboard: 1h/iteration

Implementation: 2yr/iteration

# Scale

(precursor to perf discussion)

**10k; 10M; 10G**  
(5kB per doc)

What's the actual problem?

AND queries

**10k; 10M; 10G**  
(5kB per doc)

10k

One person's email  
One forum



$$5\text{kB} * 10\text{k} = 50\text{MB}$$

10k

50MB is small!

10k

\$50 phone => 1GB RAM

## Naive algorithm

```
for loop over all documents {  
  for loop over terms in document {  
    // matching logic here.  
  }  
}
```

10k

**10k; 10M; 10G**  
(5kB per doc)

10M

~Wikipedia sized

10M

$$5\text{kB} * 10\text{M} = 50\text{GB}$$

10M

**\$2000 for 128GB server**

(Broadwell single socket Xeon-D)

10M



25 GB/s memory bandwidth

10M

$$50\text{GB} / 25 \text{ GB/s} = 2\text{s}$$

(½ query per sec (QPS))

10M

Is 2s latency ok?

10M

Is 1/2 QPS ok?

10M

# Larger service

Latency == \$\$\$

10M

# Latency == \$\$\$

<http://assets.en.oreilly.com/1/event/29/Keynote%20Presentation%202.pdf>

<http://www.bizreport.com/2016/08/mobify-report-reveals-impact-of-mobile-website-speed.html>

<http://assets.en.oreilly.com/1/event/29/The%20User%20and%20Business%20Impact%20of%20Server%20Delays.%20Additional%20Bytes.%20and%20HTTP%20Chunking%20in%20Web%20Search%20Presentation.pptx>

10M

<http://assets.en.oreilly.com/1/event/27/Varnish%20-%20A%20State%20of%20the%20Art%20High-Performance%20Reverse%20Proxy%20Presentation.pdf>

**Google: 400ms extra latency**

0.44% decrease in searches per user

# Google: 400ms extra latency

0.44% decrease in searches per user

0.76% after six weeks



# Google: 400ms extra latency

0.44% decrease in searches per user

0.76% after six weeks

0.21% decrease after delay removed

# Bing

	<i>Distinct Queries/User</i>	<i>Query Refinement</i>	<i>Revenue/User</i>	<i>Any Clicks</i>	<i>Satisfaction</i>	<i>Time to Click (increase in ms)</i>
50ms	-	-	-	-	-	-
200ms	-	-	-	-0.3%	-0.4%	500
500ms	-	-0.6%	-1.2%	-1.0%	-0.9%	1200
1000ms	-0.7%	-0.9%	-2.8%	-1.9%	-1.6%	1900
2000ms	-1.8%	-2.1%	-4.3%	-4.4%	-3.8%	3100

- Means no statistically significant change

10M

# Mobify

100ms home load => 1.11% delta in conversions

# Mobify

100ms home load => 1.11% delta in conversions

100ms checkout page speed => 1.55% delta in conversions

some random uncached query



**All**

Images

News

Maps

Videos

More ▼

Search tools

---

About 71,100 results (0.59 seconds)

10M

To hit 500ms round trip...

10M

...budget ~10ms for search

# Larger service

Latency == \$\$\$

Need to handle more than  $\frac{1}{2}$  QPS

10M



# Use an index?

Salton; The SMART Retrieval System (1971); work originally done in early 60s

# 30 - 30,000 QPS

(we'll talk about figuring this out later)

<http://www.anandtech.com/show/9185/intel-xeon-d-review-performance-per-watt-server-soc-champion/14>

Haque et al.; Few-to-Many: Incremental Parallelism for Reducing Tail Latency in Interactive Services (ASPLOS, 2015)

10M

**10k; 10M; 10G;**  
(5kB per doc)

10B

$$5\text{kB} * 10\text{G} = 50\text{TB}$$

# Horizontal scaling

(use more machines)

# Easy to scale

(different documents on different machines)

# Horizontal scaling

$10\text{G docs} / (10\text{M docs} / \text{machine}) = 1\text{k machines}$

Redmond-Dresden: 150ms



# Horizontal scaling

10G docs / (10M docs / machine) = 1k machines

1k machines \* 10 clusters = 10k machines

“[With 1800 machines, in one year], it’s typical that 1,000 individual machine failures will occur; thousands of hard drive failures will occur; one power distribution unit will fail, bringing down 500 to 1,000 machines for about 6 hours; 20 racks will fail, each time causing 40 to 80 machines to vanish from the network; 5 racks will “go wonky,” with half their network packets missing in action; and the cluster will have to be rewired once, affecting 5 percent of the machines at any given moment over a 2-day span”

# Horizontal scaling

10G docs / (10M docs / machine) = 1k machines

1k machines \* 10 clusters = 10k machines

10k machines \* 3 redundancy = 30k machines

# Horizontal scaling

10G docs / (10M docs / machine) = 1k machines

1k machines \* 10 clusters = 10k machines

10k machines \* 3 redundancy = 30k machines

30k machines \* \$1k/yr/machine = \$30M / yr

2x perf: \$15m/yr

2% perf: \$600k/yr

# Horizontal scaling

$10\text{G docs} / (10\text{M docs} / \text{machine}) = 1\text{k machines}$

$1\text{k machines} * 10 \text{ clusters} = 10\text{k machines}$

$10\text{k machines} * 3 \text{ redundancy} = 30\text{k machines}$

$30\text{k machines} * \$1\text{k/yr/machine} = \$30\text{M} / \text{yr}$

Machine time vs. dev time

10B

# **Search Algorithms**



What's the problem again?

# Posting list

Algorithms: posting list

# I N D E X R E R U M P R Æ C I P V A R U M H V I V S A T L A N T I S.

<p><b>A.</b></p> <p><b>A</b> Deps. qui semel accensus exingit nequit, 48  <i>A.</i>s, quale Corinthiacum, apud Sinas unde? 22          Aëdiones, 56          Aëris qualitas mirabilis, 56          Aëthis maris singularis, 88          Aggeres ingentes, 47          Agriculturae dediti Sinæ, ejusdem ceras habent regulas, <i>ibid.</i>          Amnis aque rubræ, 3          Anachoretæ Sinenfes, 69, 98          Anatum copia, earumque edocatio mira, <i>ibid.</i>          Animalia longioris vitæ amant Sinæ, cur? 45          Anian fretum, 104          Animal affucillimum, cui sanum dicitur, 101          Annonæ regis Prorex, 81          Antiquitatum studiis Sinæ, ac annates, 22          Antiquissima urbis in orbe rudera, 28          Aqua mirabilis naturæ, 49          Aqua ad Ferramentæ accenda apta, 52              lycæ calida, æstate frigida, 31              ingenia augens, 127              nigra, 70              odorifera, 64              quæ ex vestibus omnes maculas eximit, 67              viridis, 62              optima ad potum ex oryza conficiendum, 63          Aquæ depledyræ antiquissimus usus, 9          Aquila lignum, 109          Arbor farinæ ferax, 113          Arbor ingenis magnitudinis, 52, 91          Architecra Sinica qualis? 5, 24          Arelada M. P. Veneti quæ putetur? 10, 80, 87, &amp; alibi.          Arcus triumphales egregii, 113          Arca, 26          Arena pro vitris, ac gemmis poliendis optima, 118          Argentum vivum ubi? 67          Argentum Chemicæ arte verum, 7          Arma vilipendunt Sinæ, 4          Arsitices egregii, 91          Arandines, earumque plurimus usus, 59          Arandines notabiles, 107          Arandines maxime magnitudinis, 17          Asbeston, 1          Asie nobilitas, <i>ibid.</i>          Asis extrema nobilitas, 4          Astronomia, 81          Astronomen Sinicum quale? 112          Aves vericoloris, ac carum usus, 53          Avicula rara &amp; admiranda, 26, 128          Aviculae ex arborum foliis natæ, 4          Aurum merx, non pecunia, 34, 44, 63, 67, &amp; alibi.          Auxifera regio, 34, 59              caligigitur ex arenis flaviorum, 98              B. 28</p> <p><b>B.</b></p> <p><b>B</b> Ajuli optimi sunt Sinæ, 28          Balneæ quo pacto, ac ubi capiuntur? 32, 54          Bellata in orbe primum, 85          Bibliotheca insignis, 109          Bombyces quo pacto alantur, 3, 85, 89, &amp; alibi.          Brasilium lignum, 3          Byssi fœci feracis abundantia, 3              ejus inventio ac usus antiquissimus, <i>ibid.</i>              à Sinis ad alias pervenit nationes, 85          Byssus prima, ac secunda qualis? 41          Byssus in arboribus ulro nascens, 23</p> <p><b>C.</b></p> <p><b>C</b> Ambala ubi? 23, 43, &amp; alibi.          Canales magnifici arte excavati, 108          Canes extra aquam lapidescunt, 18          Canes, equosque vocant Sinæ, 3          Cangygu quid? 22          Caravane quæ ad Sinas veniant, 41          Caracay ubi? 29          Carbo fossilis, 53, 114          Carbunculi, 3          Carnium, ac rerum necessarium copia, 69          Castellane puces famis tempore adhibetur, 11, 31, 22          Castanea notabilis, 43          Catayum quid? &amp; ubi? 93          Catara à Sinarum qualis? 93          Caudicis lapidescunt unde? 7, 122          Cavata, ac modestia mulierum Sinenium, 63          Cedras notabilis, <i>Sinæ.</i></p>	<p>Cera albissima à vermiculis elaborata, 60          Chia folii descriptio, 83          quid sit, ejus qualitas ac usus, <i>ibid.</i>          Chiamerum Sinenium usus admirabilis, 115          Chienica ar Sinus frequens, 8          Chienica artis antiquitas, 56, 77          Chienicae artis antiquitas, 56, 77          Chienicae artis antiquitas, 85, 86          Chienica Provinciae descriptio, 110          Chifang herba mirabilis, 35, 97          Christiania lex olim apud Sinas, 3          relictur à Societate Iesù, 74, 79, 98, &amp; alibi.          Clivata extrema Asia, 112          quando Sumatrag ingreditur, 69          Cibos qui ratione fumant Sinæ, 18          Cie, live sandrachra gummi Sinenium præclarissimum, 118          Cinis loco falsi adhibetur, 94          Cimabaris, 37, 61, 69          Cinnamonum, 99          Certas destruitur in qua filius patrem occidit, 99          Civitatum in Sinâ numerus, 4          Clepydrarum usus, 9          Clinata extrema Asia, 2          Cocenobium magnificum, 44, 67, 88, 105, &amp; alibi.          Compositio corporis seu modestia Sinarum, 7          Constellationes Sinicæ quales, 13, 14          Continentiam suspiciunt, ac laudant, 6, 7, 81, 108          Corallium Sinenium apud Sinas, 109          Corea quando habitari cepta à Sinis, 131, 132          Corea est tributaria, 132          Corea Sinis tributaria, <i>ibid.</i>          in octo dividitur Provincias, <i>ibid.</i>          Corporis lineamenta, ac Sinarum habitus, 4          Cosmographia, ac terrarum mappa, antiquæ apud Sinas, 31          Croci luminis descriptio, 11          cur ita dictus? <i>ibid.</i>          Croceus piscis avis mirabilis naturæ, 107          Crocodilli fontes ac intones distinguunt, 113          Chronologia Sinicæ compendium, 12          Cubitus Sinicus quantum? 12          Cugui M. P. Veneti, quæ? 90          Cingitati philosophi æstimatio quanta apud Sinas, 10          ejus doctrina breviter explicatur, 44          ejus patria quæ, 98, 99          Cingitatus interpres optimus, 3          Cuprum candidum, D. 2          Descriptio varia extreme Asiae, 4          Divitiae, ac merces ingentes, 8          Doctoris Pauli laus, 4          Domus, ac habitatio Sinarum qualis? 5, 28          Duellum de flore inter duo sidera, urbi nomen dedit, 20          Dynastæ absoluti domini in Sinis alique, E. 24          Leplantes vigiles, &amp; excubiitores, 28, 69, 72, 73, 78, 114          Tempora nobilissima, 13          Epochæ Sinici Calendarii quæ? 13          Equus aquarius, 133          Error de ortu Japoniæ, 133          Errores de Sinarum regno apud Europæos, 3          Ethica philosophia Sinica, 5          Extrema Asia quid? 1          ejus nomen, <i>ibid.</i>          Terminus, E. 1</p> <p><b>E.</b></p> <p><b>E</b> Abellis Sinarum, 37, 39, 49, 60, 64          Falcones, 37          Fanum notabile, 43, 67, 96, &amp; alibi.          Fanum igni dicitur, 92          Fanum ex quo forma petunt, 92          Fe doctrinæ adulti Sinæ, 123          unde ad Sinas illata, 124          Febris certo anni tempore innoxia, reliquo lethalis, 56          Fese animal homines vorat ridendo, 20          Felix raro, ac in deliciis, 22          Ferrum leutum, 103          Ferrum fundunt, &amp; effingunt vasa, 4, 11          Ferret lapides, 3          Ferrea turris ingens, 80          Ferrea columna, 82          Ficus Europææ, 124          Fidelis Christiani propagatio, 35, 74, 78, 94          Fingula Chia adhibetur, 93          Figura regni Sinarum quadrata, 20          Flos suavissimus Mogurum, 63</p>	<p>Meu-</p>
---	---	-------------

See <http://nlp.stanford.edu/IR-book/> for implementation details

HashMap[term] => list[docs]

Algorithms: posting list

# Bloom filter

Algorithms: bloom filter

# BitFunnel

Algorithms: bloom filter

# What about an array?

Algorithms: bloom filter





# How many terms?

Algorithms: bloom filter



**Crista Lopes**

@cristalopes

 Follow

The largest C++ file we found in GitHub has 528Mb, 57 lines of code. Contains the first 50,847,534 primes, all hard coded in an array.

RETWEETS

1,491

LIKES

1,882



4:53 PM - 15 Sep 2016



1.5K



1.9K



Algorithms: bloom filter

One site has 37B primes

Algorithms: bloom filter

GUIDs, timestamps, DNA, etc.

Algorithms: bloom filter

# Why index that stuff?

Algorithms: bloom filter

GTGACCTTGGGCAAGTTACTTA  
ACCTCTCTGTGCCTCAGTTTCCT  
CATCTGTAAAATGGGGATAATA

Algorithms: bloom filter



Most terms aren't in most  
docs => use hashing

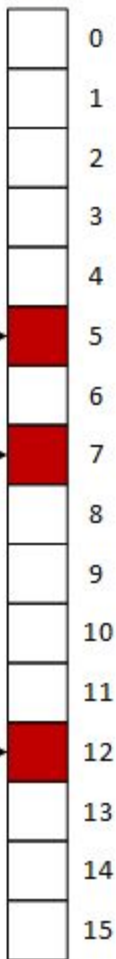
Algorithms: bloom filter



# Bloom Filters

Algorithms: bloom filter

SET



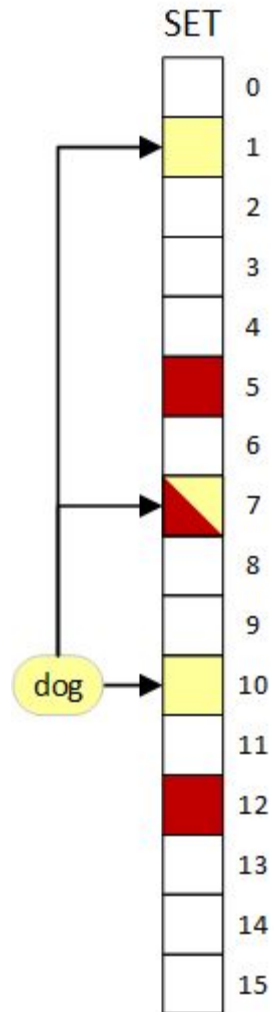
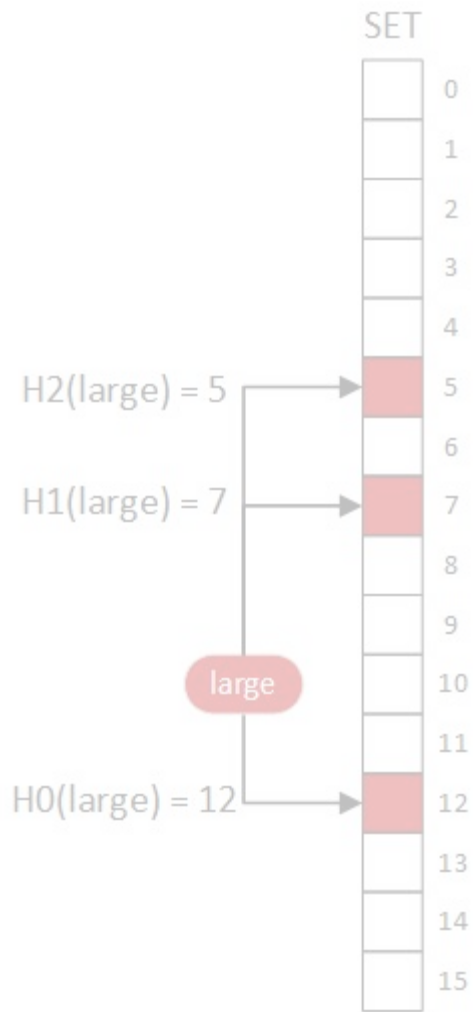
$H_2(\text{large}) = 5$

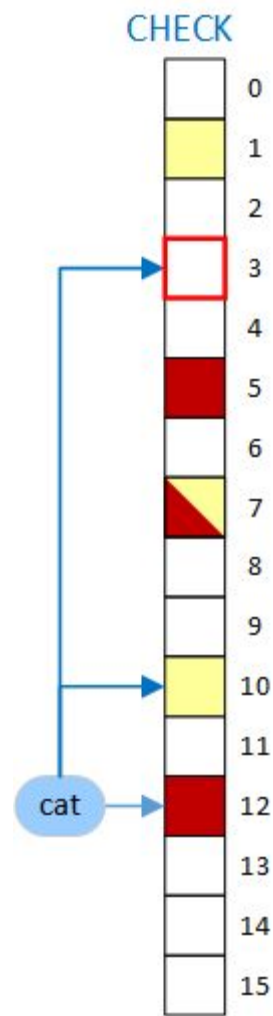
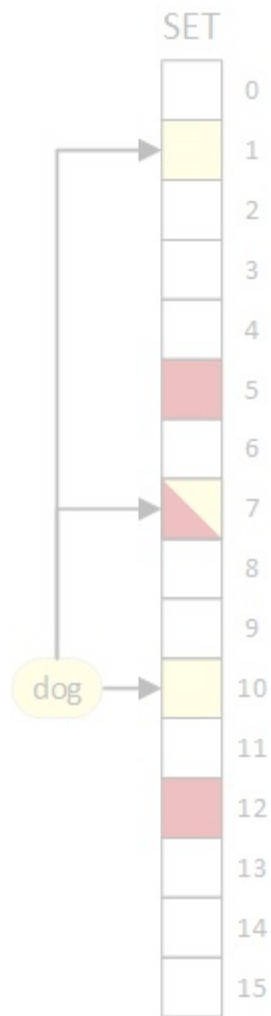
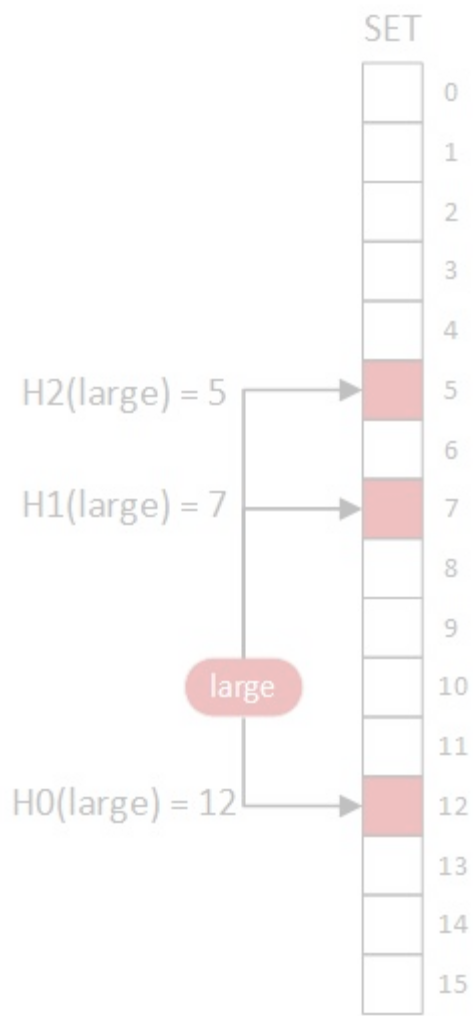
$H_1(\text{large}) = 7$

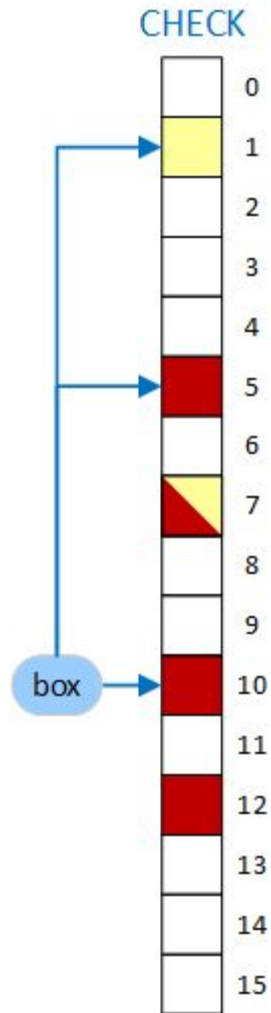
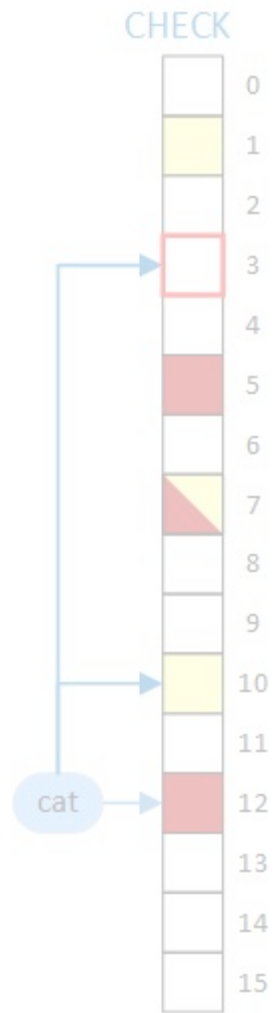
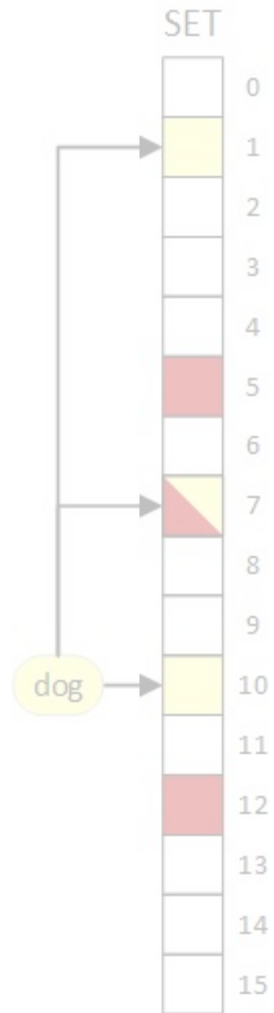
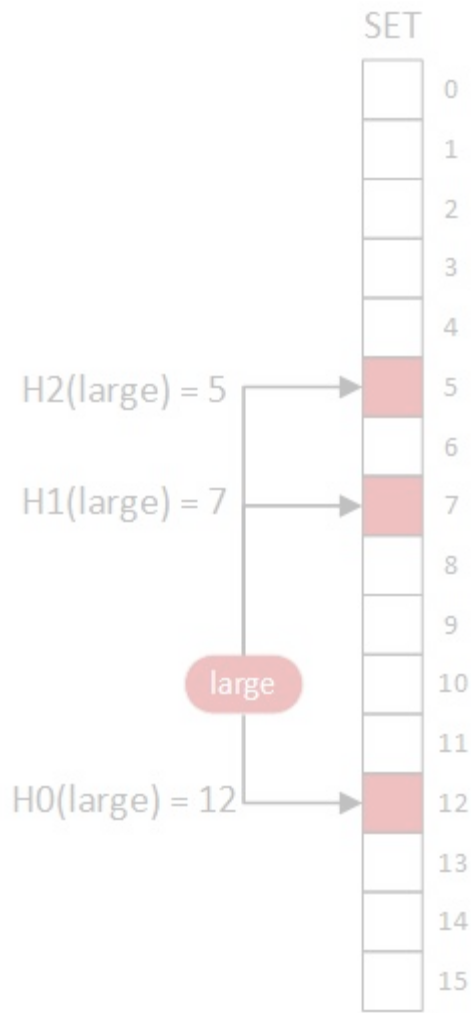
large

$H_0(\text{large}) = 12$

0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15







# Probability of false positive?

Algorithms: bloom filter

(assume 10% bit density)

**1 location: .1 = 10% false positive rate**

Algorithms: bloom filter

(assume 10% bit density)

1 location:  $.1 = 10\%$  false positive rate

2 locations:  $.1 * .1 = 1\%$  false positive rate

Algorithms: bloom filter



(assume 10% bit density)

1 location:  $.1 = 10\%$  false positive rate

2 locations:  $.1 * .1 = 1\%$  false positive rate

3 locations:  $.1 * .1 * .1 = 0.1\%$  false positive rate

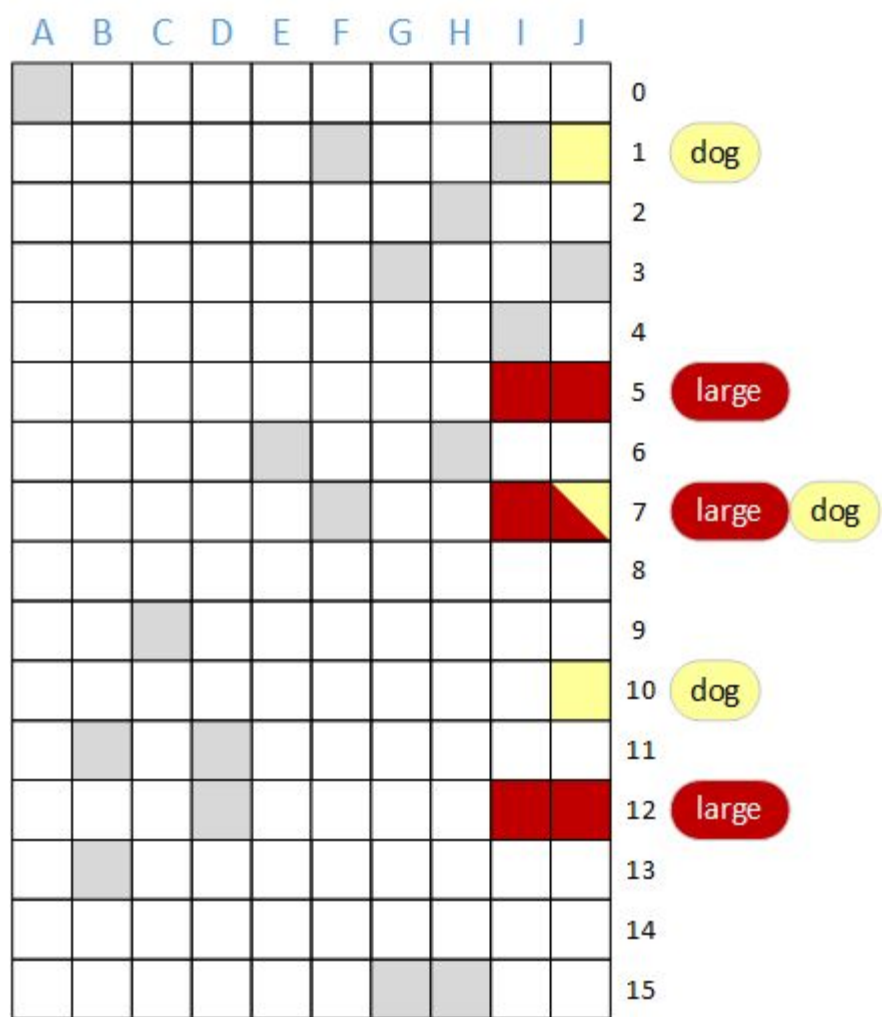
Algorithms: bloom filter

Linear cost  
Exponential benefit

Algorithms: bloom filter

# Multiple Documents Multiple Bloom Filters

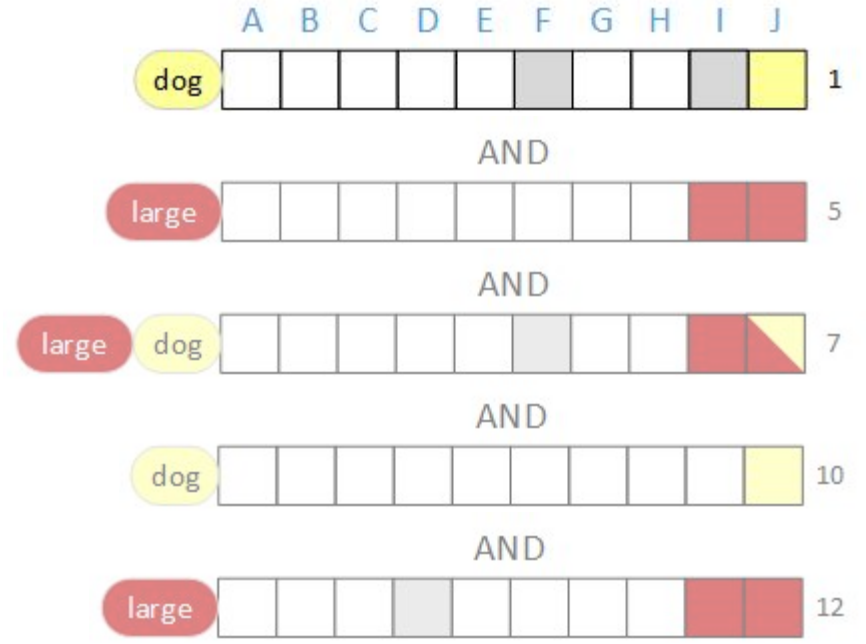
Algorithms: bloom filter



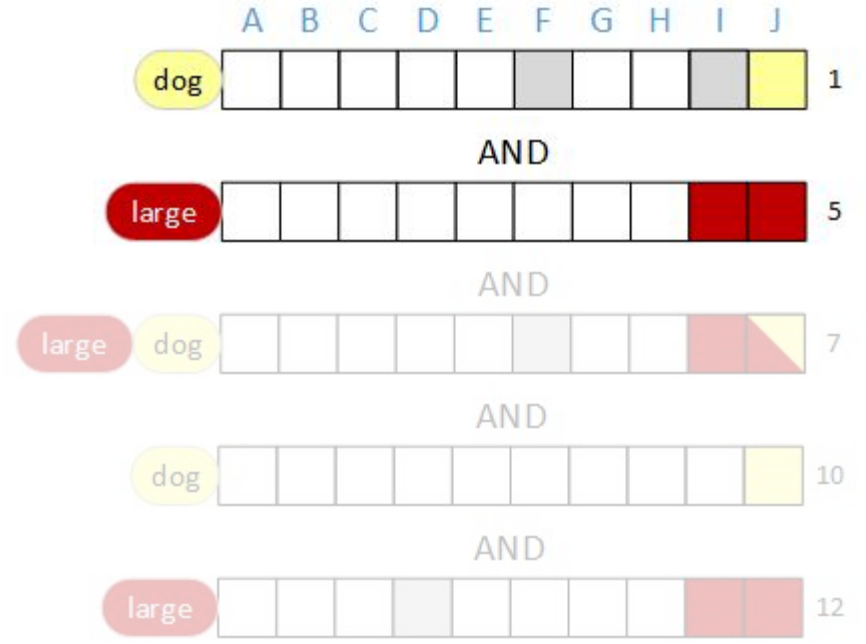
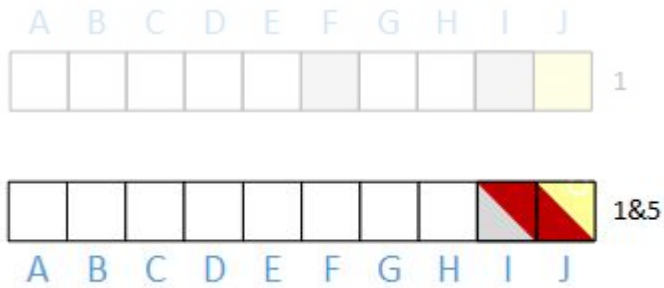
# Do comparisons in parallel!

Algorithms: bloom filter



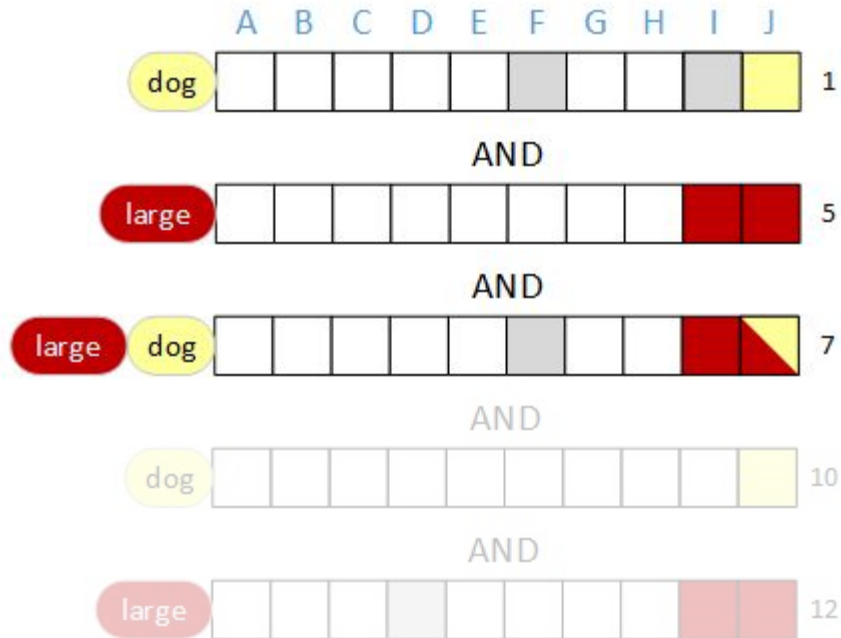
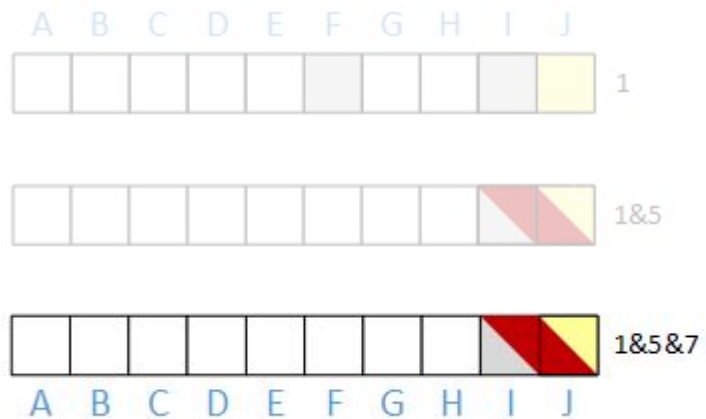


Algorithms: bloom filter

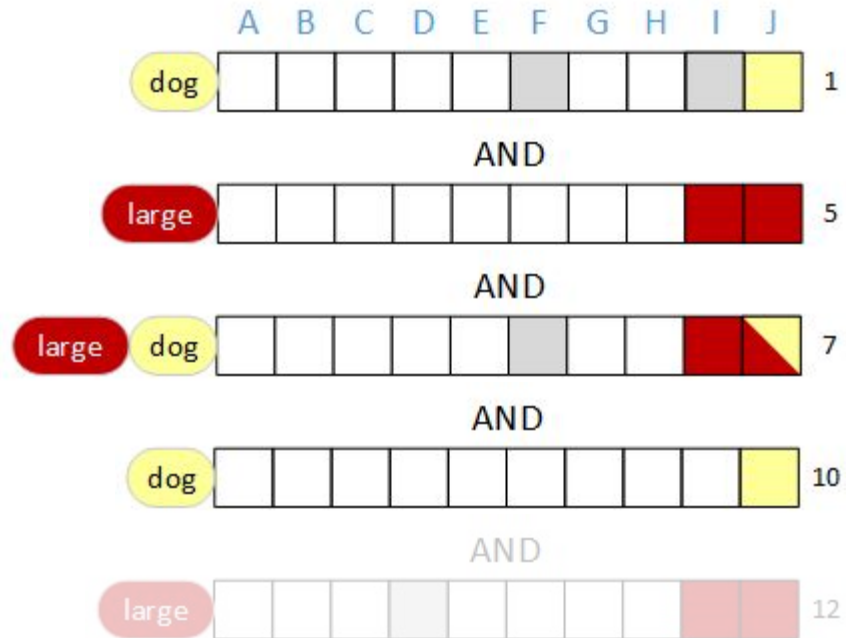
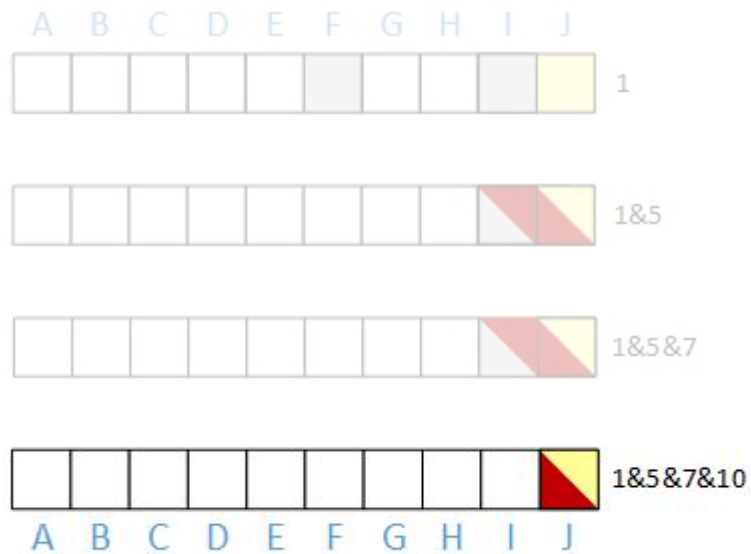


Algorithms: bloom filter

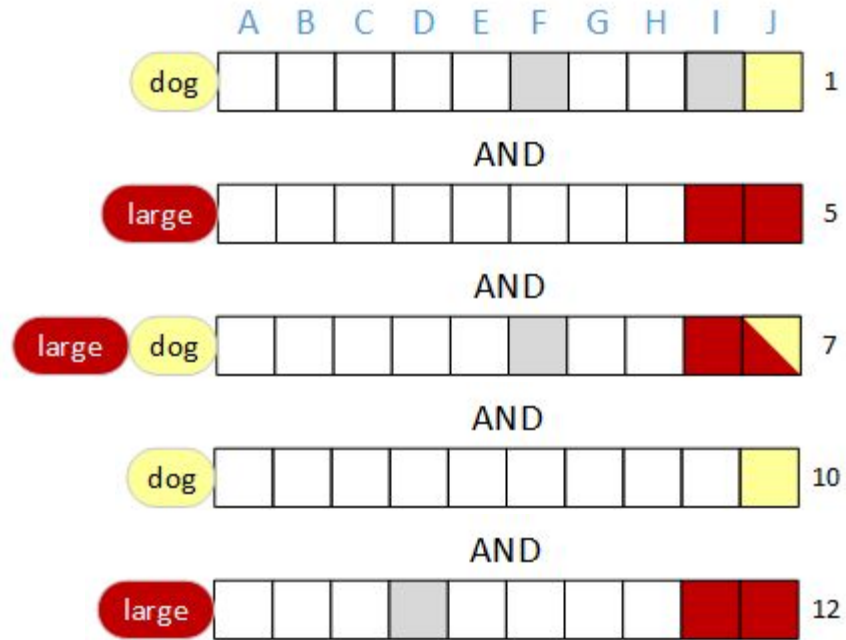
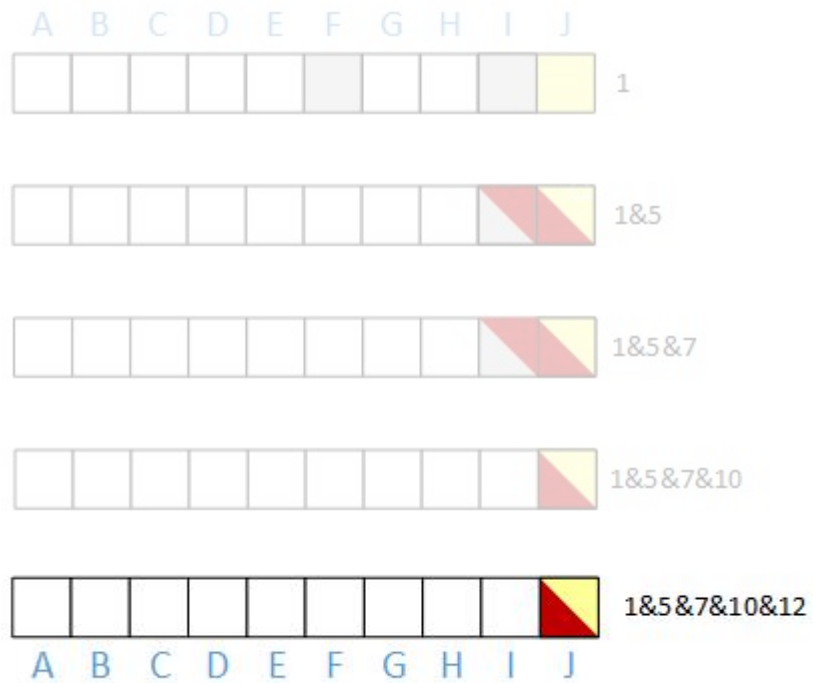




Algorithms: bloom filter



Algorithms: bloom filter



Algorithms: bloom filter

K L M N O P Q R S T

										0
										1
										2
										3
										4
										5
										6
										7
										8
										9
										10
										11
										12
										13
										14
										15

dog

large

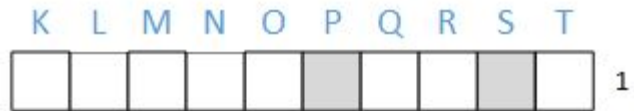
large

dog

dog

large

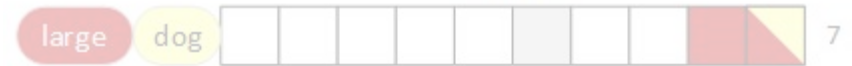




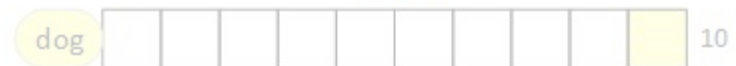
AND



AND



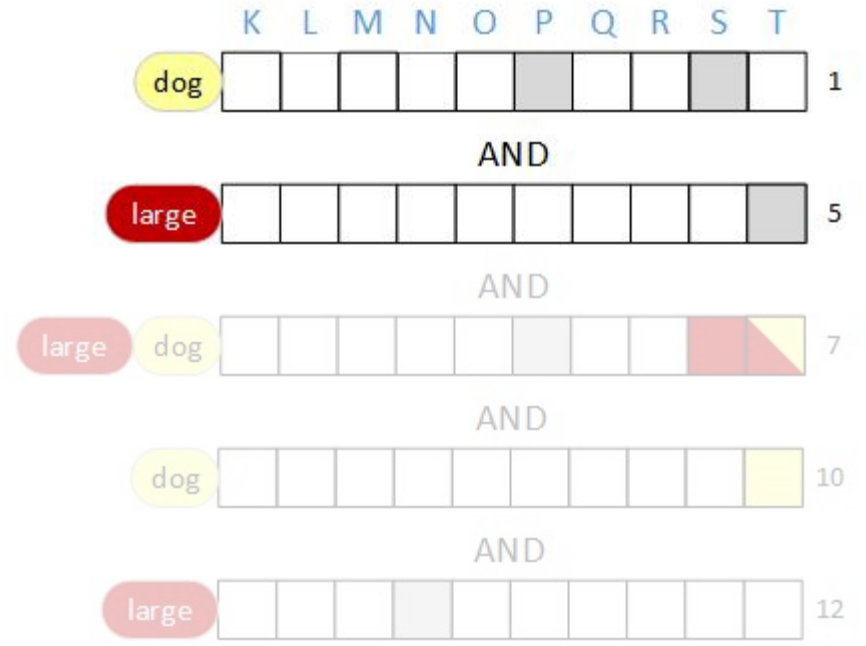
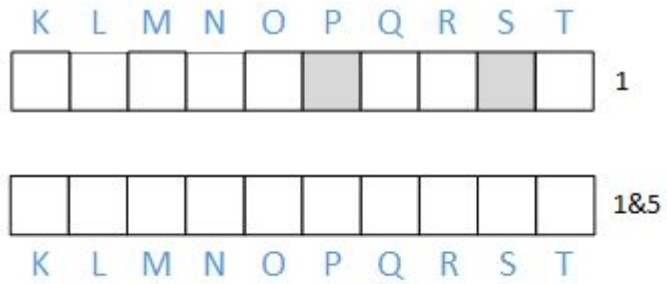
AND



AND



Algorithms: bloom filter



Algorithms: bloom filter

**How do we estimate perf?**



# Cost model

## Number of operations

Perf estimation

**512-bit “blocks”**  
(pay for memory accesses)

Perf estimation

How many memory accesses  
per block?

Perf estimation

$$\begin{aligned}
\mathbf{P}\{X > i\} &= \mathbf{P}\{\text{the accumulator at level } i \text{ is not all zeros}\} \\
&= \mathbf{P}\left\{\bigcap_i \neq \vec{0}\right\} \\
&= 1 - \mathbf{P}\left\{\bigcap_i = \vec{0}\right\} \\
&= 1 - \prod_{j=1}^{512} \mathbf{P}\left\{j^{\text{th}} \text{ bit in } \bigcap_i \text{ is } 0\right\}
\end{aligned}$$

<http://bitfunnel.org>

Perf estimation

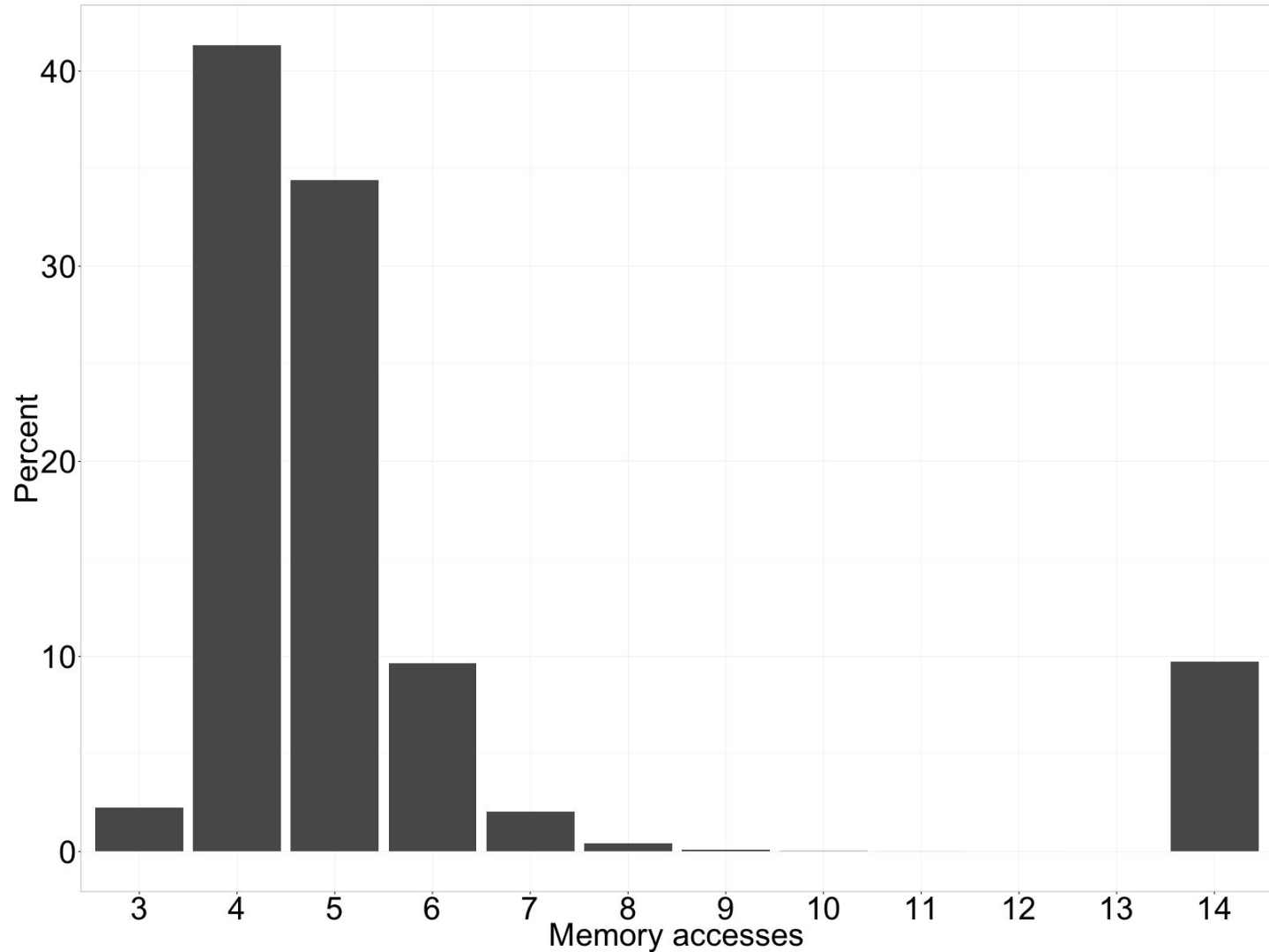
Conditioning on the whether document  $j$  is a match:

$$\begin{aligned} \mathbf{P} \left\{ j^{th} \text{ bit in } \bigcap_i \text{ is } 0 \right\} &\stackrel{=}{\geq} \begin{cases} 0, & j^{th} \text{ document is a match} \\ 1 - (d - s)^i, & \text{otherwise} \end{cases} \\ &\stackrel{=}{\geq} \begin{cases} 0, & \text{w.p. } s \\ 1 - (d - s)^i & \text{w.p. } 1 - s \end{cases} \\ &\geq (1 - (d - s)^i) \cdot (1 - s) \end{aligned}$$

Returning to the CDF of  $X$ :

$$\mathbf{P}\{X > i\} \leq 1 - \left( (1 - (d - s)^i) \cdot (1 - s) \right)^{512}$$

Perf estimation



# Why do we have so many rows?

Term rewriting

Perf estimation

# Term Rewriting

“Large yellow dog”

Perf estimation



# Term Rewriting

“Large yellow dog” ||

“Golden Retriever”

Perf estimation

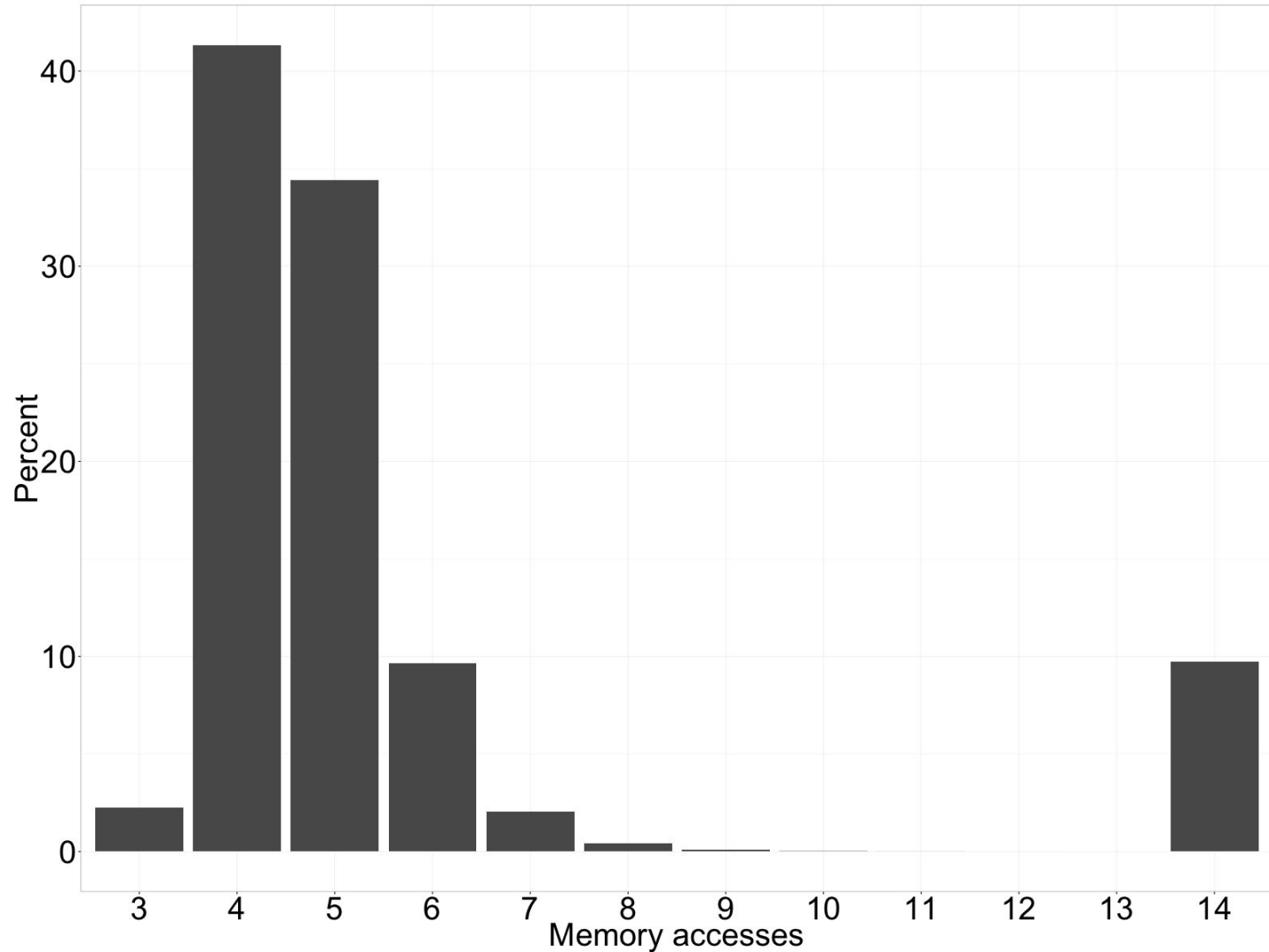
# Term Rewriting

“Large yellow dog” ||

“Golden Retriever” ||

“Old Yeller” ||

Perf estimation



# Expected performance?

Perf estimation

10 M docs / 512 bits per block = 20k “blocks”

Perf estimation

10 M docs / 512 bits per block = 20k “blocks”

20 k-blocks \* 5 transfers per block = 100 kT

Perf estimation

10 M docs / 512 bits per block = 20k “blocks”

20 k-blocks \* 5 transfers per block = 100 kT

25 GB/s / 512 bits per transfer = 390 MT/s

Perf estimation

10 M docs / 512 bits per block = 20k “blocks”

20 k-blocks \* 5 transfers per block = 100 kT

25 GB/s / 512 bits per transfer = 390 MT/s

**390 MT/s / 100 kT = 3900 QPS (with rounding)**

Perf estimation



# Actual performance?

Perf estimation

Actual performance  $\sim$  similar

Perf estimation

# Small factors

Perf estimation

# Large factors

Perf estimation

# Ranking results

Perf estimation

# Ingestion

(faster than querying)

Perf estimation

# Ingestion is just setting bits

Perf estimation

# Hierarchical bloom filters

Perf estimation



# Complicating issues?

Perf estimation



**Conclusions?**

# False conclusions

Search is simple

# False conclusions

Search is simple

Bloom filters are better than posting lists

# False conclusions

Search is simple

Bloom filters are better than posting lists

You can easily reason about all performance

**Conclusions!**

You can reason about perf



It's often just arithmetic

# Acknowledgements

Thanks to Leah Hanson, Mike Hopcroft, Julia Evans, Hari Angepat, David Turner, Danielle Sucher, Ikhwan Lee, Tejas Sapre, Raul Jara, Rich Ercolani, Bert Muthalaly, Harsha Nori, Jeshua Smith, Bill Barnes, Gary Bernhardt, Marek Majkowski, Tom Crayford, Gina Willard, Laura Lindzey, Larry Marbuger, Siddarth Anand, Eric Lemmon, Tom Ballinger, and [Anonymous Reviewer] for feedback.

[bitfunnel.org/strangeloop](https://bitfunnel.org/strangeloop)

[github.com/bitfunnel/bitfunnel](https://github.com/bitfunnel/bitfunnel)

[danluu.com](https://danluu.com)

# Unused slides

(thar be dragons)

SLIDE FOR HOMEWORK. TODO: USE DIFFERENT TEMPLATE

Why are posting lists standard?

# Literature on alternatives

“ **Signatures files** were proposed in [23] and **shown to be inferior** to inverted indexing in [24]. “

“ Inverted indexes have been benchmarked as the most generalisable, and well performing structure (Zobel et al., 1998). **The experiments in this thesis are therefore conducted solely on an inverted index system.**”

“While this technique provides a relatively low computation overhead, studies by Zobel et al. [1998] have shown that inverted files significantly outperform signature files. **We will now focus the analysis on inverted files as it is generally considered to be the most efficient indexing method for most IR systems.**”

“The other two mechanisms are usually adopted in certain applications even if, recently, **they have been mostly abandoned in favor of inverted indexes because some extensive experimental results** [194] have shown that: Inverted indexes offer better performance than signature files and bitmaps, in terms of both size of index and speed of query handling [188]”

“Zobel et al. [16] compared inverted files and signature files with respect to query response time and space requirements. They found that the inverted files evaluated queries in less time than the signature files and needed less space. Their results showed that the **signature files were much larger, more expensive to construct and update, their response time was unpredictable, they support ranked queries only with difficulty, they did not scale well and they were slow**”

## Zobel et al., actual quotes

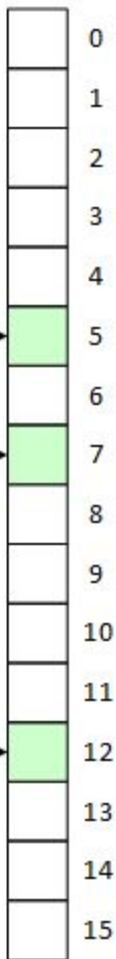
“Inverted file indexes with in-memory search structures **require no more disk accesses** to answer a conjunctive query than do bitsliced signature files.”

“One of the difficulties in the comparison of inverted files and signature files is that many variants of signature file techniques have been proposed, and **it is possible that some combination of parameters and variants will result in a better method.**”



Citations are lossy

SET

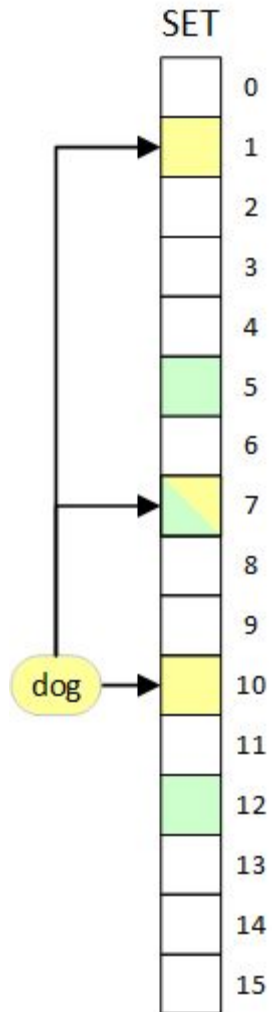
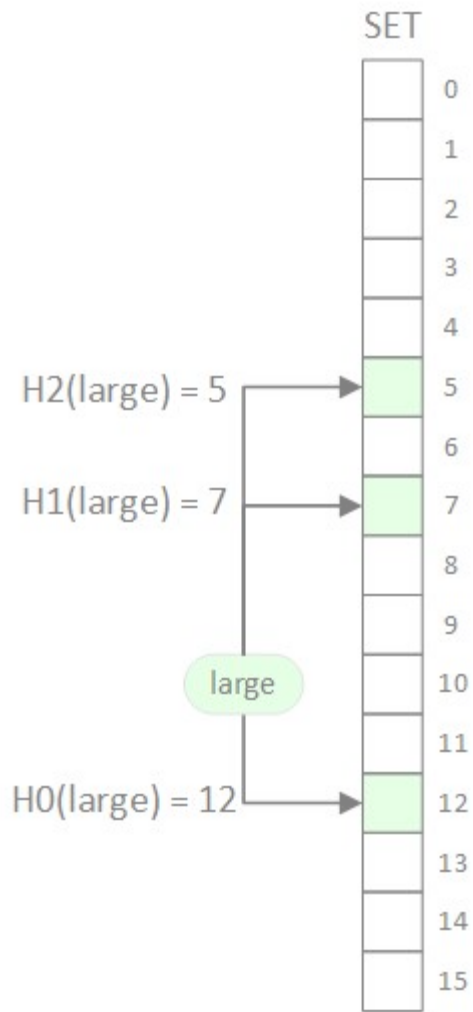


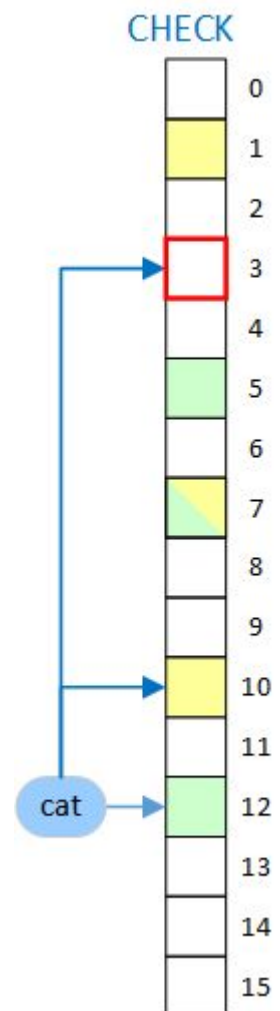
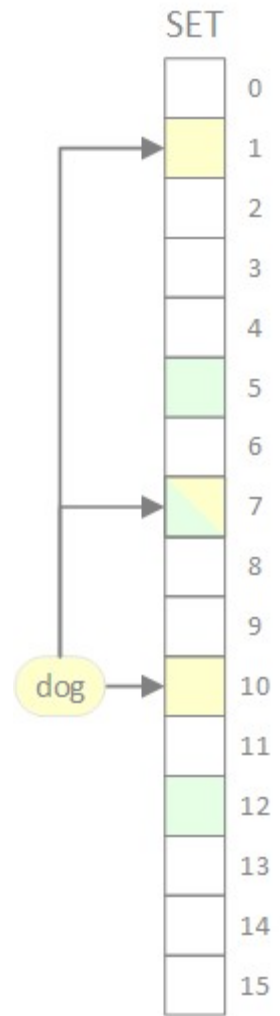
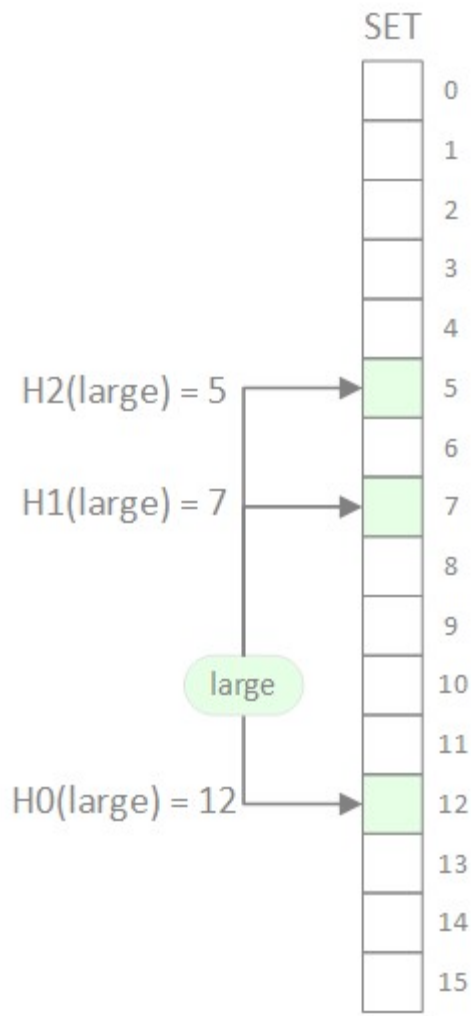
$H2(\text{large}) = 5$

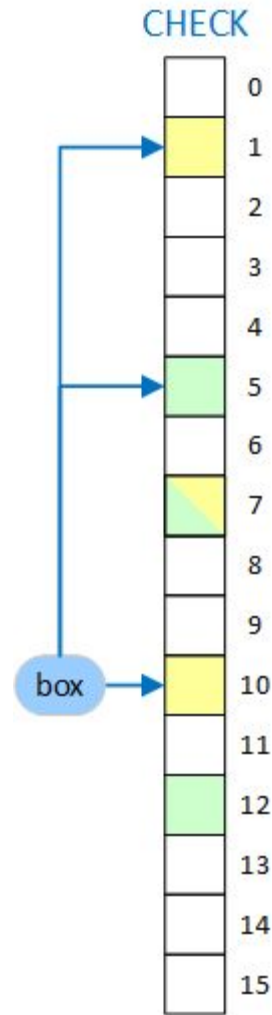
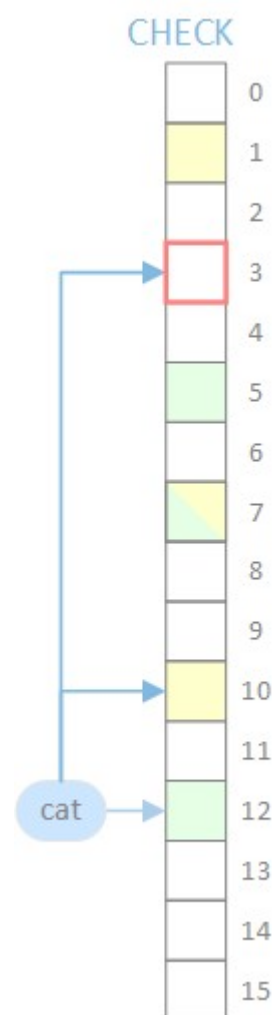
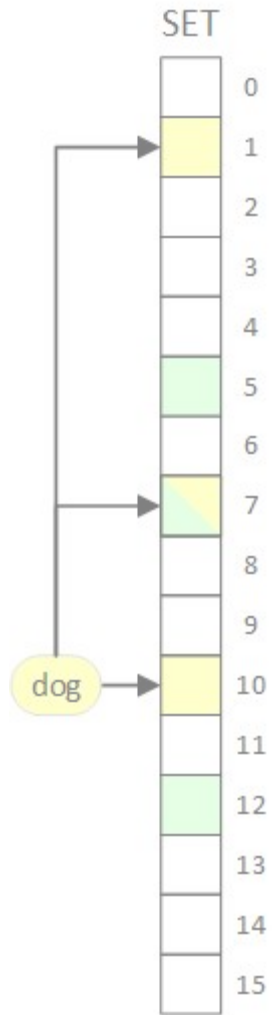
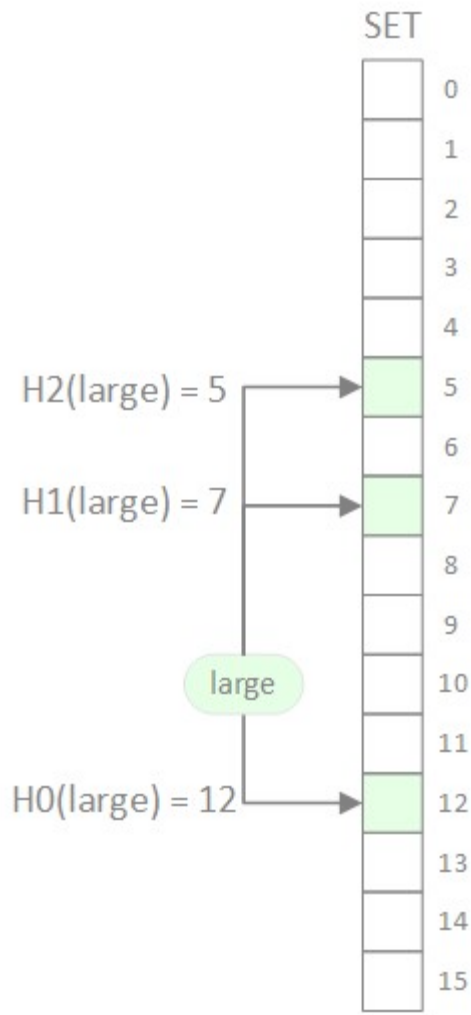
$H1(\text{large}) = 7$

large

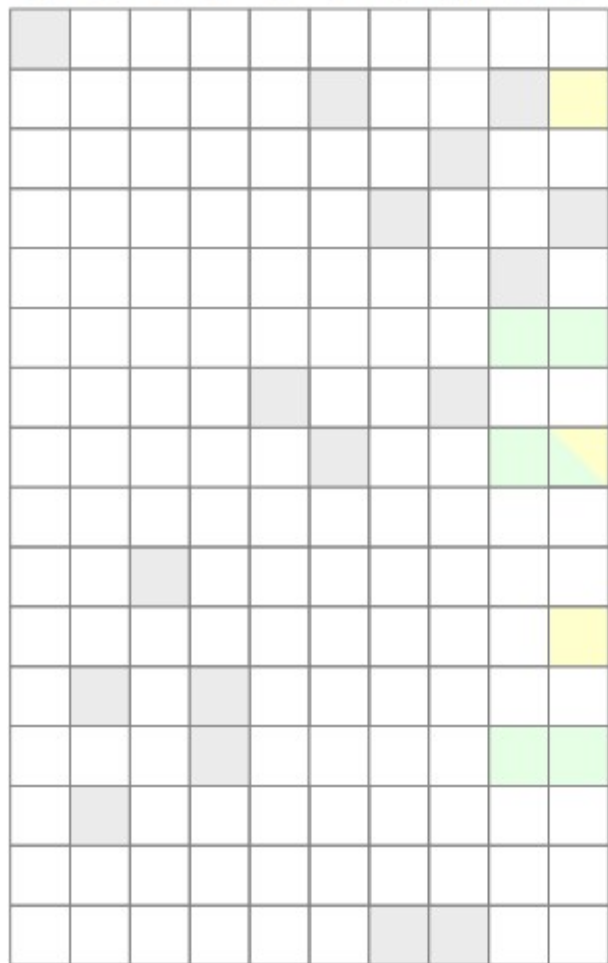
$H0(\text{large}) = 12$







A B C D E F G H I J



dog

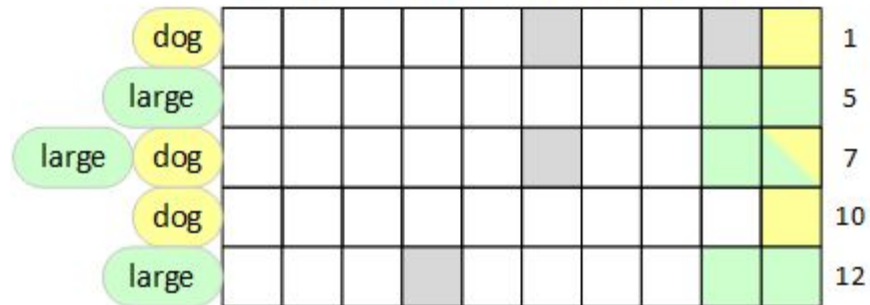
large

large

dog

large

A B C D E F G H I J



dog

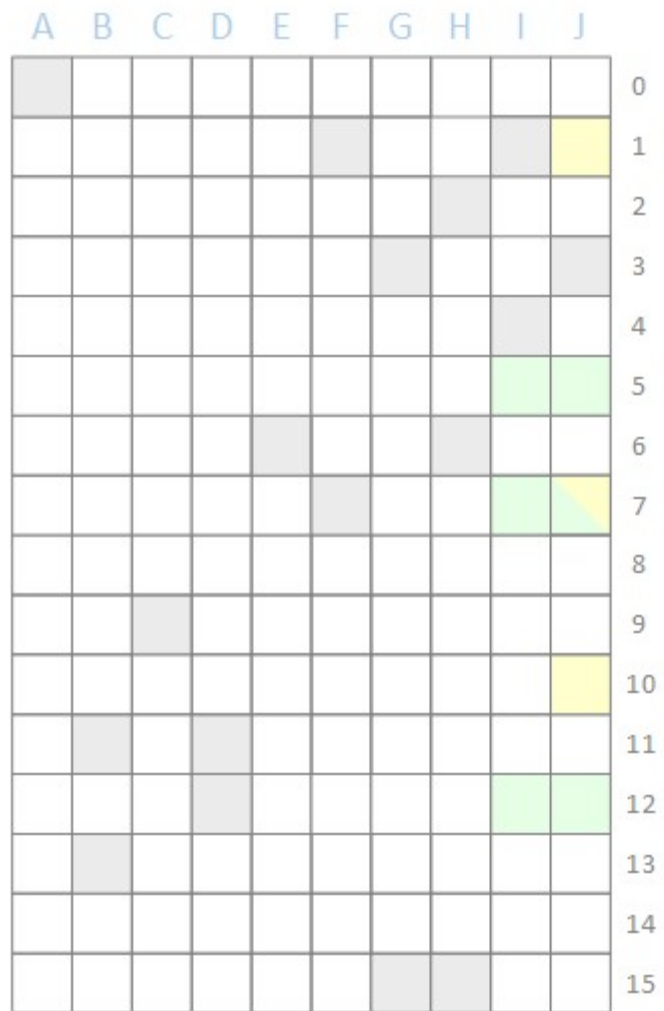
large

large

dog

dog

large



0

1 dog

2

3

4

5 large

6

7 large dog

8

9

10 dog

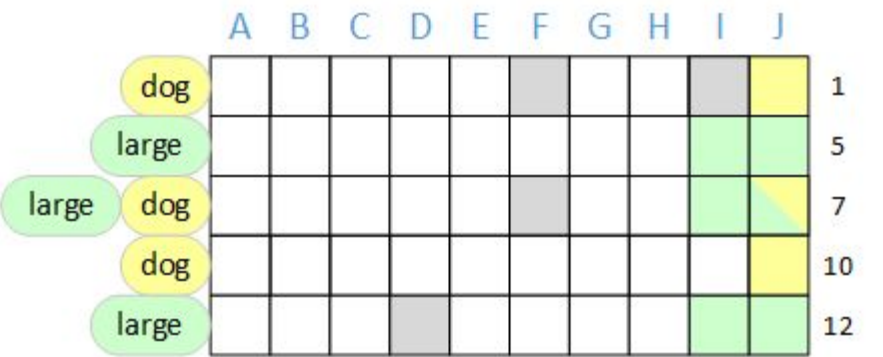
11

12 large

13

14

15



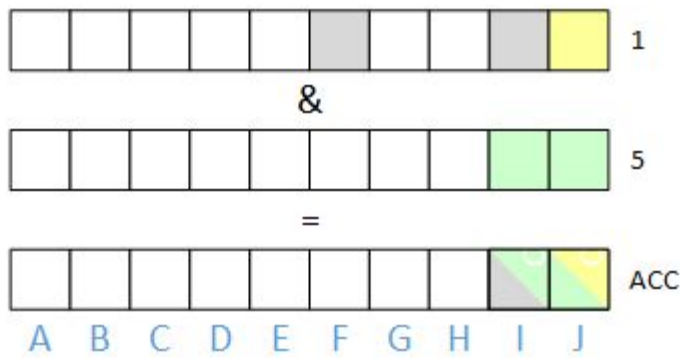
dog

large

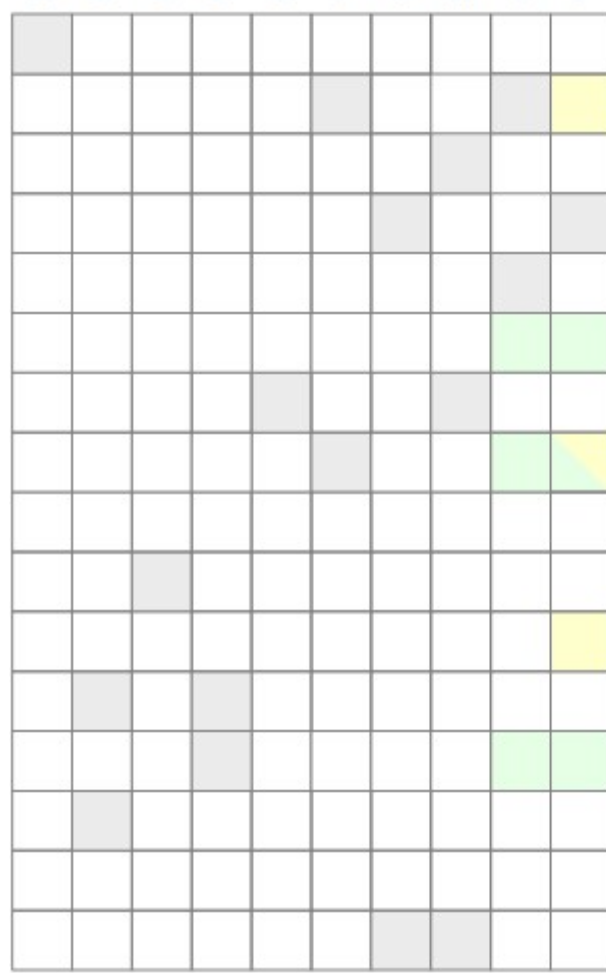
large dog

dog

large

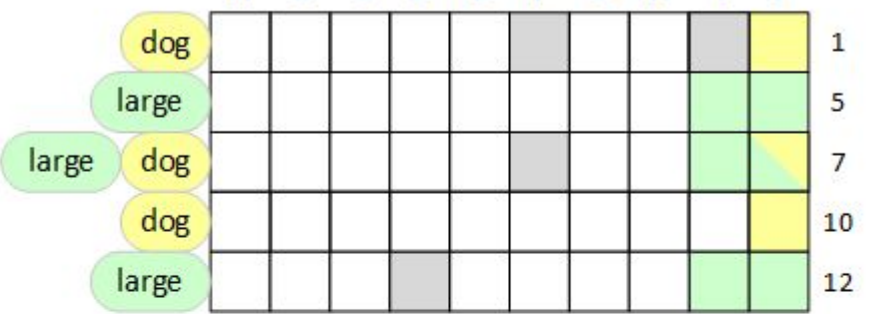


A B C D E F G H I J



0  
1 dog  
2  
3  
4  
5 large  
6  
7 large dog  
8  
9  
10 dog  
11  
12 large  
13  
14  
15

A B C D E F G H I J

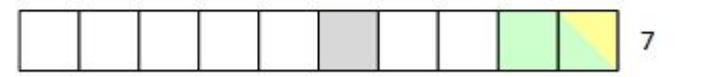


dog  
large  
large dog  
dog  
large

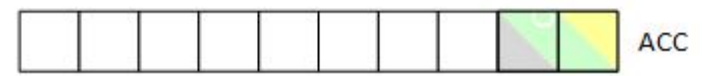
1  
5  
7  
10  
12



&

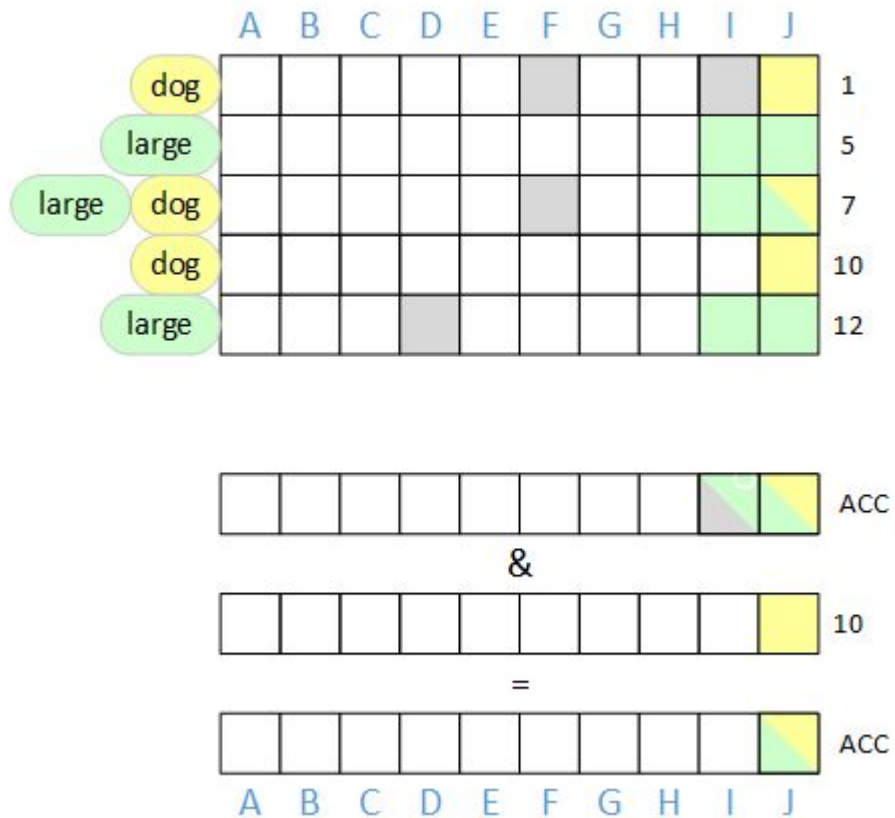
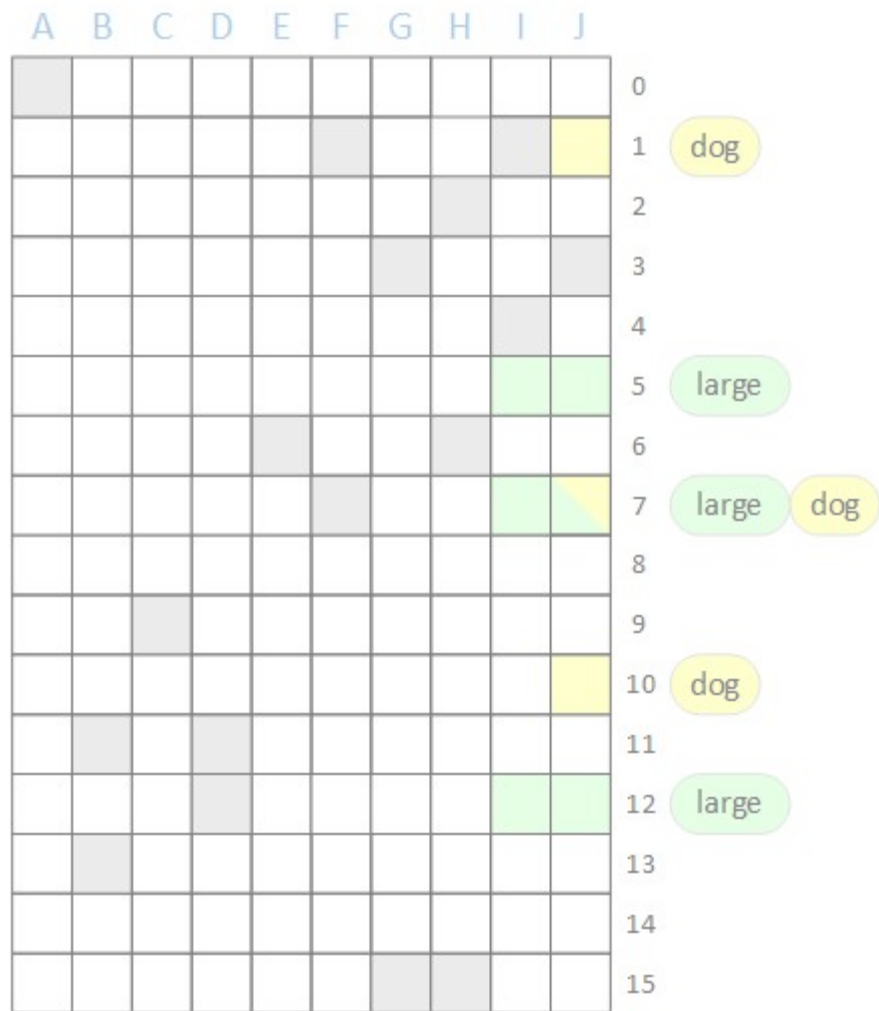


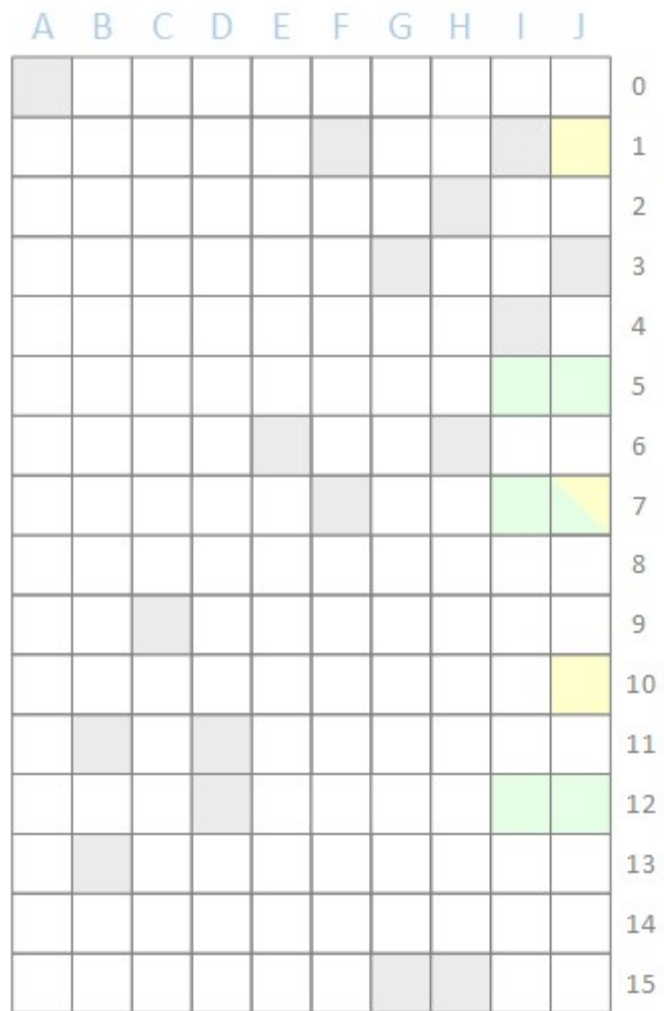
=



A B C D E F G H I J







0

1 dog

2

3

4

5 large

6

7 large dog

8

9

10 dog

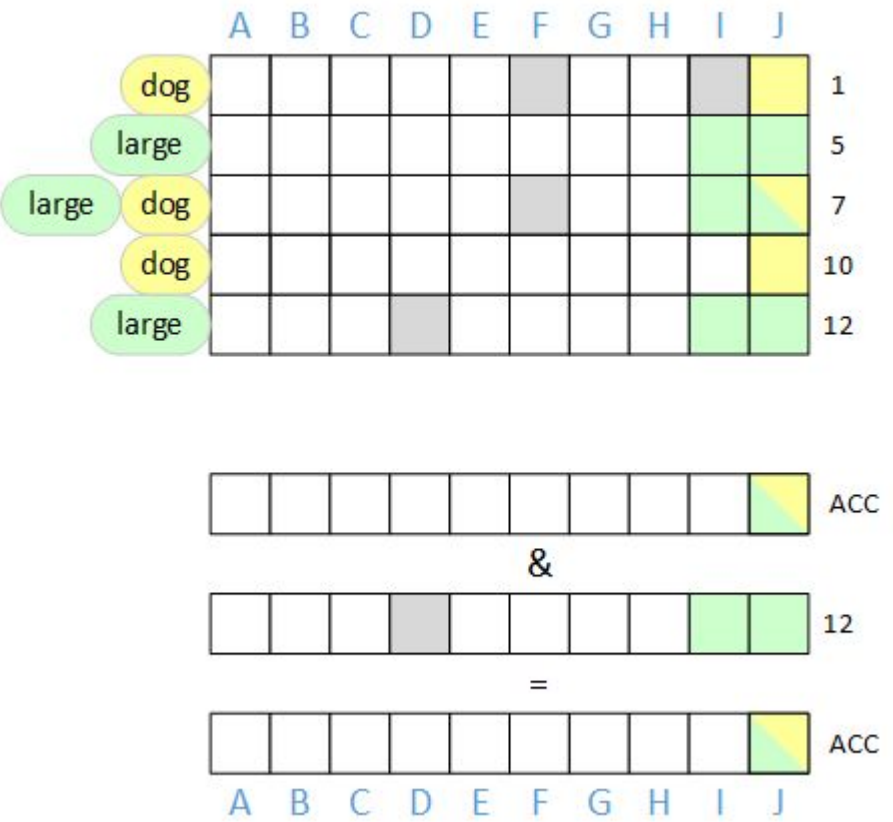
11

12 large

13

14

15



dog

large

large dog

dog

large

1

5

7

10

12

ACC

12

ACC

A B C D E F G H I J

A	B	C	D	E	F	G	H	I	J	
█										0
					█			█		1 dog
							█			2
						█				3
								█		4
									█	5 large
				█			█		█	6
					█					7 large dog
										8
		█								9
								█		10 dog
	█		█							11
			█							12 large
	█									13
										14
						█	█			15

A B C D E F G H I J

											0
											1
											2
											3
											4
											5
											6
											7
											8
											9
											10
											11
											12
											13
											14
											15

dog

large

large dog

dog

large

A B C D E F G H I J

											1
											5
											7
											10
											12

dog

large

large

dog

dog

large

											1
--	--	--	--	--	--	--	--	--	--	--	---

&

											5
--	--	--	--	--	--	--	--	--	--	--	---

=

											ACC
--	--	--	--	--	--	--	--	--	--	--	-----

A B C D E F G H I J

Search: why do we care?

$\$20\text{M}/\text{yr} * 2\% \text{ savings}$

$=$

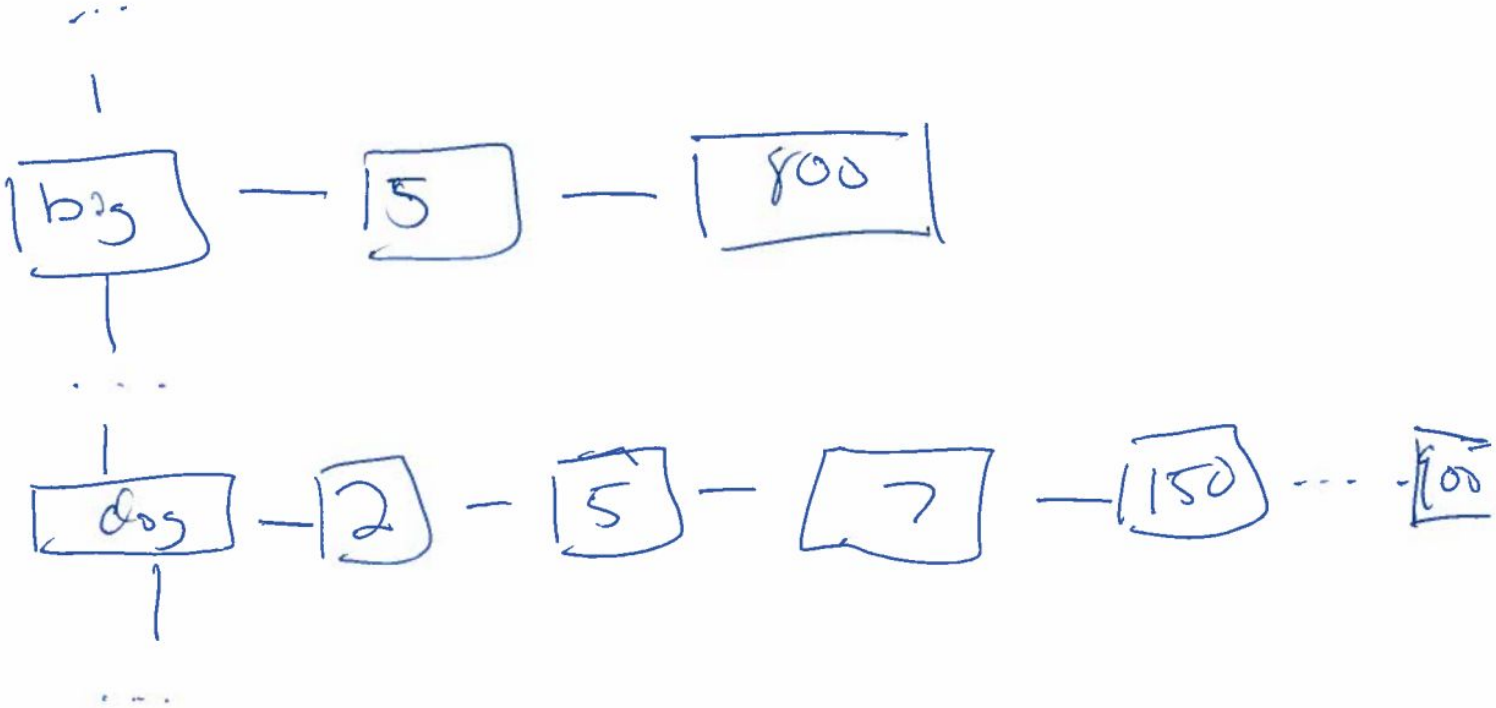
$\$400\text{k}/\text{yr}$

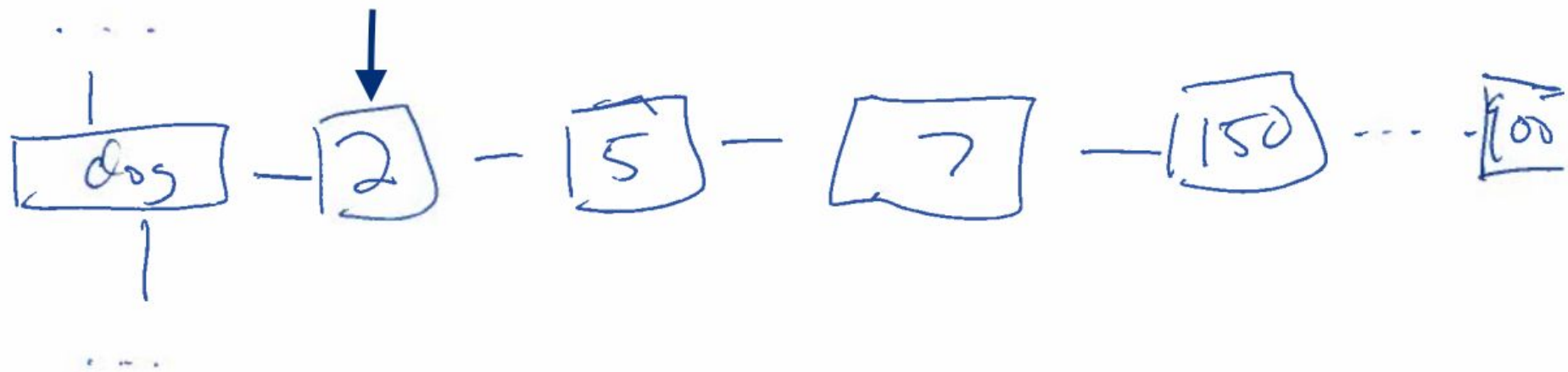
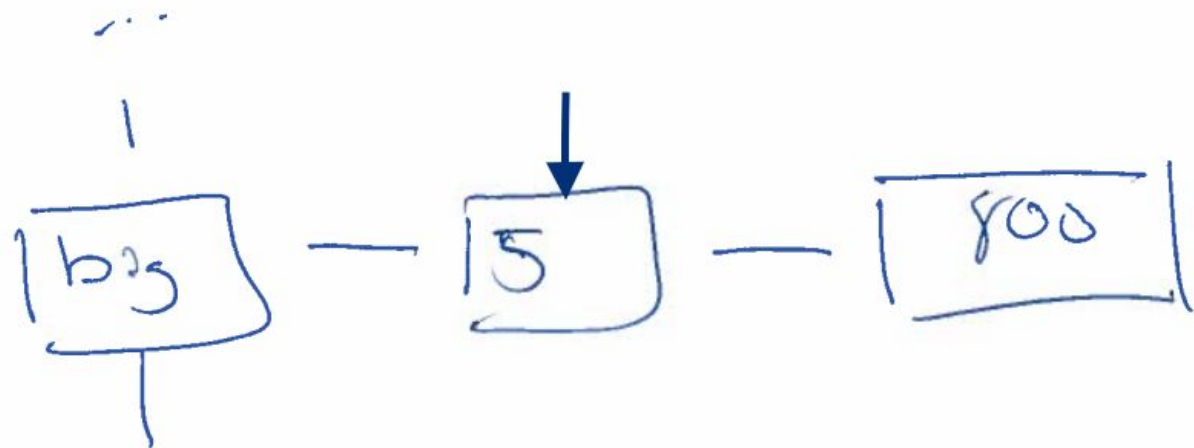
How things fit together

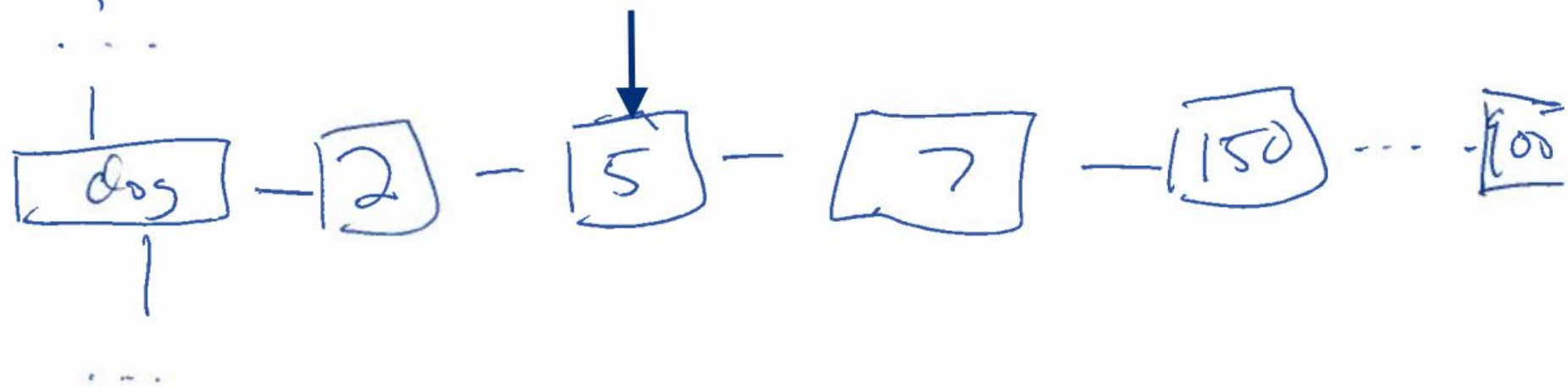
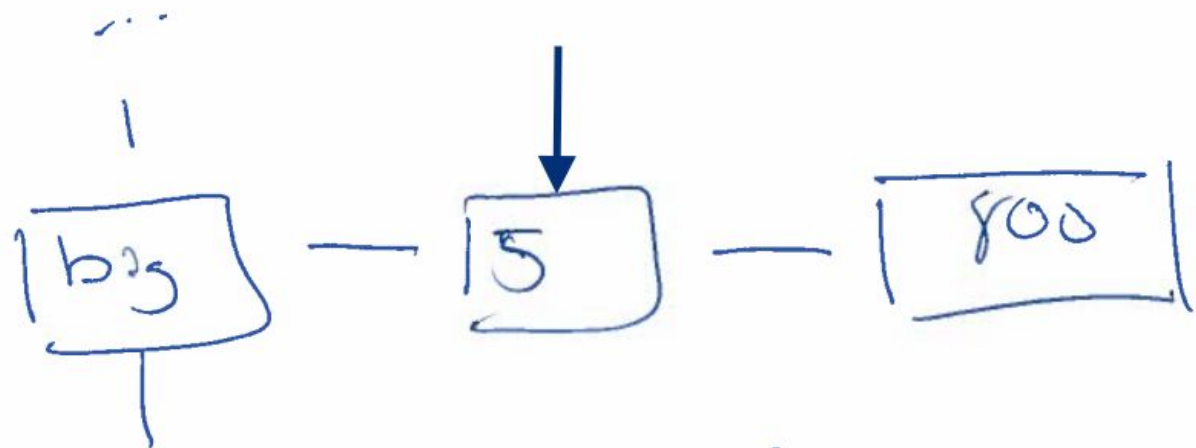
TODO: add diagram

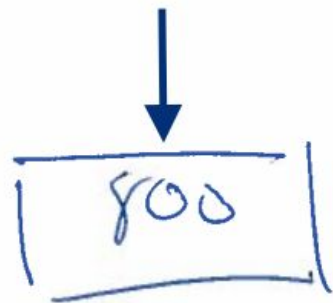
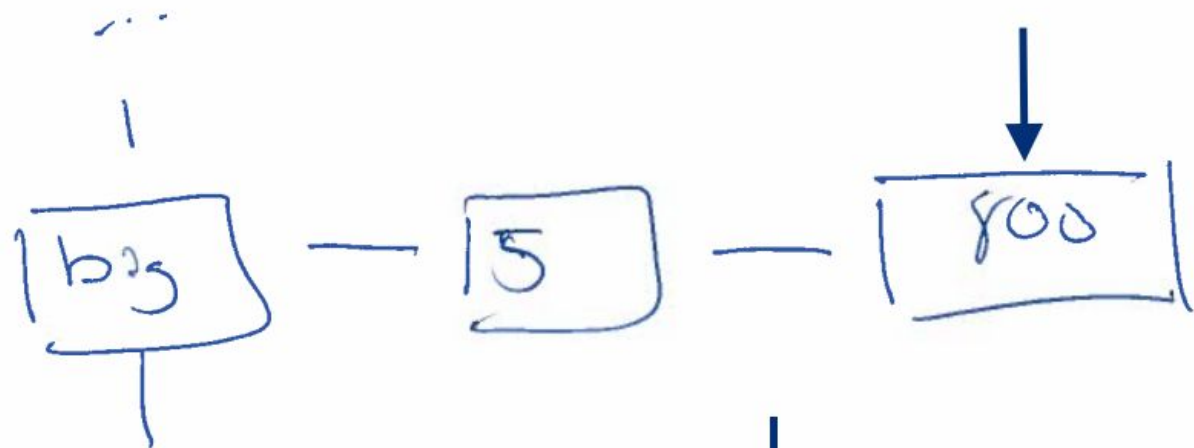


# Posting list









How many terms?

TODO: pseudo-code

TODO: diagram about how bits drop out

TODO: search is a high dynamic range problem.

TODO: higher rank rows

TODO: sharding by document length

TODO: diagram of how things fit together. Could just be concentric circles

# Posting lists are standard

---

**Coursera:**

Term Vocabulary and Postings Lists (IIR Ch. 2)

**Notes:**

[\[powerpoint\]](#) [\[PDF/6\]](#) [\[PDF/1\]](#)

**Readings**

[IIR Ch. 2](#)

MG Ch. 3.6 4.3

MIR 7.2

[Porter's stemmer \(MIR\)](#), [Porter stemming algorithm \(Official\)](#)

[A skip list cookbook \(Pugh 1990\)](#)

[Fast phrase querying with combined indexes \(Williams, Zobel, Bahle 2004\)](#)

[Efficient phrase querying with an auxiliary index \(Bahle, Williams, Zobel 2002\)](#)

---

**Coursera:**

Index Compression (IIR Ch. 5)

**Notes:**

[\[powerpoint\]](#) [\[PDF/6\]](#) [\[PDF/1\]](#)

**Readings:**

[IIR Ch. 5](#)

MG 3.3, 3.4

[Compression of inverted indexes for fast query evaluation \(Scholer et al. 2002\)](#)

[Inverted index compression using word-aligned binary codes \(Anh and Moffat 2005\)](#)

---

# Posting list optimizations

Skip list

Delta compression

etc.



Search

Perf: how to think about it?

Performance

Search is BIG

# Parsing / Tokenization

Harder than it sounds

# Search is a big problem

## Tokenization

Some languages mix alphabets, are partially left-to-right and right-to-left, etc.

Can't drop non-alphanumeric characters (C# vs C++)

## Multi-language queries

## Ranking / Relevance

## Distributed Systems

etc.