Turkish Journal of Computer and Mathematics Education *Vol.13 No.03 (2022), 728-735*

Research Article

Counting distinct pixel values from a video stream and comparing different approaches of FM algorithm

Anandteertha Rao [a], Atharva Vaidya [b], Yagnesh Narayanan [c]

[a, b, c] Under Graduate Students of South India Education Society Graduate School of Technology, University of Mumbai, Navi Mumbai, Maharashtra, India.

Abstract: We display a comparative study of different approaches that can be used to count the number of distinct elements in a stream data. The normal approach resembles the flajolet martin algorithm, the average approach, the median approach and the combined approach take into account more than one hash function. The paper provides a detailed analysis on a video stream with all the algorithms applied.

Keywords: Flajolet Martin Algorithm, Queries, Database, Stream data, Big Data, Memory, Hash Functions.

1. Introduction

Every day on the internet, more than 2.5 quintillion bytes of data are created. To analyse this data, one has to collect this data, store it in a safe place, clean it and then perform analysis. One of the major problems faced by big data engineers is dealing with Un useful or redundant data. A lot of time and memory is used to store and analyse this extra data which turns out to be fruitless in the end. Thus, the removal of duplicate data becomes extremely essential to cut the analysis cost and reduce redundancy. Data cleaning can be done using various techniques but before cleaning the data, it is necessary to know the amount of useful data present in the dataset. Therefore, before the removal of duplicate data from a data stream or database, it is necessary to have knowledge of distinct or unique data present. A way to do so is by hashing the elements of the universal set using the Flajolet Martin Algorithm. The Flajolet Martin algorithm is used in a database query, big data analytics, spectrum sensing in cognitive radio sensor networks, and many more areas. It shows superior performance as compared with many other methods to find distinct elements in a stream of data.

Using the conventional brute force approach to find the distinct elements in a stream it takes a considerable amount of time and memory. For the same stream if we use the Flajolet Martin algorithm it is found that the memory used is half of the memory that is used in the above method. For large data sets or data streams, which require a lot of space brute force approach cannot be used. So, for this purpose, the Flajolet Martin algorithm is used. Not only does it occupy less memory, but it also shows better results in terms of time in seconds.

1.1. Objectives:

- The primary objective of our project is to occupy less memory for counting distinct pixel values from a video stream. If the stream contains n elements with m of them unique, this algorithm should run in O(n) time and should take O(log(m)) memory.
- To devise an algorithm to further improve the accuracy of this approximation algorithm by implementing Flajolet Martin algorithm using three approaches –
    - Average Approach
    - Median Approach
    - Combined Approach

2.Literature Survey

According to **Philippe Flajolet and G. Nigel Martin [1]**, counting distinct elements in a large dataset or stream data can be approximated by using a randomized approach, which works on the principle of hashing and performing a few operations to get the result. This approach proved to be effective against the traditional method of keeping all the data in memory at the same time to calculate the results, as its space complexity was O(log(m)).
Kevin J. Lang and Oath Research [2] described a new cardinality estimation algorithm that is extremely space-efficient. In an empirical comparison against the compressed HyperLog sketches, the new algorithm wins on all the three dimensions of the time, space and accuracy trade off. The prototype uses zstd compression library, and produces sketches that are smaller than the entropy of HLL, so no possible implementation of compressed HLL

can match its space efficiency. The paper's technical contributions include analyses and simulations of the three new estimators, accurate values for the entropies of FM85 and HLL, and a non-trivial method for estimating a double asymptotic limit via simulation.
Jérémie O. Lumbroso [3] stated that the article is a historical introduction to data streaming algorithms that was written as a companion to How Philippe Flajolet Flipped Coins to Count Data. Instead of maintaining exact counters, Morris suggested making increments in a probabilistic manner. But quickly pointed out that doing so using constant probabilities is not very useful as either the probability of an increment is too large or too small. This suggests that the probability should depend on the current value of the counter. But although these applications highlight the space-saving aspect of Approximate Counting, it would be mistaken to think that Approximate Counting is no longer relevant, with nowadays' huge storage sizes.

3.Discussion

Existing algorithm proposed by **Philippe Flajolet and G. Nigel Martin [1]** focuses on a randomized approach for counting distinct elements in stream data. It makes use of a hashing function and proves to be more efficient than traditional methods in terms of memory usage. The article by **Jérémie O. Lumbroso [3]** was an acknowledgement of flajolet's algorithm to count data. But instead of using exact counters, a probabilistic approach was followed. But in doing so he concluded that using constant probabilities wasn't fruitful. Conventional Flajolet Martin algorithm uses a single hash function. This paper discusses that using multiple hash functions and some further processing like averaging and taking median can optimize the algorithm further.

4.Scope

As the Flajolet Martin algorithm is an approximation algorithm, for the future scope we plan to calculate the standard deviation along with the distinct elements which can then be used to determine the bounds on the approximation with a desired maximum error.

4.1. Proposed Solution

1. Using the existing system that uses a brute force approach to find distinct elements in a stream, takes up a lot of time and memory as each element is taken into consideration.
2. The brute force approach is not suitable for large data sets or data streams as a large amount of space is required.
3. For this purpose, the Flajolet Martin algorithm is used as it occupies less memory and also shows better results in terms of time.
4. To improve the accuracy of Flajolet Martin algorithm a common solution has been to run the algorithm multiple times with different hash functions and combine the results from the different runs. For these three approaches are used –
    4.1. Average approach - Use multiple hash functions and use the average R instead.
    4.2. Median approach - Use multiple hash functions from the above step and use the median of the average R. This gives fairly good accuracy.
    4.3. Combined approach – We use a grouping factor to group the maximum of each function and then take the average of each group. And then we take the median of these average values to get the maximum value.

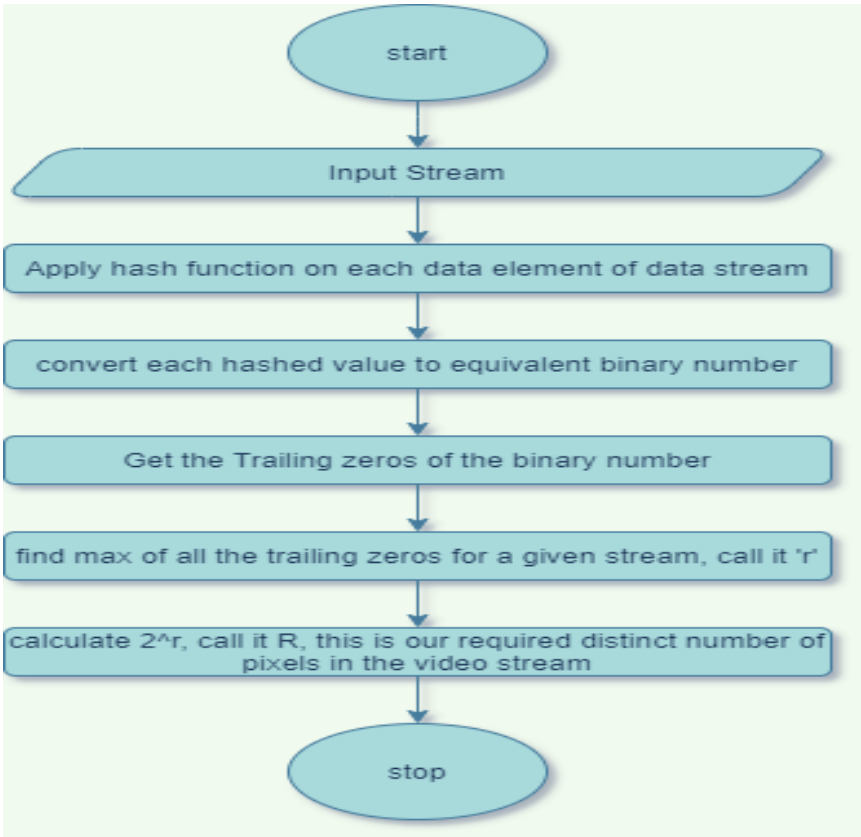
5. Methodology

5.1. Normal Approach

In this approach we take a data point or in our case a pixel value (r, g or b) and pass it through the hash function. Once we get the value of the hash function with the pixel value as a parameter, we calculate its binary equivalent. Finally, we calculate the number of trailing zeros of each value generated by the hash. We call the maximum of these trailing zeros as 'r' which is used to get the total number of distinct elements in the stream. 'R' is calculated as 2^r which is nothing but the total number of distinct elements present inside the stream.

The flajolet-martin algorithm is an estimation method, and would provide results as an estimation rather than definite results, hence to be certain of our results we should use median analysis in case of high variance in our data or use average approach to get better results.

Figure 1. Showing a simple flow chart describing the architecture of the flajolet martin algorithm



5.2. Average Approach

1. This algorithm uses the normal algorithm as a subroutine
2. In this we take multiple hash functions h1, h2, h3 …
3. Then we run the normal algorithm using each of these hash functions and then we find the maximum number of trailing zeros for each of them, hence for each hash function we will have one maximum value of trailing zeros.
4. Now we take the *average* of these maximum values and then 2 raised to this average value is the approximate number of unique elements in the input stream.

5.3. Median Approach

1. This algorithm uses the normal algorithm as a subroutine.
2. In this we take multiple hash functions h1, h2, h3 …
3. Then we run the normal algorithm using each of these hash functions and then we find the maximum number of trailing zeros for each of them, hence for each hash function we will have one maximum value of trailing zeros.
4. Now we take the *median* of these maximum values and then 2 raised to this median value is the approximate number of unique elements in the input stream.

5.4. Combined Approach

1. This algorithm uses the normal algorithm as a subroutine.
2. In this we take multiple hash functions h1, h2, h3 …
3. Then we run the normal algorithm using each of these hash functions and then we find the maximum number of trailing zeros for each of them, hence for each hash function we will have one maximum value of trailing zeros.
4. Say the grouping factor is 3, then we make groups of 3 with respect to the maximum values corresponding to each hash function. We then take the average of these 3 values corresponding to each of these groups.
5. Hence, we will have one average value for each group of 3. Now we will take the median of all these average values calculated from groups of 3. Finally, 2 raised to this number will be our approximate number of unique values.

6.Results

Figure 2. Denotes the input format for the algorithm and the types of hash functions required

```
hash format = (ax+b)%c
enter the number of hash you require?
 5
enter the 1st hash coefficients
 1 10 287
enter the 2nd hash coefficients
 2 10 578
enter the 3rd hash coefficients
 3 5 791
enter the 4th hash coefficients
 1 2 271
enter the 5th hash coefficients
 60 87 15874
enter the grouping size
 2
enter the url to get video stream:-
 scoob.mp4
```

Figure 3. Denotes the sample output from normal FM algorithm

```
total number of elements having abs difference below 20 = 67
```

	normal R	real R	abs diff
0	32	44	12
1	256	142	114
2	256	167	89
3	64	98	34
4	128	161	33

Figure 4. Denotes the sample output from average FM algorithm

```
total number of elements having abs difference below 20 = 44
```

	avg R	real R	abs diff
0	51	44	7
1	307	142	165
2	192	167	25
3	179	98	81
4	205	161	44

Figure 5. Denotes the sample output from average of r's approach for FM algorithm

```
total number of elements having abs difference below 20 = 13
```

	avg_r	real R	abs diff
0	16	44	28
1	64	142	78
2	64	167	103
3	32	98	66
4	64	161	97

Figure 6. Denotes the sample output from median of r's approach for FM algorithm

```
total number of elements having abs difference below 20 = 76
```

	median_r	real R	abs diff
0	32	44	12
1	256	142	114
2	128	167	39
3	64	98	34
4	128	161	33

Figure 7. Denotes the sample output from median of R's approach for FM algorithm.

```
total number of elements having abs difference below 20 = 76
```

	medianR	real R	abs diff
0	32	44	12
1	256	142	114
2	128	167	39
3	64	98	34
4	128	161	33

Figure 8. Denotes the sample output using a combined approach by taking the average of all r's.

```
total number of elements having abs difference below 20 = 77
```

	combine_r	real R	abs diff
0	32	44	12
1	256	142	114
2	64	167	103
3	128	98	30
4	128	161	33

Figure 9. Denotes the sample output using a combined approach by taking the average of all R's.

```
total number of elements having abs difference below 20 = 74
```

	combineR	real R	abs diff
0	48	44	4
1	384	142	242
2	96	167	71
3	160	98	62
4	192	161	31

Figure 10. Denotes the algorithm used and the number of datapoints which give the absolute difference between the real number of distinct elements and the number of distinct elements given by that algorithm, less than 20

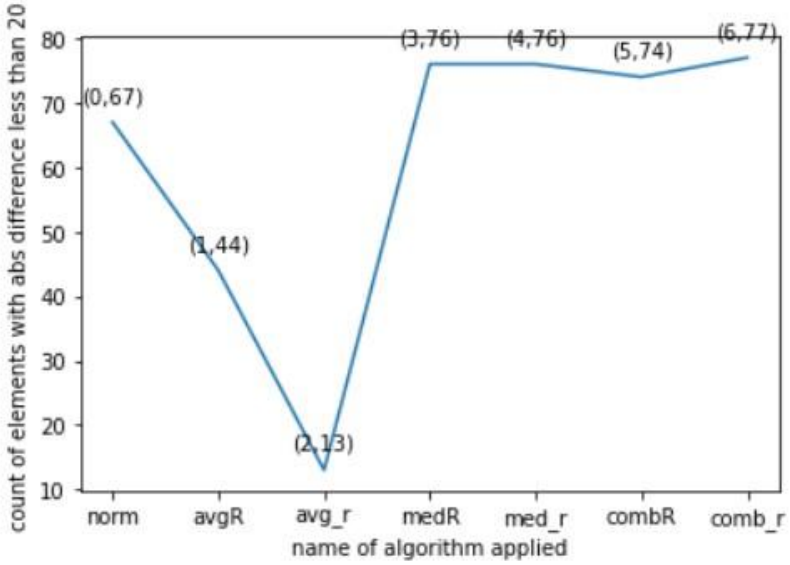


Figure 11. Denotes the algorithm used and the number of datapoints which give the absolute difference between the real number of distinct elements and the number of distinct elements given by that algorithm, less than 15.

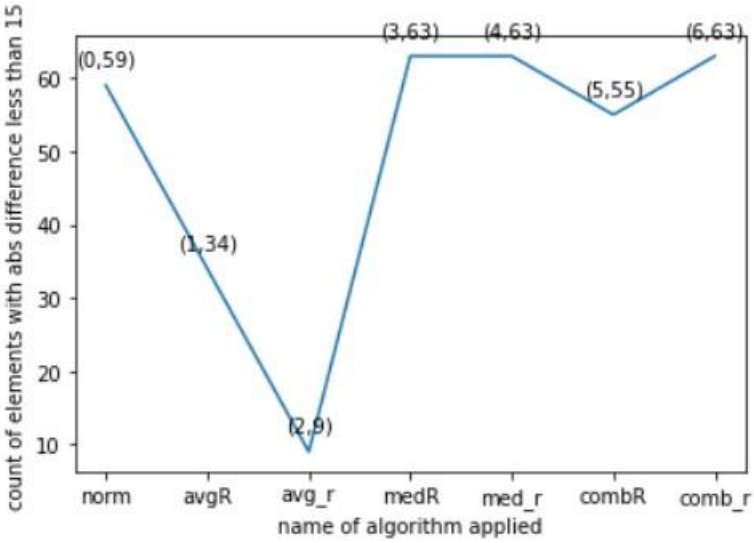
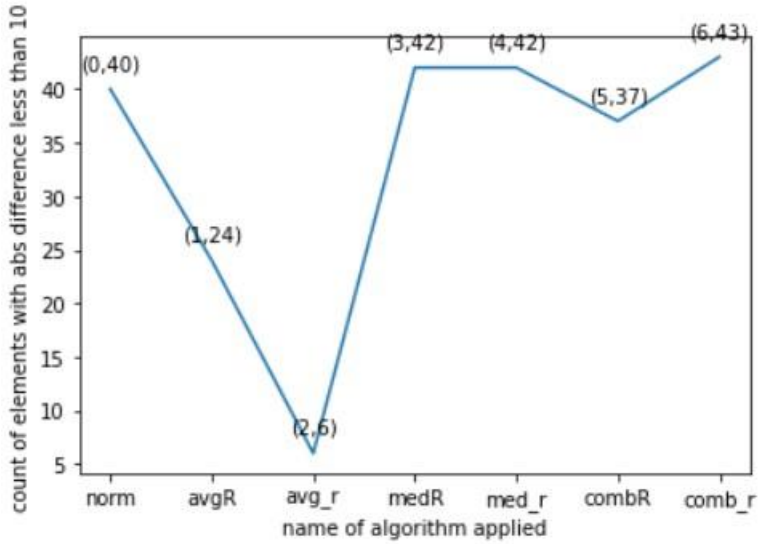


Figure 12. Denotes the algorithm used and the number of datapoints which give the absolute difference between the real number of distinct elements and the number of distinct elements given by that algorithm, less than 10.



According to the observations made from Figure. 10, Figure. 11 and Figure. 12 we can derive the following inference:

1. The hash used has a huge impact on the output.
2. The algorithm using ‘r’ or ‘R’ for average or median should be chosen wisely otherwise it would perform poorly.
3. The trends for different algorithms and their performance based on the absolute difference with the actual number of distinct elements remains the same.
4. With this trend we can safely assume that:
   4.1. Average approach performs poorly.
   4.2. Median approach gives us the best results when compared to other approaches. Considering the count of elements with absolute difference less than 20(Figure. 10), less than 15(Figure. 11), and less than 10(Figure. 12) it is observed that the Median approach provides us with highest accuracy.
   4.3. Combining using avg of all r’s along with both the median cases gives us about the same results as that of normal approach.

4.4. The normal FM algorithm, although not the best, still gives us great results and can be used when we need the results quickly as combining approaches and median approaches would require added complexity for time and space.

7.Conclusion

The Flajolet-martin algorithm hence was successful to get the total count of distinct elements in a video stream. We were able to conclude that the normal FM algorithm which requires just one hash function performed efficiently and gave great results, the median approach which uses more than one hash function and the combine approach which takes the average of all r's (maximum trailing zeros of each hash) performed the best and maintained their performance even when we increased the absolute difference between the actual value of distinct numbers and the value got by the respective algorithm. The FM algorithm which used the average of all r's to get the total number of distinct numbers in the stream performed really poorly.

Hence to conclude we could say that normal FM algorithms can be used if you want to get the results quickly or if you can spare some really powerful servers then going with a combined approach using average of all r's or the median of all R's or r's would prove to give you the best results.

References (APA)

Philippe Flajolet Inria, Rocyuencourt, 78153 Le Chesnay, France and G. Nigel Martin IBM Detlelopnzent Lahoralory, Hursley Park, Winchester, Hampshire S0212JN, United Kingdom. Probabilistic Counting Algorithms for Data Base Applications. Received June 13, 1984; revised April 3. 1985.

Kevin J. Lang and Oath Research. Back to the Future: An Even More Nearly Optimal Cardinality Estimation Algorithm (August 22, 2017).

Jérémie O. Lumbroso. The Story of HYPER LOG: How Flajolet Processed Streams with Coin Flips. (December 2013).

Ziv Bar-Yossef, T.S. Jayram, Ravi Kumar, D. Sivakumar and Luca Trevisan. Counting Distinct Elements in a Data Stream. (2002).