Anthony Ventresque
anthony.ventresque@ucd.ie

Big Data Programming

COMP47470

Streaming (2)

School of Computer Science, UCD   Scoil na Ríomheolaíochta, UCD

FILTERING

Summarising vs. Filtering

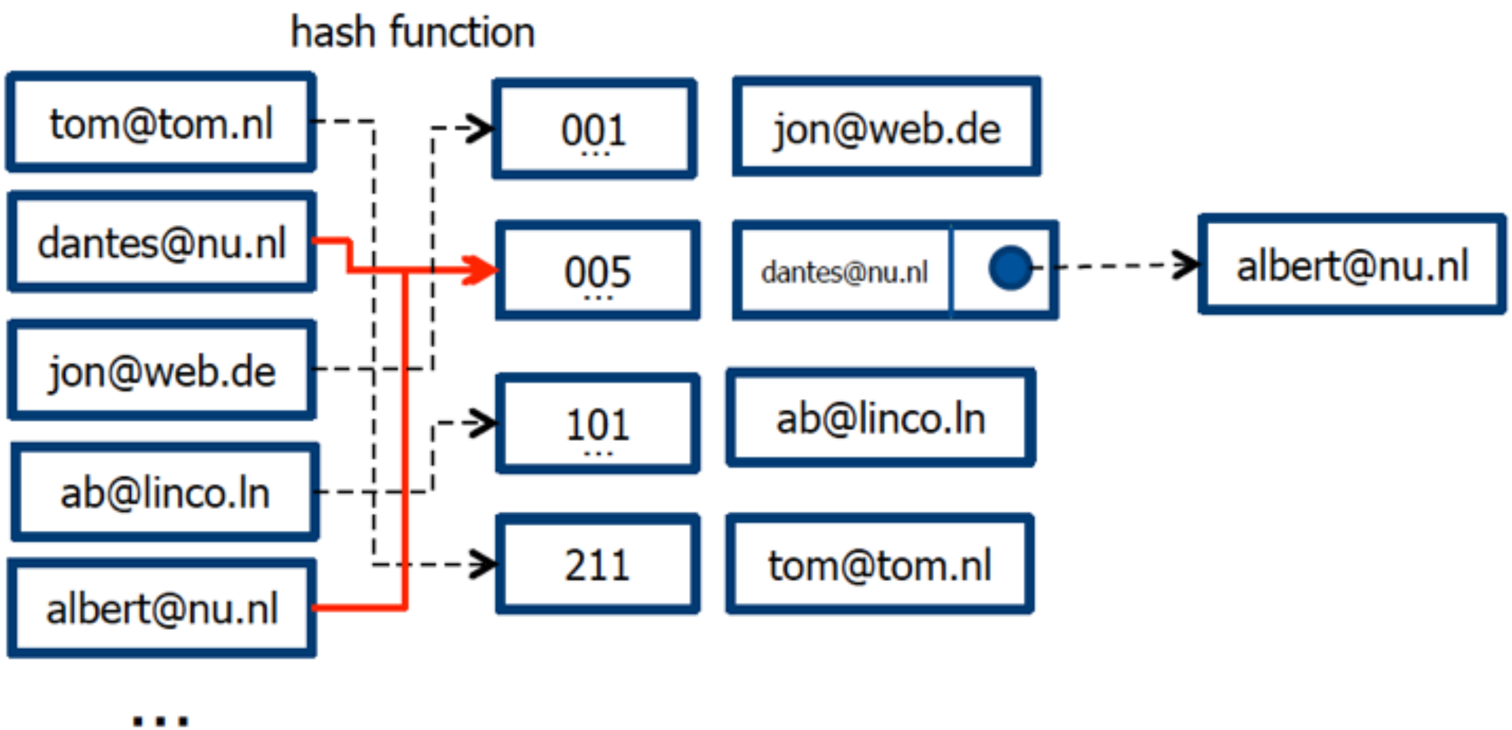
- So far: all data is useful, summarise for lack of space/time
- Now: not all data is useful, some is harmful
- Classic example: spam filtering
  - Mail servers can analyse the textual content
  - Mail servers have blacklists
  - Mail servers have whitelists (very effective!)
  - Incoming mails form a stream; quick decisions needed (delete or forward)
- Applications in Web caching, packet routing ...

Problem Statement

- A set *W* containing *m* values (e.g. IP addresses, email addresses, etc.)
- Working memory of size n bit
- Goal: data structure that allows fast checking whether the next element in the stream is in *W*
  - return TRUE with probability 1 if the element is indeed in W
  - return FALSE with high probability if the element is not in *W*

Reminder: Hash Functions

- Each element is hashed into an integer (avoid hash collisions if possible)



Bloom Filter

- Given
  - a set of hash functions {h1, h2, …, hk}, hi: W -> [1,n]
  - a bit vector of size n (initialised to 0)
- To add an element to W:
  - compute h1(e), h2(e), …, hk(e)
  - set the corresponding bits in the bit vector to 1
- To test whether an element is in W
  - compute h1(e), h2(e), …, hk(e)
  - sum up the returned bits
  - return TRUE if sum=k, FALSE otherwise

Bloom Filter: Element Testing

- **Case 1**: the element is in W
  - $h_1(e), h_2(e), \ldots, h_k(e)$ are all set to 1
  - `TRUE` is returned with probability 1
- **Case 2**: the element is not in W
  - `TRUE` is returned if due to some other element all hash values are set

What is the probability of a false positive?

→ What is the probability of k bits being set to 1?

→ What is the probability of the j^{th} bit being set to 1?

Bloom Filter: Element Testing

- **Case 1**: the element is in W
  - $h_1(e), h_2(e), \ldots, h_k(e)$ are all set to 1
  - `TRUE` is returned with probability 1
- **Case 2**: the element is not in W
  - `TRUE` is returned if due to some other element all hash values are set

$$
\begin{aligned}
P(BV_j \text{ set after } m \text{ inserts}) &= 1 - P(BV_j \text{ not set after } m \text{ inserts}) \\
&= 1 - P\left(BV_j \text{ not set after } k \times m \text{ hashes}\right) \\
&= 1 - \left(1 - \frac{1}{n}\right)^{k \times m}
\end{aligned}
$$

Bloom Filter: Element Testing

- **Case 1**: the element is in W
  - $h_1(e), h_2(e), \ldots, h_k(e)$ are all set to 1
  - `TRUE` is returned with probability 1

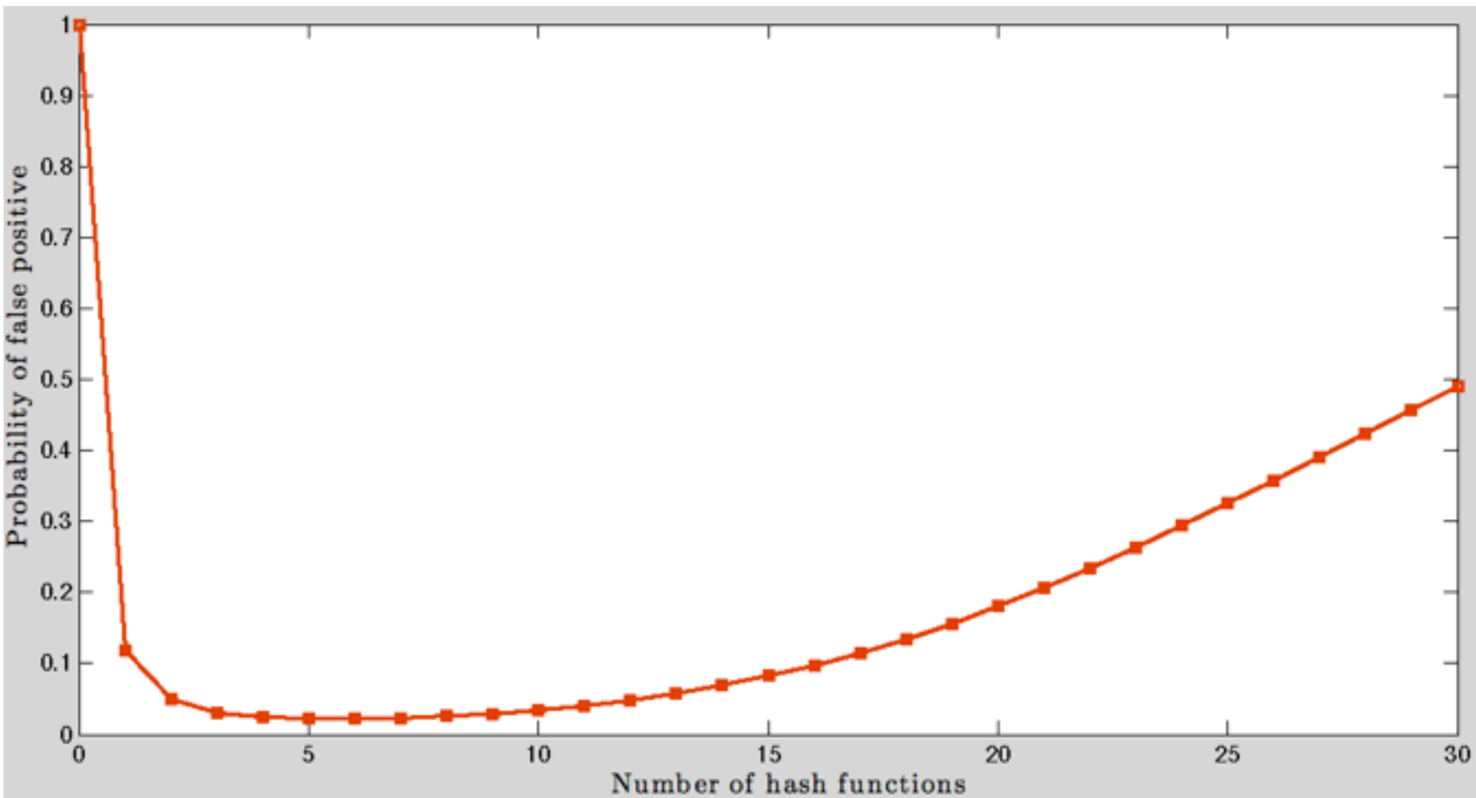
- **Case 2**: the element is not in W
  - `TRUE` is returned if due to some other element all hash values are set

$$
\begin{aligned}
P(BV_j \text{ set after } m \text{ inserts}) &= 1 - P(BV_j \text{ not set after } m \text{ inserts}) \\
&= 1 - P\left(BV_j \text{ not set after } k \times m \text{ hashes}\right) \\
&= 1 - \left(1 - \frac{1}{n}\right)^{k \times m}
\end{aligned}
$$

$$
P(\text{false positive}) = \left(1 - \left(1 - \frac{1}{n}\right)^{km}\right)^{k}
$$

Bloom Filter: How Many Hash Functions are Useful?

Example: $m = 10^9$ whitelisted IP addresses and $n = 8 \times 10^9$ bits in memory



Bloom Filter Tricks

- Union of two Bloom filters of the same type in terms of hash functions and bits
  - OR the two bit vectors
- To half the size of a Bloom filter with a filter size the power of 2
  - OR first and second half together.
    When hashing the higher order bit can be masked.
- Bloom filter deletions?
  - Not possible in the standard setup.
  - Solution: counting bloom filters (instead of bits use counters that increment/decrement).

DISTINCT ELEMENT ESTIMATES

Application areas

- Number of distinct queries issued to a search engine
- Unique IP addresses passing packages through a router
- Number of unique users accessing a website per month
- Number of different people passing through a traffic hub (airport, etc.)
- Database query plans (original motivation for FM)

FM-sketch (Flajolet-Martin)

- Approach: hash data stream elements uniformly to N bit values, i.e.:

$$h : a_i \rightarrow \{0, 1\}^N$$

- Assumption: the larger the number of distinct elements in the stream, the more distinct the occurring hash values, and the more likely one with an unusual property appears

FM-sketch (Flajolet-Martin)

- One possibility of interpreting **unusual** is the **hash tail**: *the number of 0's a binary hash value ends in*

100110101110	1001101011100	10011000000000

for all $a_i \in S$ (our stream):

$$h(a_i) \rightarrow \{0,1\}^N$$

maximum hash tail seen so far

$$K = \textit{max-tail}_{a_i \in S}\ h(a_i)$$

return $|\hat{S}| = 2^K$

N must be long enough; there must be more possible results of the hash function than elements in the universal set.

FM-sketch (Flajolet-Martin)

- **Intuitive justification**

$$P(h(a)\ \textit{has tail length of at least}\ r) = \frac{1}{\underbrace{2\times 2\ldots\times 2}_{r\text{ 0's occur}}} = \frac{1}{2^r}$$

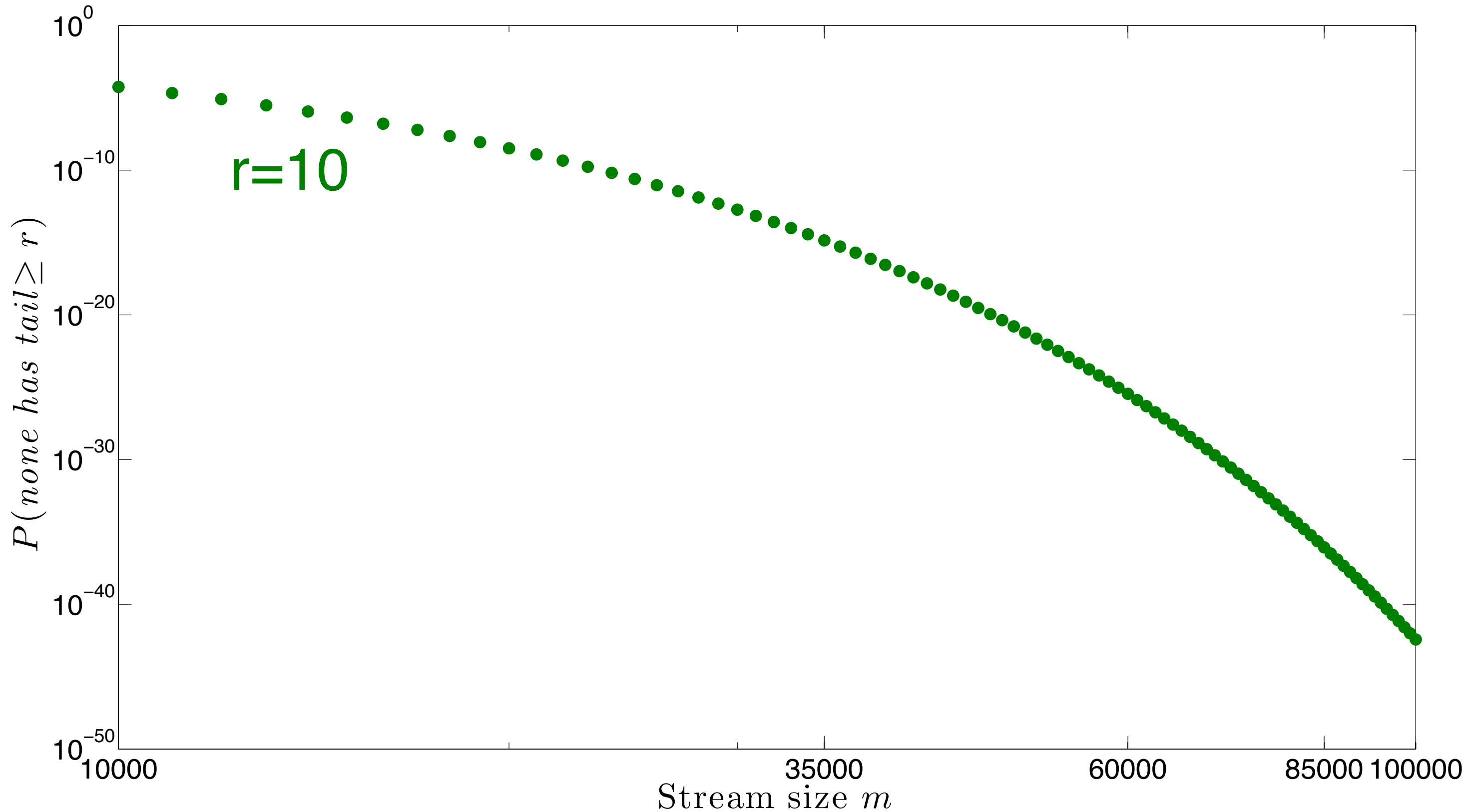
- **When there are m distinct elements in the stream**

$$P(\textit{none has tail length} \geq r) = \left(1 - \frac{1}{2^r}\right)^m$$

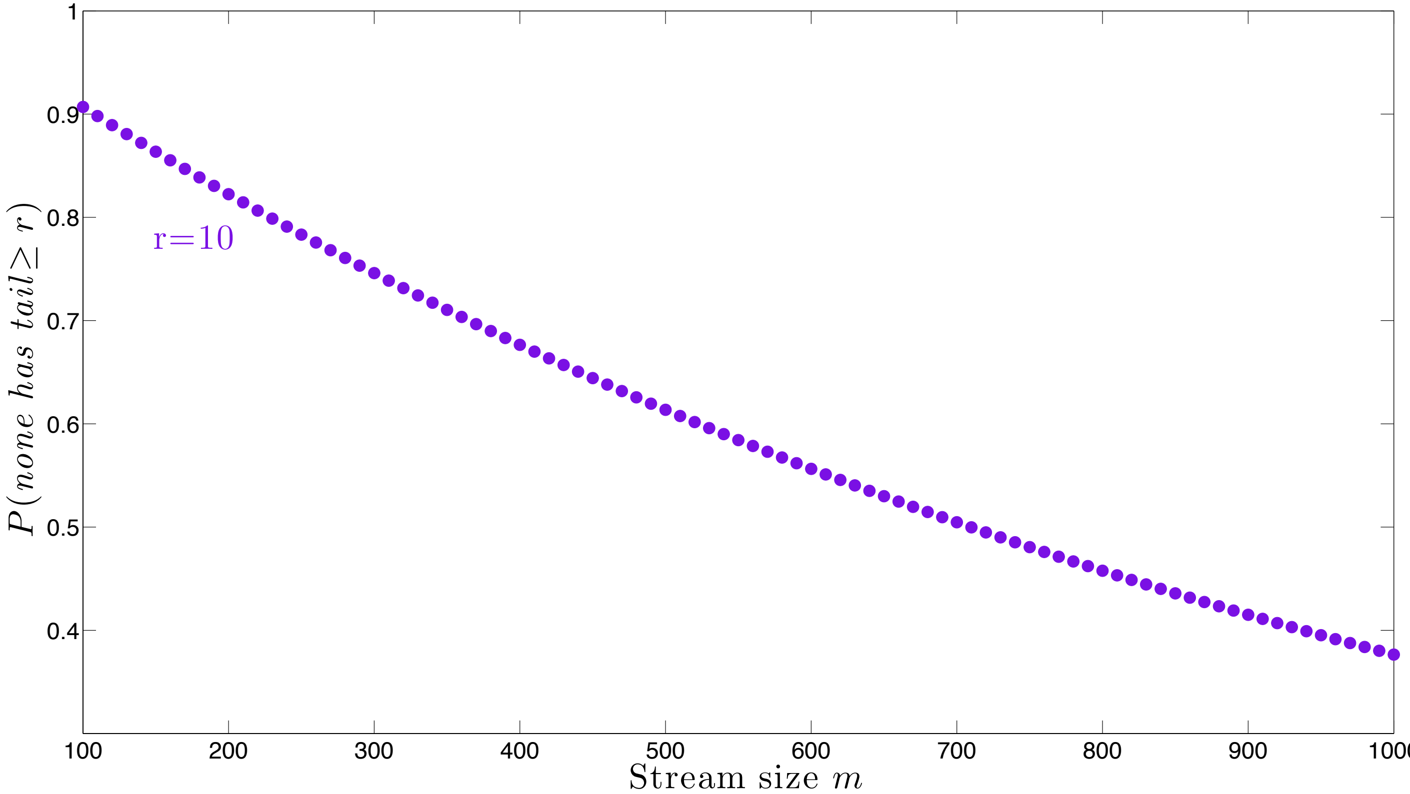
$$\textit{if } m \gg 2^r : \textit{the prob. of finding a tail } \geq r \textit{ reaches 1}$$

$$\textit{if } m \ll 2^r : \textit{the prob. of finding a tail } \geq r \textit{ reaches 0}$$

FM-sketch (Flajolet-Martin)



FM-sketch (Flajolet-Martin)



FM-sketch (Flajolet-Martin)

- **Practical setup**
- *axb* independent sketches
  a groups of *b* sketches each
- **Median of means** for a stable result

Mean: outliers can lead to an overestimate

Median: always a power of 2