MapReduce

MapReduce adopts the method of first distribution and then synthesis. First, a map function is created to process the split data set, and then a reduce function is created to merge all the value values. You only need to care about how to divide the input data, the scheduling and error handling of the computer cluster, and the communication between the computers in the management cluster. MapReduce can make full use of the rich resources of the distributed system. Take the simplest example, such as 9 computers. A task that takes a few days to calculate can be quickly completed by 99 computers in a few minutes. A typical MapReduce calculation often consists of thousands of machines, processing terabytes of data. This number is quite staggering.

This paper focuses on parallel computing and its fault tolerance. Parallel computing was originally designed to meet the needs of a large number of computing fields, such as weather forecasting, oil exploration, and artificial intelligence research. Now its research content is constantly expanding, and it also pays attention to it. New computing models such as neural computing, quantum computing, etc. As the beauty of mathematics tells us, sometimes two unrelated things can be solved with one algorithm, but how to achieve it and what method to achieve, it must rely on ourselves. Some masters and scholars have been searching for the best for their entire lives. Simple and sophisticated algorithms, I have to say that it is a lucky thing to be able to directly learn their ideas and research results.

Google File System

Google File System GFS is a scalable distributed file system for large-scale, distributed applications that access large amounts of data. It runs on cheap common hardware, but can provide fault tolerance. It can provide a large number of users with services with higher overall performance.

The concept of gfs (google file system) is the distributed storage of files. Since Google did not have enough capital costs at the time, Google organized a lot of second-hand, cheap and accessible servers to process massive amounts of data. This is the gfs introduced by Google at the time. This massive data solution is called distributed storage of files-all the files we want in the entire system are distributed and stored on each server, unlike the previous mysql, oracle , It tends to be stored on a few efficient servers. This is a traditional relational database solution.

In GFS, the master is a single point, and only one master is in the active state at any time. A single point simplifies the design, centralized scheduling is much more convenient, and there is no need to consider the nasty "split brain" problem. But the single point poses a challenge to the throughput and usability of the system. So how to avoid a single point becoming a bottleneck? Two feasible methods: reduce interaction and fast failover.

Relatively speaking, the cost of distributed storage of gfs files is relatively low, and the hardware cost of traditional relational databases is relatively high. Moreover, the relational database cluster will be problematic when it reaches a certain degree of upward expansion, because oracle and mysql are generally composed of sever and client, while gfs separates the sever and the client, so that once the sever bursts, the following client is useless. . Moreover, with the huge amount of high-concurrency application data, the traditional relational database can no longer bear the processing problem of this amount of data. Naturally, the distributed storage method of files has emerged, which solves the problem of data storage.

Of course, gfs is based on many experiments, and there are many assumptions as the premise. When designing a distributed system, we must pay attention to whether our application is suitable for gfs and not blindly follow gfs.

BigTable

Bigtable is a sparse, distributed, and persistent storage multi-dimensional sorted Map. The index of the Map is the row key, column key, and timestamp; each vaulue in the Map is an unparsed byte array. The row key in Bigtable can be any string, and the read and write operations of each row are atomic; the row key in Bigtable is sorted and stored in lexicographic order, and the rows in the table can be dynamically partitioned. A partition is called a tablet, which is the smallest unit of data distribution and load balancing. Since the row keys are stored in lexicographic order, the query speed is milliseconds with the row key as the condition.

The persistent state information of the tablet is stored on the GFS. The update operation is submitted to the REDO log. Among these update operations, the most recently submitted ones are placed in a sorted cache, which we call memtable; earlier updates are stored in a series of SSTables.

With the execution of the write operation, the size of the memtable continues to increase. When the size of the memtable reaches a threshold, the memtable will be frozen, and then a new memtable will be created; the frozen memtable will be converted into an SSTable, and then written into GFS.

The client program can combine multiple column families into a localized group. A separate SSTable is generated for each locality group in the Tablet. Splitting the column families that are not visited together in the same city into different locality groups can improve the efficiency of read operations

The client program can control whether the SSTable of a local group needs to be compressed. Generally, two-pass, customizable compression is used

In order to improve the performance of read operations, the Tablet server uses a second-level cache strategy. The first level is used to cache the Key-Value pairs of the Tablet server through the SSTable interface; the block is the second-level cache, which is used to cache the Block of the SSTable read from the GFS.

The entire BigTable design meets the needs of most big data programs and breaks the structured storage of relational databases. It can be deployed on thousands of servers and can store PB-level data, which provides a solid foundation for the rapid development of the entire Internet industry. Theoretical basis and successful cases.

Efficient Social Graph GenerationWith the rapid development of today’s information age, both the number of users of social platforms and the number of works are rising sharply. Clearly, both real-world and synthetic graphs are necessary to evaluate the performance and scalability.

However, there are three major problems in the process of obtaining real-world social graphs.

Firstly, it usually takes significant time to analyze and process. Secondly, it is difficult to achieve a desired big real network. In other words, we cannot predefine the size and properties of the resultant network perfectly. Thirdly, the characteristics of large complex networks are often destroyed in the crawled real networks. Then, how to rapidly generate customized big social graphs becomes an important but basic research problem

FastSGG can efficiently and extensibly generate a trillionscale graph with the characteristics of a real social graph. First judge whether there is a graph that satisfies the given pattern and is NP-complete. Therefore, it is difficult to generate a graph that strictly satisfies the mode in polynomial time. For node attribute generation, the topological structure and attribute value can be associated, that is, connected nodes often have similar attributes. The user can specify the attribute value distribution. In addition, nodes in the same community have similar attribute values. So that FastSGG can use user-defined configuration to generate graphs.

In the paper, a degree distribution generation (D2G) model is proposed to speed up the process of graph generation. The D2G model uses a heuristic method to generate degrees and target vertices for the source vertices. The time complexity of this model for generating camber angles or determining target vertices for source vertices is O(1), so it can significantly improve the speed of social graph generation.

There are two social graph generation methods:LFR,S3G2. However, all these social graph generators do not support streaming locally to generate data.