ComS 535x:
Research Report

Due on May 8, 2015

Instructor: Professor Aduri, Pavankumar

Yuanyuan Tang, Chenguang He

Contents

Hashing and Bloom Filters

Bloom Filters is the one of most advanced data structure invent in recent years. It use the hash functions to reduce the space complexity. It is a probabilistic data structure, so the false positive exists. However it is relatively small, and there is no false negatives.Although Bloom filters allow false positive, for many applications the space efficiency take over this drawback, when the probability of an error is sufficiently low. In the following section, we discuses the hash function, structure of bloom filter and the usage of bloom filter.

A hash function is a function that take an item of data and process it to produce a value or key. It is a mapping that map two objects in a relation. For example, we can easily define the hash function of a string with the sum of all character and mod a large prime. When we want to check the whether two string are same, we only need to compute the hash function and check the result whether is same. This approach can save a lot of memory to store strings. However, a hash function is hard to be a one to one mapping function, since when two different data values to produce a same hash value, we call it hashing collision. A good hash function can reduce the hashing collision by spreading the data into the array. In the class, we discuss two normal hash functions: Random Hash Function and Deterministic Hash Functions.

Random hash function is defined by choosing the large prime p and define a and b from the range of [1,p-1], the function is $h(x) = (ax + b)\%p$. The advantage of using the random function is that, the function is defined by random large prime p which the selection of p can be variant, in other word, if we want two different random hash function, we just select different large prime p, it is very efficiency to build the function. Also, when we use the random hash function, we only need to store the [a,b,p] in a 3-tuple. It save a lot of space to store the functions themselves.

Deterministic hash function is defined by a constant large prime p, we try to build a formalized hash function h, such that, h is defined by take a large prime 109951168211, we call it FNV64PRIME and a offset of 14695981039346656037, we call it FNV-64INIT, the hash function is that: h = h xor s[i] and h = (h*FNV64PRIME)$\%2^{64}$. We call it deterministic hash function, because the parameters of hash function are constant, therefore, if we want to use different hash function, we need to have a algorithm to compute the offset, it is given in http://www.isthe.com/chongo/tech/comp/fnv/. The advantage of deterministic hash function is that, when we want to use a different hash function, we only need to change the offset of hash function to get the new function, it save the time to find the prime.

The Bloom Filter is build by constructing k hash function, and it has the interface of add(), contains(). However, the original approach of Bloom Filter can not move the element, the reason is that you can not unset a bit that appears to belong to a data item because it might also be set by another data item. When we add an item into Bloom Filter, it compute k hash function of the item and set the bitset of k values to true. When we search an item in Bloom Filter, we compute the k hash functions and check whether one of them is true in bitset. The Bloom Filter trade the accuracy for space, we can calculate the false positive rate: $(1 - e^{\frac{-kN}{M}})^k$, where k is the number of hash function, N is the size of item size S, M is the size of bit table.

The application of Bloom Filter uses in many large companies. Quora implements shared bloom filter in feed backend to filter out the question people have seen before. Facebook uses bloom filter for their typeahead search for fetching friends and stories. Google uses bloom filter in Chrome to filter the attack websites. Those companies implements the bloom filter in their own ways, however, the idea of bloom filter is widely accepted in industry.

Document Similarity

Document Similarity is a large used technique to find out how similar two document are. There are many powerful tools in Document Similarity, such as, Min Hash, Jaccard Similarity and Cosin Similarity. In the following section, we discuses the Min Hash and Locality Sensitive Hashing, which is a technique to have the signature of a document, when we want to compare two document, we only compare their signatures, it saves a lot of time to look at the entire document. Finally, we discuses the usage of those tools in industry.

In order to compute Min Hash, we firstly need to process the document as vector. The reason is that, the document are arbitrarily large, we do not have enough memory to hold the entire set of document and compare all of them with each other. The idea is that, we need to do some pre-process to compute the entire document into vector. The first thing we need to do is that, we need to filter the STOP word, such that, are, the. and. Those words are too frequency in documents, they are not helpful for making the signature. Secondly, we define the length of shingle to split the words in document, a shingle is a substring of a word, the reason that we need to do shingling is that, words have different tense, shingling can reduce the affect of tense. Finally, we compute T_fIDF.

T_fIDF is one of most significant way to find out the weight of a word in a document. T_fIDF consists of two parts, first part is T_f and second part is IDF. T_f represents term frequency, it means that the frequency of a term in one document. It normally represent in a vector, we call it term-frequency vector. Finally, we need to normalizing the term-frequency vector, because we do not want the number in vector is too different with each other.IDF represent the Inverse Document Frequency. It means that the number of a document a term appears. The reason that we need to compute IDF is that, some words are too frequency showing in the set of document, even they are not STOP words, they are may not very important. In order to define a word is important, we have to compute IDF to reduce the affect of those words. The document vector define as the product of T_f and IDF.

Once we have the set of document vector of documents, we can compute the Min Hash matrix. The Min Hash matrix represent the signature of set of documents. Firstly, it computes k different permutation and for each of them, compute the minimum value in each document vector. It result in a set of k minimum values for each document as the column, and number of document as the row. The reason that we use Min Hash to represent the document vector is that, the original document vector are too spares, there are many null pointers in the vector, by computing the Min Hash, it largely reduce the size of document vector.

Finally, we compute the Locality Sensitive Hashing. It split the Min Hash matrix into k-tuples and pick a random hash function to compute the k-tuples into a set. It define two document are similar, if two document that are mapped into the same bucket. Because the step of computing Locality Sensitive Hashing can reduce the sensitive of words, so we can compute the possibly of two document are mapped into same bucket is that $1-(1-s^r)^b$. It is relatvily small if we choose b and r so that s approximately equal to $(1/b)^{1/r}$.

Crawling and Page Rank

Information Retrieval