Sketch based Load Balancer For Distributed Streaming Data

Zhuo Cheng, Junyan Pu, Xiang Si
Carnegie Mellon University
zhuoc2,junyanp,xsi@andrew.cmu.edu

ABSTRACT

There has been a lot of research work on load balancing for distributed computing, but existing work using consistent hashing only supports point queries (given a specific key, return the corresponding value). For modern scientific computing, we also want to support range queries (given a specific range, return all values whose keys fall in that range). There are some challenges we need to solve to support both load balancing and range queries for scientific computing: 1. disk writing is slow compared to network and CPU computing 2. memory is limited and most memory should be allocated to scientific computing 3. there might be some stragglers and their existence will greatly slow down the whole process. In this paper, we first run some simulation experiment and show that they have scalability issues on accuracy and memory usage. Then, we present a backup disk idea to solve the straggler problem. We monitor the writing progress of all processes and when identifying an straggler, we offload the records originally belong to straggler to the backup disk. Finally we design and implement an end-to-end system based on this idea. Our result shows that the disk writing phase takes most of the time and using a backup disk could speed up this phase by 10x when there is a straggler. We also shows that when no stragglers are present, using a backup disk reduces the overall disk writing time by 1.1x by taking over additional workload caused by imbalanced partitions.

1 INTRODUCTION

In modern scientific computing, some physicists or cosmologists use supercomputers to do simulation experiments. The experiments usually consist of two stages:

1. **Online simulation:** Scientists leverage supercomputers to simulate interactions between billions of particles and write records of particles to disks. Each record has multiple attributes, such as particle ID, temperature, magnetic field, timestamp, location, and velocity. When writing to disks, scientists will choose one attribute which they are interested in the most as the key, and the rest of the record will be treated as value corresponding to the key.
2. **Offline query:** After the simulation procedure is done, scientists will use different analysis tools to further examine the records. Under most circumstances, they will focus on a small portion of the particles (e.g. particles with high energy) and examine those particles' records more carefully. Existing solution only supports point queries and we want to support range queries in addition to point queries. By point query, we mean that each query comes with a specific key and we return all associated records with that key. By range query, we mean that query comes with a key range and we return all records whose keys fall in that range. When scientists perform a query with key 1-50, we want to support it in an elegant way instead of doing 50 different point queries.

Figure 1 shows a simplified procedure of scientific computing experiment. For the simplicity, we only show 3 processes while there could be hundreds or thousands of processes in the real experiments.

In online simulation stage, each process will generate lots of records and those records are stored in a key-value format. (e.g. key is the particle ID, value is other attributes). Records' key from different processes might have some overlap. (In our example as shown in Figure 1, each process will generate records with key range in 1-6). In order to support range queries, processes shuffle records before writing to disks, so that each disk will store records whose keys fall in a specific range. (In our example, disk 1 will store records with key range in 1-2, disk 2 will store records with key range in 3-4 and so on). In this way, when the range query comes, we only need to read a few disks to retrieve the corresponding records, which will make the query stage more efficient.

There are several challenges we need to address for our system:

1. **Disk is the bottleneck during online simulation stage:** Disk writing speed is relatively slower compared to network communication and scientific computing. So we need to partition the records evenly across multiple disks to maximize the disk writing bandwidth utilization.
2. **Memory is limited:** We want to allocate as much memory as possible to the scientific computing procedure and hence can not hold all generated records in memory. So we need to calculate partition boundaries on-the-fly