Assignment 2 outline

Tuesday, 29 March 2022 1:48 PM

1. Make a program, draw and analyse (????) S-curve in three conditions.
2. Read section 4.3.1, calculate **false-positive rate** in alter conditions, in three hash function and four hash function.
3. MapReduce in Hadoop for friend recommendation. With source code, description.
4. Data streams. Answer questions regarding Flajolet-Martin Algo. 6 sub questions
5. Write 1200 words for 3.6 3.7

Question 2

Saturday, 2 April 2022 8:51 PM

4.3 Filtering Steams

Bloom filtering, a method that easy to eliminate most tuples do not meet the criterion. When set is too large to store in main memory.

4.3.1 example

A set S for one billion email address. Stream consist of pairs with an email and an email address. A address takes 20 bytes, so can not store them all in main memory.

We use main memory as a "bit array"(???), assume 1 GB memory, have eight billion bits(one byte equal to eight bits), hash each member S to one bit, set bit to 1(1 billion). All other elements keep zero(7 billions).

Once the stream coming, if the address hashes to 1, pass. Or, drop. About 7/8 spam will be eliminated. We can check the rest with secondary memory(disk), or use cascade filters, eliminate 7/8 every level.

4.3.2 The Bloom Filter

A Bloom filter include:

1. An array of n bits, initial 0 for all.
2. **A collection of hash function h1, h2, h3... Each maps key to n buckets, corresponding to n bits of bit array.**
3. Set S of m key values.

Purpose: allow all key in S get through. Reject most keys which are not in S.

Take each key value in S and hash it using each of hash function

> To initialize the bit array, begin with all bits 0. Take each key value in S and hash it using each of the k hash functions. Set to 1 each bit that is $h_i(K)$ for some hash function h_i and some key value K in S.

For coming K ,if h1(K), h2(K), h3(K) … are ALL 1 in bit array. Pass. Else, fail.

4.3.3 Analysis of Bloom Filtering

Key value in the S will surely pass. But if not in S still have possibility could pass.

This is **False Positive.**

n = bit array length	m = number of S set	k = number of hash function

suppose x targets, y darts

A dart didn't hit the particular target = (x-1)/x (Like there is 10 target, I only want to hit 10, equal chance no missing. 9/10 chance hits other targets)

None of y darts hit the particular target = ((x-1)/x) ^ y (with y tries)

Approximate to e^(-y/x).

Back to example above one billion member set S is 10^9 darts. Eight billion "ONE" bits is 8*10^9 targets. Probability given target can not be hit by all darts is e^(-y/x) = e^(-1/8) , hit is 1-e^(-1/8) = 0.1175

With variable on table, target x = n, dart = k*m. Remain zero is e^(-km/n)

The false positive is 1-e^(-km/n) with k hash function (1-e^(-km/n)) ^ k

1. N = 10 billion bits, m = 2 billion set, k = 3? K = 4?
   K = 3 false positive = (1-e^(-3*2 b/10 b)^3 = 0.09185
   K = 4 false positive = (1-e^(-4*2 b/10 b)^4 = 0.09195

2. k = n/m * ln(2)

Question 4

Sunday, 3 April 2022 6:06 PM

Flajolet-Martin Algorithm?

4.4 Counting distinct elements in a stream

4.4.1 The Count-Distinct Problem
Suppose to know how many different elements have appeared in the steam.
Example: website wants to know how many different users in the access.
When data is massive huge, can not store them in main memory.

Alter method: estimate the number of distinct elements with using less memory

4.4.2 Flajolet-Martin Algorithm
Idea: more different elements in the stream, more different hash-value.
Use the "unusual hash value", which ends in many 0 (???????)

When apply h function to an elements a. h(a) will be generated. With several 0's end (maybe zero of 0). This number is the tail length for a and h. Let R be the maximum tail length, there should be 2^R for number of distinct elements (R digits, 1/0)

A given element a has h(a) ending at least r 0's is 1/(2^r) = 2^-r
If there are m distinct elements in stream, possibility that none of them r tail length is (1 - 2^-r)^m. The possibility of not finding stream element with r 0's at the end is e^(-m2^(-r))

$e^{-m2^{-r}}$.

If m much larger than 2^r, possibility find a tail length at least r approaches to 1
If much less than 2^r, possibility find a tail length at least r approaches to 0

So, 2^R (R is largest tail length) Estimate

4.4.3 Combining estimates
1. Group the hash functions into small groups, take their average
2. Take the median of average.
3. Group should be of size at least a small multiple of log2 m (???????)

Part 1: data include 3,1,4,6,5,9
Q1: h(x) = (2x+1) mod 32 7 3 9 13 11 19
7: 111
3: 11
9: 1001
13: 1101
11: 1011
19: 10011
Maximum tail: 0 estimate number: 2^0 = 1

Q2: h(x) = (3x+7) mod 32 16, 10, 19, 25, 22, 2

16: 10000
10: 1010
19: 10011
25: 11001
22: 10110
2: 10
Maximum tail: 4 estimate number: 2^4 = 16

Q3: h(x) = 4x mod 32 12, 4, 16, 24, 20, 4
12: 1100
4: 100
16: 10000
24: 11000
20: 10100
4: 100
Maximum tail: 4 estimate number: 2^4 = 16

Part 2: data include 4, 5, 6, 7, 10, 15
Q4: h(x) = (6x+2) mod 32 26, 0, 6, 12, 30, 28
26: 11010
0:
6: 110
12: 1100
30: 11110
28: 11100
Maximum tail: 2 estimate number: 2^2 =4

Q5: h(x) = (2x + 5) mod 32 13, 15, 17, 19, 25, 3
13: 1101
15: 1111
17: 10001
19: 10011
25: 11001
3: 11
Maximum tail: 0 estimate number 2^0 = 1

Q6: h(x) = 2x mod 32 8, 10, 12, 14, 20, 30
8: 1000
10: 1010
12: 1100
14: 1110
20: 10100
30: 11110
Maximum tail: 3 estimate number 2^3 =8

Question 3

Monday, 4 April 2022 3:51 PM

A people you might know social network recommendation algorithm

Key idea: If two people have a lot of mutual friends, system will recommend them to one another to get connected. For each user U, it recommends N = 10 users who are not already friends with U, but have the most number of mutual friends in common with U.
(10 people with most mutual friends) (possible to solve in single map reduce job)

Output: format <User><TAB><Recommendations>
<Recommendations> is a comma separated list of ID that <User> might know, order in decreasing order number

Question 5 3.6 outline

Sunday, 10 April 2022 12:51 PM

The Theory of Locality sensitive functions

The steepness of S-curve reflects how effectively can avoid false positive and false negatives in candidate pairs. Besides the minhash, other functions can produce candidate pairs efficiently.

These function can apply to space of set and Jaccard distance to another space or distance. Three conditions is needed:

1. They must be more likely to make close pairs be candidate pairs than distant pairs.(close should better than far)
2. They must be statistically independent.
3. They must be efficient in two ways: (a) they identify candidate pairs must be significantly less than scanning all pairs. (b) they must be combinable to build functions that can better avoid false positive and negatives (less bias)

3.6.1 Locality-Sensitive Functions

Consider functions take two items and render a decision about whether these items should be a candidate pair. In general, use a function F to hash item make decision base on whether they are equal. F(x) = F(y), means f(x, y) can be a candidate pair.

A collection of this form functions is called a family of functions.

Let d1 < d2 be two distances according to some distance measure d. A family F of function (d1, d2, p1, p2) sensitive for every f in F (p is probability)

1. If d(x, y) <= d1, the probability f(x) = f(y) is at least p1
2. If d(x, y) >= d2, the probability f(x) = f(y) is at most p2

3.6.2 Locality-Sensitive Families for Jaccard distance

For the moment only one way to find a family of locality-sensitive functions: use family of minhash functions, assume the distance is Jaccard distance. A minhash h make x and y are candidate pair only if h(x) = h(y).

Example: d1 = 0.3, d2 =0.6 . Asset the family of minhash function is (0.3, 0.6, 0.7, 0.4) sensitive family. (p1 = 1 - d1, p2 = 1 - d2). If Jaccard distance between x and y at most 0.3, SIM(x, y) >= 0.7, at least 0.7 chance minhash function will send x and y to same value.

If distance at least 0.6, SIM(x, y) <= 0.4, at most 0.4 chance x and y be sent to the same value, only d1 < d2 required.

3.6.3 Amplifying a Locality-Sensitive Family

Suppose have a given (d1, d2, p1, p2) family F. We can construct a family F' by AND-construction on F into (d1, d2 , (p1)^b, (p2)^b). The And construction is reducing the probability of a collision, also amplifies the difference probabilities of collision.

The Or construction turns (d1, d2, p1, p2) into (d1, d2, 1 - (1 - p)^b, 1 - (1-p2)^b). Or construction boosts the chance of collision in F', also increase the probability of the collision more nearby point than for points far away.

The AND construction lower all probabilities, but with F and r judiciously, we can make small probability p2 get close to 0 while higher p1 stay away from 0.

Same the Or construction rise probabilities. But with appropriate F and b, we make larger probability approach 1 and smaller one away from 1. What we intend to do is cascade AND and Or construction in any order to make the low probability close to 0 and high probability close to 1. (more construction be used, higher values r and b that be picked. That would take longer time to apply the function from this family)

Suppose a 4-way Or construction and 4-way AND construction.
F(0.2, 0.6, 0.8, 0.4). P= 0.8, 0.4 (1 - (1 - p)^4)^4 = 0.9936, 0.5740.
Then constructed family is (0.2, 0.6, 0.9936, 0.574) sensitive.