Bloom-Filter

DE SOUSA Benoît und SCHNEIDER Eloi

December 3, 2019

1 Idee des Bloom-Filters

Heutzutage gibt es viele Datenbanken und sehr oft wird nachgesehen, ob ein Wort (in unserem Fall) bereits in der Datenbank ist. Diesen Abfragenwerden als vergleichbare Algorithmen bezeichnet, dh sie vergleichen das getestete Wort mit jedem Wort in der Datenbank. Aber solche Anfragen benötigten sehr viel Speicherplatz.
1970 fand Burton Howard Bloom eine Lösung. Er entwickelte den Bloom-Filter.
Seine Idee war wie folgt. Er wollte die Abfragen in der Datenbank mit einem Bloom-Filter reduzieren, der vor der Datenbank platziert wird.
Er besteht aus einem m-stelligen Bit-Array (das am Anfang mit Nullen gefüllt ist) und aus k verschiedenen Hashfunktionen mit einem Wertebereich von 0 bis m-1 (m und k sind reelle Zahlen). Die Wörter, die in der Datenbank gespeichert werden, treten zuerst in den Bloom-Filter. Die k Hashfunktionen verarbeiten diesen Wörter und geben als Ergebnis k Zahlen zwischen 0 und m-1 zurück. Diese Zahlen stellen alle Bit in dem m-stelligen Bit-Array dar, die auf 1 eingesetzt werden müssen.
Die Abfrage können dann beginnen. Der Bloom-Filter erhält zuerst die Abfragen. Das getestete Wort folgt dem gleichen Prozess wie die anderen Wörter. Hier wieder werden Zahlen zwischen 0 und m-1 zurückgegeben. Der Bloom-Filter wird diese Zahlen/Positionen mit seinem Bit-Array vergleichen. Wenn es Unterschiede gibt, dann befindet sich das getestete Wort sicherlich nicht in der Datenbank. Andernfalls ist das getestete Wort nicht unbedingt in der Datenbank vorhanden. Es ist möglich, dass die durch die Hashfunktionen des getesteten Wortes erzeugten Positionen in die Positionen 1 fallen. Solche Fehler werden als falsch positive Ergebnisse bezeichnet.

Um einen Bloom-Filter verwenden zu können, ist es wichtig, die Anzahl der gespeicherten n Wörter und die Fehlerwahrscheinlichkeit p zu kennen, dh die Wahrscheinlichkeit, dass ein falsch positive Ergebnisse vorliegt. Die Werte von n und p werden verwendet, um die Länge des Bit-Arrays m und die Anzahl der Hash-Funktionen k zu ermitteln.

2 Vorteilen

Space and time advantages

3 Nachteilen

4 Beispiel aus der Praxis

5 Test unserer Datenstruktur