Bloom filters

Ben Langmead

Please sign guestbook (www.langmead-lab.org/teaching-materials) to tell me briefly how you are using the slides. For original Keynote files, email me (ben.langmead@gmail.com).
Bloom filter

Imagine we start with a hash table ...

... and start taking things away
Bloom filter

Imagine we start with a hash table ...

... and start taking things away

Take away values

Now we have a hash set
Imagine we start with a hash table...

... and start taking things away

Take away values

Now we have a hash set

Now take away keys
Bloom filter
Bloom filter

```
1
0
0
0
0
0
1
0
0
0
0
0
1
0
0
0
0
0
0
0
0
0
1
0
0
0
0
0
```
Bloom filter: inserting

Start with all 0s
Bloom filter: inserting

To insert item, hash it to a bucket and set bit to 1 (if not already)
Bloom filter: inserting

\[ h_1(x_1) \]
\[ h_1(x_2) \]
\[ h_1(x_3) \]
Bloom filter: inserting

\[ h_1(x_1) \]
\[ h_1(x_2) \]
\[ h_1(x_3) \]
\[ h_1(x_4) \]

If already 1, this is a *collision*

Life goes on
Bloom filter: querying

Use same hash function $h_1$ to hash query item; check if bucket is 0 or 1
Bloom filter: querying

If filter says "0" -- item is definitely not present
Bloom filter: querying

If filter says "0" -- item is definitely not present

If filter says "1" -- item *may be* present

...or may be a collision

One-sided error

Randomized algorithms can exploit one-sided error via repeated trials

$h_1(z)$
Bloom filter

What if we used many hashes/filters; adding each item to each?

\[ h_1 \quad h_2 \quad h_3 \quad \ldots \quad h_k \]

Will all filters have the same number of set (=1) bits?

Not necessarily; one hash/filter might have more/fewer collisions than another
Bloom filter

\[ h_{\{1,2,3,...,k\}}(y) \]
Bloom filter

If any query yields 0, item is not in the set.
Bloom filter

If all queries yield 1, item may be in the set; or we might have collided $k$ times.
Bloom filter

Say item is not present; chance that \( k \) filters all return 1 is \( p_1^k \)

...increasing \( k \) reduces error exponentially

Say \( p_1 = 50\% \) for all filters; 10 filters give collective error rate of \( \left( \frac{1}{2} \right)^{10} < 0.1 \% \)

\( p_1 \) : fraction of bits set
\( k \) : number of hash functions
Bloom filter

Rather than use a new filter for each hash, one filter can use $k$ hashes
Rather than use a new filter for each hash, one filter can use $k$ hashes ($k = 4$ here).
Rather than use a new filter for each hash, one filter can use $k$ hashes ($k = 4$ here)
Bloom filter

Say filter has $n$ bits and uses $k$ hash functions

Assume hashes are well behaved, i.e. uniform and independent

What's the probability a given bit is 0 after $m$ items have been inserted?

$$\left(1 - \frac{1}{n}\right)^{mk}$$
Bloom filter

Probability a given bit is 0 after $m$ items are inserted

What's the probability of a false positive?

Where all $k$ hash functions find a 1?

Call this $f$

$$p_0 = \left( \frac{1}{n} \right)^{mk}$$

$$p_1 = 1 - p_0$$

$$p_1^k = (1 - p_0)^k$$

$$= \left(1 - \left( \frac{1}{n} \right)^{mk} \right)^k$$
Bloom analysis: road ahead

\[ p_0 = \left( 1 - \frac{1}{n} \right)^{mk} \]

\[ f = \left( 1 - \left( 1 - \frac{1}{n} \right)^{mk} \right)^k = (1 - p_0)^k = p_1^k \]

1. Approximations in terms of \( e \)
2. Choosing \( k \) (# hashes) for fixed \( m, n \)
3. Choosing \( m/n \)
4. Analyzing fullness with Balls & Bins
Bloom filter

Rewrite \( \left( 1 - \frac{1}{n} \right)^{mk} \) in terms of \( e \):

\[
\left( 1 - \frac{1}{n} \right)^{mk} = e^{\ln \left( \left( 1 - \frac{1}{n} \right)^{mk} \right)} = e^{\ln \left( 1 - \frac{1}{n} \right)^{mk}} = e^{\ln \left( 1 - \frac{1}{n} \right)^{mk}}
\]

Can we approximate \( \ln \left( 1 - \frac{1}{n} \right) \)?
Bloom filter

Taylor expansion of $\ln(1 + x)$: $x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots$

"Mercator series"

If $x$ is small (e.g. 0.001), terms beyond first are really small ($<1$ millionth)

$$\ln \left(1 - \frac{1}{n}\right) \approx -\frac{1}{n}$$

$$e^{\ln \left(1 - \frac{1}{n}\right)mk} \approx e^{-\frac{mk}{n}}$$
Bloom filter

\[ p_0 = \left(1 - \frac{1}{n}\right)^{mk} \approx e^{-\frac{mk}{n}} = \tilde{p}_0 \]

\[ f = (1 - p_0)^k \approx \left(1 - e^{-\frac{mk}{n}}\right)^k = (1 - \tilde{p}_0)^k \]

Approximation using Taylor expansion of \( \ln(1 + x) \)

\[ e^{-\frac{m}{n} \cdot k} \]

Where \( \frac{m}{n} = \# \text{ data items per slot} \)
Bloom filter

How do we choose # of hash functions $k$?

Can we have too many?
Yes, quickly clogging the filter with 1s

Too few?
Yes, failing to get much exponential decrease in error

$p_1^k$

$p$ is too close to 1

Small $k$, not much compounding
Bloom filter

We know that f.p. rate \( (1 - e^{\frac{m}{n}k})^k \) gets large for small and large \( k \). To find minimum, find where derivative is 0.
Bloom filter

\[
\frac{d}{dk} \left( 1 - e^{-\frac{mk}{n}} \right)^k \text{ is tricky, so log the function first}
\]

\[
\frac{d}{dk} \left[ k \ln \left( 1 - e^{-\frac{mk}{n}} \right) \right]
\]

At top level, use product rule

\[
\frac{d}{dk} \left[ k \right] \cdot \ln \left( 1 - e^{-\frac{mk}{n}} \right) = \ln \left( 1 - e^{-\frac{mk}{n}} \right) + k \cdot \frac{d}{dk} \left[ \ln \left( 1 - e^{-\frac{mk}{n}} \right) \right]
\]

Need chain rule in here
Bloom filter

\[
\frac{d}{dk} \left[ k \ln \left( 1 - e^{-\frac{mk}{n}} \right) \right] = \ln \left( 1 - e^{-\frac{km}{n}} \right) + \frac{km}{n} \cdot \frac{e^{-\frac{km}{n}}}{1 - e^{-\frac{km}{n}}}
\]

Derivative is zero when \( k = \ln 2 \cdot \frac{n}{m} \)

\[
e^{-\frac{km}{n}} = e^{\ln 2 \cdot \frac{n}{m} \cdot -\frac{m}{n}} = e^{-\ln 2} = \frac{1}{2}
\]

\[
\frac{km}{n} = \ln 2 \cdot \frac{n}{m} \cdot \frac{m}{n} = \ln 2
\]
Bloom filter

\[
\frac{d}{dk} \left[ k \ln \left( 1 - e^{-\frac{mk}{n}} \right) \right] = \ln \left( 1 - e^{-\frac{km}{n}} \right) + \frac{km}{n} \cdot \frac{e^{-\frac{km}{n}}}{1-e^{-\frac{km}{n}}}
\]

\[
e^{-\frac{km}{n}} = \frac{1}{2}
\]

\[
\frac{km}{n} = \ln 2
\]

Derivative is zero when \( k = \ln 2 \cdot \frac{n}{m} \)

Best choice of \( k \): \( k^* = \ln 2 \cdot \frac{n}{m} \)
Bloom filter

\[
(1 - e^{\frac{-mk}{n}})^k
\]

\[n/m = 10\]

\[k^* = \ln 2 \cdot 10 = 6.93\]
Bloom filter

If we pick ideal $k$ (# hashes) for fixed $m$, $n$, what fraction of the filter do we expect to be set bits?
Bloom filter

\[ e^{-mk/n} = \tilde{p}_0 \]

\[ \frac{-mk}{n} = \ln \tilde{p}_0 \]

\[ k = -\ln \tilde{p}_0 \cdot \frac{n}{m} \]

\[ -\ln \tilde{p}_0 \cdot \frac{n}{m} = \ln 2 \cdot \frac{n}{m} \]

\[ \ln \tilde{p}_0 = -\ln 2 \]

\[ \tilde{p}_0 = \frac{1}{2} \]

Best choice of \( k \):

\[ k^* = \ln 2 \cdot \frac{n}{m} \]

With \( k \) chosen optimally, filter is 50% set bits