Lesson 4

Stream Computing Methods

Big Data Stream Computing

- Stream computing is a way to analyze and process Big Data in real time to gain current insights to take appropriate decisions or to predict new trends in the immediate future
- Implements in a distributed clustered environment
- High rate of receiving data in stream .

Stream computing Applications

- Financial sectors,
- Business intelligence,
- Risk management,
- Marketing management,
- Search engines, and
- Social network analysis

Data Stream Algorithms Efficiency Measurements

1. Number of passes (scans) the algorithm must make over the stream
2. Available memory
3. Running time of the algorithm.

Sampling methods Obtaining a Representative Sample

- First category, probabilistic sampling is a statistical technique
- Second category, non-probabilistic sampling uses arbitrary or purposive (biased) sample selection instead of sampling based on a randomized selection

Reservoir Sampling Method

- A random sampling method, choosing a sample of limited data items from a list containing a very large number of items randomly
- The list is larger than one that upholds in the main memory
- Example 7.4

Concise Sampling

- Concise sampling like the reservoir sampling method, with a difference that a value that appears once is stored as a singleton, whereas a value that appears more than once is stored as a (value, count) pair
- Inserts a new data item in the sample with a probability of 1/n.

Counting Sampling

- A refinement of concise sampling in terms of accuracy
- The method maintains the sample in the case of deletion of data items as well
- Decrementing the count value upon deleting a value
- [Deletion mean after reading moving to next.]

Procedures for calculating sample sizes

(i) estimation, called confidence interval approach, and

(ii) hypothesis testing. Statistics prescribes Chi-squared, T-test, Z-test, F-test, P value for testing the significance of a statistical inference

Filtering of Stream

- Identifies the sequence patterns in a stream
- Stream filtering is the process of selection or matching instances of a desired pattern in a continuous stream of data

Example

- Assume that a data stream consists of tuples
- Filtering steps: (i) Accept the tuples that meet a criterion in the stream, (ii) Pass the accepted tuples to another process as a stream and (iii) discard remaining tuples

Filtering of Stream: The Bloom Filter Analysis

- A simple space-efficient data structure introduced by Burton Howard Bloom in 1970.
- The filter matches the membership of an element in a dataset.

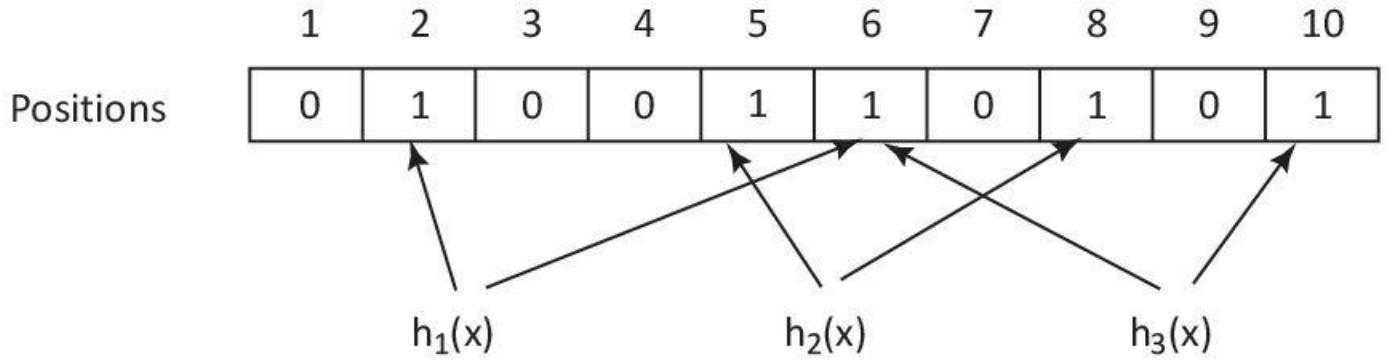
Bloom Filter

- The filter is basically a bit vector of length m that represent a set S = {x_1, x_2, . . . , x_m} of m elements,
- Initially all bits 0. Then, define k independent hash functions, h_1, h_2, . . . , and h_k. Each of which maps (hashes) some element x in set S to one of the m array positions with a uniform random distribution.

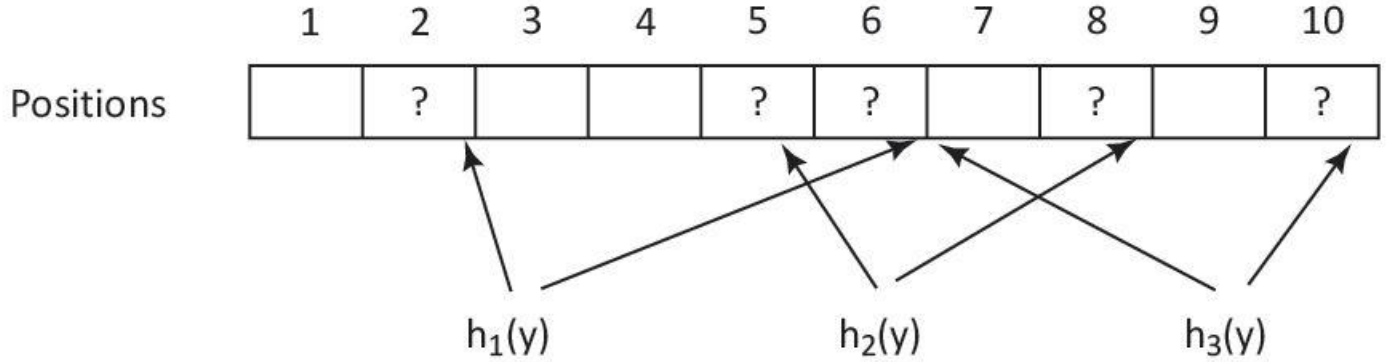
Bloom Filter

- Number k is constant, and much smaller than m. That is, for each element $x \in S$, the bits hi (x) are set to 1 for $1 \leq i \leq k$. [$\in$ is symbol in set theory for 'contained in'.]

Figure 7.4 (a) Inserting an element x in bit vector (filter) of length m = 10, (b) finding an element y in an example of Bloom Filter



(a) Inserting an element x in bit vector (filter) of length m = 10



(b) Finding element y is in the filter

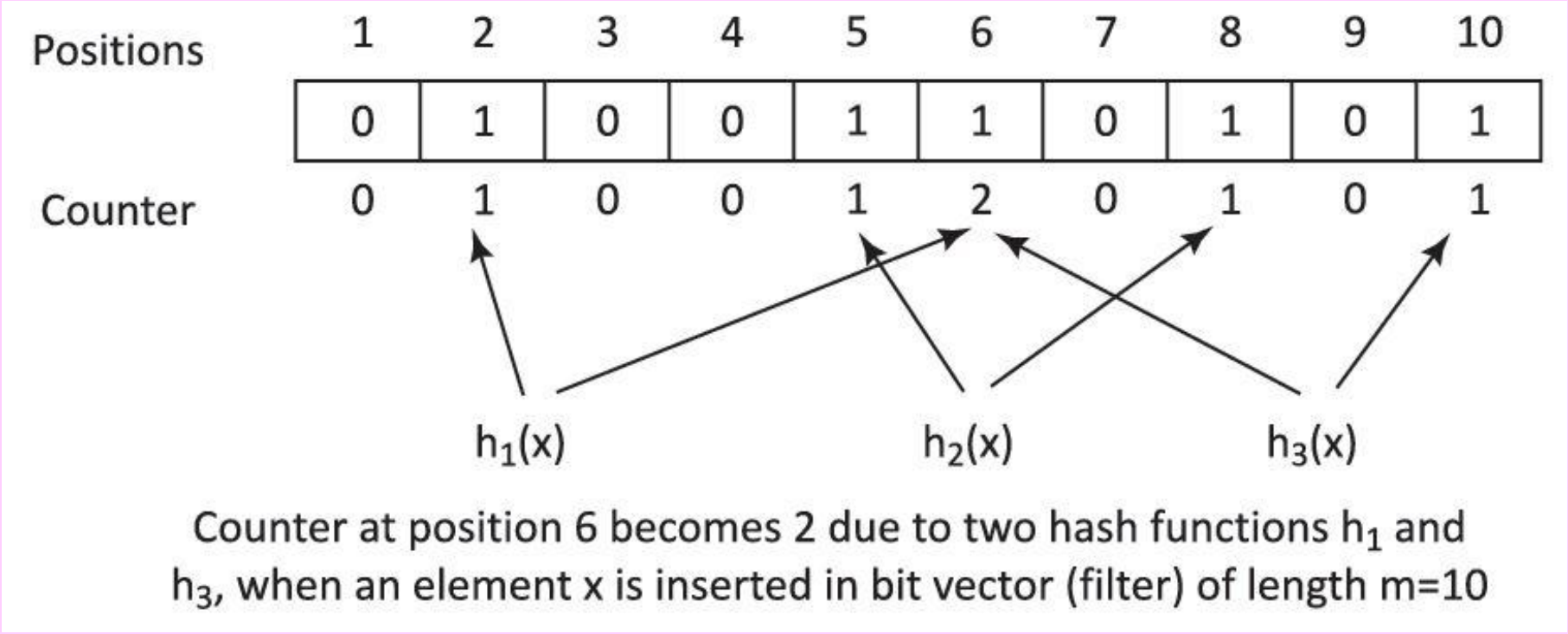
Counting Bloom Filter: A Variant of Bloom Filter

- Maintains a counter for each bit in the Bloom filter
- The counters corresponding to the k hash values increment or decrement, whenever an element in the filter is added or deleted, respectively

Counting Bloom Filter: A Variant of Bloom Filter

- As soon as a counter changes from 0 to 1, the corresponding bit in the bit vector is set to 1. When a counter changes from 1 to 0, the corresponding bit in the bit vector is set to 0. The counter basically maintains the number of elements that hashed

Figure 7.5 An example of Counting Bloom Filter



Counter at position 6 becomes 2 due to two hash functions h_1 and h_3, when an element x is inserted in bit vector (filter) of length m=10

Counting Distinct Elements in a Stream

- Relates to finding the number of dissimilar elements in a data stream
- The stream of data contains repeated elements
- This is a well-known problem in networking and databases

Counting Distinct Elements in a Stream and Count Distinct Problem

- If n possible elements a_1, a_2, …, and a_n are present then for an exact result *n* spaces are required. In the worst case, all *n* elements can be present. Let m be the number of distinct elements. The objective is to find an estimate of m using only s storage units, where s ≪ m.

Bitmap algorithm to compute distinct elements

- Example 7.6

The Flajolet–Martin (FM) Algorithm

- FM method approximates the m, number of distinct (unique) elements, in a stream or a database in one pass
- The stream consisting of *n* elements with m unique elements runs in O (*n*) time and needs O (log (*m*)) memory. e

.. FM Algorithm

- Thus, the space consumption calculates with the maximum number of possible distinct elements in the stream, which makes it innovative

Features of the FM algorithm

(i) Hash-based algorithm.

(ii) Needs several repetitions to get a good estimate.

(iii) The more different elements in the data, the more different hash values are obtained

•

Features of the FM algorithm

(iv) Different hash values suggest the chances of one of these values will be unusual [the unusual property can be that the value ends in many 0s (alternates also exist)].

- Example 7.5.

Counting of 1's in a Window

- **Infinite Stream Processing**
- **Figure 7.3:** Volume of data is too large that it cannot be stored. Hardly a chance exists to look at all of it.
- Important queries may be likely to ask only about the most recent data or summaries of data.

Sliding Time Window Method for Data Stream Processing

- A popular model for infinite data stream processing. (Window refers to time interval during which stream raised and processed the queries)
- The receiving of data elements is taking place one by one arrived.

Sliding Time Window Method for Data Stream Processing

- Statistical computations are over a sliding time-window of size N (not over the whole stream) in time-units
- Window covers the most recent data items arrived.

Datar-Gionis-Indyk-Motwani (DGIM) algorithm

- Gives solution when N very large, (assume N is 1 Billion)
- DGIM algorithm suggests that store just the O [$\log_2 (\log_2 N)$] bits per stream
-

Datar-Gionis-Indyk-Motwani (DGIM) algorithm

- Concept of time buckets used
- A time window divides in a number of buckets, different from hash buckets

Decaying Windows

- Useful in applications which need identification of most common elements
- Decaying window concept assigns more weight to recent elements
- The technique computes a smooth aggregation of all the 1's ever seen in the stream, with decaying weights.

Decaying Windows

- When element further appears in the stream, less weight is given.
- The effect of exponentially decaying weights is to spread out the weights of the stream elements as far back in time as the stream flows

Summary

We learnt:

- Big Data Stream Computing
- Sampling the stream
- Filtering
- Bloom Filter, Counting Bloom Filter
- Counting distinct elements
- Flajolet–Martin Algorithm with one pass

Summary

We learnt:

- Counting of 1's in a Window
- Sliding time window
- DGIM algorithm
- Decaying Windows

End of Lesson 4 on

Stream Computing Methods