Literature Review

Jionglin Tao

Abstract

In this project,it studies lossless compression algorithms and bloom filter, and implement a new lossless compression and decompression algorithm. A bloom filter is a space-efficient probabilistic data structure that enables constant-time membership queries, and it has false positive. To avoid false positive, it uses a witness data structure in this project. For video compression, it uses a pre-processing with video frames in this project, And it optimizes the new lossless compression algorithm in this project. After implementing the algorithm, it evaluates this new algorithm against existing lossless compression algorithms and get the conclusion.

Key words:Lossless Compression, Bloom Filter, Video compression

1. Introduction

In an age of limited network bandwidth, data compression is particularly valuable. The prospect of efficient data compression enabling large web applications to be made possible on the move is very attractive. With the advent of the Big Data era, the volume of data and the rate at which it is growing are at an all-time high. With the rapid development of 5G technology, there is an increasing demand for applications and application scenarios such as edge computing and the Internet of Things. With limited transmission networks and storage capacity, data compression technology plays an increasingly important role.

Compression plays a centric role in reinforcing and enabling applications, where enormous amount of data get maneuvered every instant. Off the different types of data being maneuvered, different applications have different requirements for data compression schemes. Choosing an efficient and cost-effective compression algorithm to provide an optimal solution is very challenging and complex and thus involves performing tradeoff analysis on various factors which are important to the application[1].

Data compression algorithms have many interesting applications. For data compression, Any particular compression is either lossy or lossless. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by removing unnecessary or less important information[2]. And in this project, it studied in lossless compression and decompression algorithm.

This project studied the literature about data lossless compression algorithms, and implemented a new lossless compression and decompression algorithm with bloom filter. A Bloom filter is a space-efficient probabilistic data structure, conceived by Burton Howard Bloom in 1970, that is used to test whether an element is a member of a set. This new compression algorithm used bloom filter and used witness data structure to avoided the false positive rate of bloom filter to achieve lossless compression.

This project discusses the feasibility and correctness of the new algorithm and studied how the performance of the algorithm can be improved. It also evaluates the algorithm with existing algorithms and get the conclusion.

2. Problem and hypothesis

I try to use bloom filter to implement the lossless compression and decompression algorithm. Bloom filter (BF) is a space-efficient probabilistic data structure that enables constant-time membership queries. Let

S={x1,x2,…,xl} be a set of l elements such that S ⊆ U , where U is a universal set. BF represents such n elements using a bit vector of length n . All of the m bits in the vector are initialized to 0. Specifically, to insert an element x , a group of k independent hash functions, {h1,h2,…,hk} , are employed to randomly map x into k positions {h1(x),h2(x),…,hk(x)} (where hi(x) ∈ [0,m−1]) in the bit vector. Then the bits in these k vector positions are all set to 1. To query whether an arbitrary element is a member of set S , BF maps the element into its bit vector with the k hash functions and thereafter checks whether all the k bits are 1s. If any bit at the k hashed positions of the element is 0, the BF concludes that this element does not belong to the set; otherwise, the BF indicates that the queried element belongs to the set S [3].

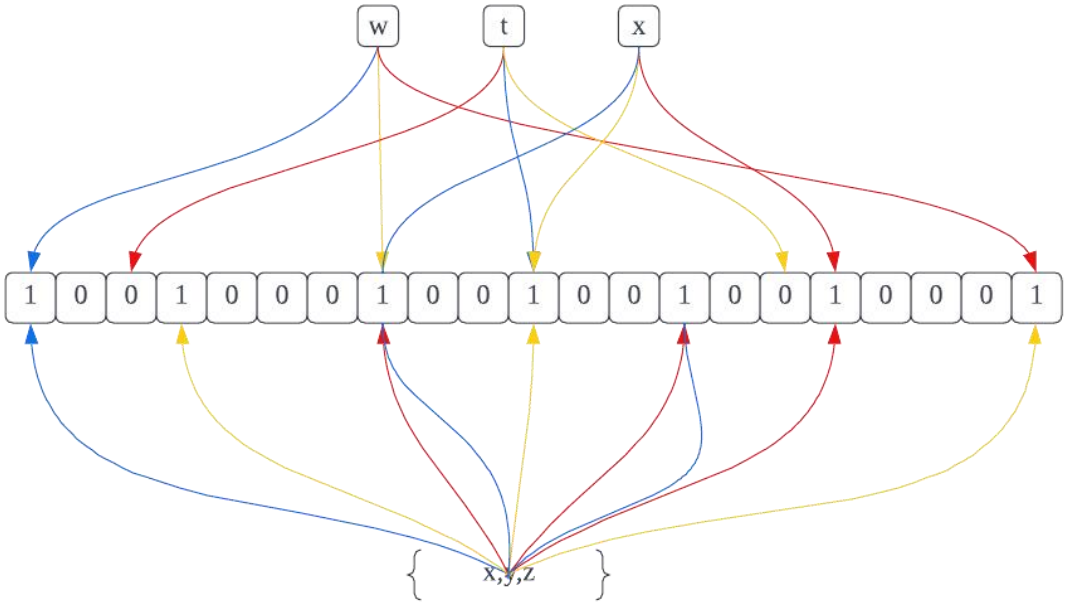

Figure1.Bloom Filter Example

As shown in Figure 1, add all the elements in the set {x,y,z} to the bitmap, get different hash function values for each element as the position and point to the bitmap, set the value of these positions as 1. The bitmap of bloom filter size is m=21. The in put size n=3. And the different coloured lines in the figure indicate different hash functions, there are k=3 hash functions. Query w, t, x whether them are in set {x,y,z}. w has a hash function value pointing to 0, so it must not be in the set {x,y,z}. t has all hash values pointing to 1, but it is not in the set {x,y,z}, so t is a false positive generated by the Bloom filter. x has all hash function values pointing to 1, and x is in the set {x,y,z}. The Bloom filter has false positives, but querying an element that is not in the set, then that element must not be in the set.

As a probabilistic data structure, BF supports fast membership query with potential false positive errors. The intrinsic characteristics of BF as Table1[3].The characteristics are all useful in compression algorithm, and it needs to avoid the false negative of bloom filter to achieve lossless compression effect. And I will use the witness data structure to solve this problem.

Space-efficient	BF programmes each element in a given set with a n -bit vector, irrespective of the number of bits in a bit representation of an element. With each element as input, the k independent hash functions will select k bits in the bit vector and set the chosen bits as 1s. The caused space overhead, i.e., the value of n , is only proportional to the number of elements l , and will not be affected by the length of the elements.
Constant-time query	By employing BF, querying the membership of elements in U can be simplified as binary checking of the corresponding k bits. If all the k bits are 1s, BF believes the queried element belongs to the set S , otherwise not. Thus the time-complexity of querying an element is O (k), which is much faster than trees (O (log n)) and table or list (O (n)). Note that, when the Bloom filter is implemented and k will be a constant. Then both the insertion and query complexity will be O (1).
One-sided error	Intrinsically, BF suffers from unavoidable false positive errors during a query, but no false negative errors. Specifically, if BF infers that an element x is not in the set S , users can exactly trust the judgment. By contrast, if BF concludes that x belongs to S , users cannot rule out the probability that $x \notin S$.

Table1.Intrinsic Characteristics of BF

In hypothesis testing,To query a element e whether in a set T that “e ⊆T, H(e) ⊆ H(T)?” will have 4 kinds outcome. True positive, false positive,true negative, false negative. As table1. But test in bloom filter don’t have false negative because of one-sided error characteristic.

True Positive	e ⊆T, H(e) ⊆ H(T)
False Positive	e⊄T,H(e) ⊆H(T)

False Negative	e ⊆T,H(e)⊈H(T)
True Negative	e⊈T,H(e)⊈H(T)

Table 2.True&False Positive&Negative

3. Solution

To implement to new lossless compression algorithm,it needs to eliminate the false positive of bloom filter, but the false positive can not eliminate directly. In this project, it used a witness data structure to record the false positive of bloom filter, the witness is constructed in data compression,and when decompression data, it can use witness to avoid the false positive.

In compression process, if an element passed constructed bloom filter, then check it whether in input set. If it in input set,then it is true positive,add it as a true positive to witness, else it is a false positive, add it as a false positive to witness. After all element queried, append the witness to bloom filter.

In decompression process, if an element passed the bloom filter, check the witness, if it is a true positive, add it as a true to out put,else it is false positive, add it as false to output. So that the witness can avoid the false positive of bloom filter.

For video compression, I will do a pre-processing. In the field of video compression a video frame is compressed using different algorithms with different advantages and disadvantages, centered mainly around amount of data compression. These different algorithms for video frames are called picture types or frame types. The three major picture types used in the different video algorithms are I, P and B. They are different in the following characteristics As table2.

I-frame (Intra-coded picture)	A complete image, like a JPG or BMP image file.
P-frame (Predicted picture)	holds only the changes in the image from the previous frame.
B-frame (Bidirectional predicted picture)	saves even more space by using differences between the current frame and both the preceding and following frames to specify its content.

Table2.I,P,B frame

The purpose of pre-processing is to make the input bit sequence more sparse, without directly compressing the video. The method uses the idea of

I-frames and P-frames. By recording only one I-frame and encoding all other frames as p-frames, the input bit sequence will be sparser and then compressed by means of a Bloom filter, which will result in much better compression effect.

To optimize the algorithm, The length of the Bloom filter and the number of hash functions need to be calculated and optimised.“Probability and Computing Randomized Algorithms and Probabilistic Analysis”[4] and “Optimizing Bloom Filter: Challenges, Solutions, and Comparisons”[2] described the method and process of optimizing bloom filter.

4. Evaluation and Conclusion

To evaluate the new lossless algorithm, there are various parameters to evaluate compression, like complexity of the algorithm, memory required to implement the algorithm, how fast the algorithm performs on a given machine (compression speed), the amount of compression (compression ratio), how closely the reconstruction resembles the original (image quality), reduced energy consumption etc[1]. I will compare the Compression Rate, Coding Complexity and Compression Speed between new algorithm and existing lossless compression algorithms.

After evaluation the new algorithm, it will know the advantages and disadvantages of the new compression algorithm. Then I will know what future works need to implement.

[1] S. Rao and P. Bhat, "Evaluation of lossless compression techniques," 2015 International Conference on Communications and Signal Processing (ICCSP), 2015, pp. 1655-1659, doi: 10.1109/ICCSP.2015.7322799.

[2] Pujar, J.H.; Kadlaskar, L.M. (May 2010). "A New Lossless Method of Image Compression and Decompression Using Huffman Coding Techniques". Journal of Theoretical and Applied Information Technology. 15 (1): 18–23.

[3] L. Luo, D. Guo, R. T. B. Ma, O. Rottenstreich and X. Luo, "Optimizing Bloom Filter: Challenges, Solutions, and Comparisons," in IEEE Communications Surveys & Tutorials, vol. 21, no. 2, pp. 1912-1949, Secondquarter 2019, doi: 10.1109/COMST.2018.2889329.

[4] Mitzenmacher, Michael and Eli Upfal [2005]. Probability and Computing Randomized Algorithms and Probabilistic Analysis. Cambridge. isbn: 978-0-52-183540-4.