MP4 Report - Group#38 - yiteng3(Zhang) &maojunx2(Xu)

Project Design

There are four roles in our distributed machine learning system: client, coordinator, worker, and back_up_coordinator. The general workflow is as follows: the user inputs a query (a batch of files + job name) from the client, and the client sends the query to the coordinator. When the coordinator receives a query, there are two situations: one is that there is no job in the system, and the coordinator will allocate all the spare VM to the query; the other is that there is already a job in the system, and the coordinator The inference speed of the two jobs will be estimated, and the number of VMs of the two jobs will be redistributed according to the speed ratio. If the coordinator will monitor the query rates of the two jobs in real time, if the average query rate difference exceeds 20%, the coordinator will reduce the number of VMs for the faster job or increase the number of VMs for the slower job.

After each worker receives the query distributed by the coordinator, it directly starts the sub-thread for reasoning, and the reasoning result is returned to the coordinator. The coordinator summarizes the query results of the two jobs and writes them into a file, and then transfers the result file to the SDFS system. The client can obtain the inference results at any time through the get command.

The coordinator will synchronize the historical data and reasoning results of the query to the back_up_coordinator in real time. The back_up_coordinator monitors the coordinator status through the heartbeat mechanism. When the coordinator crashes, the back_up_coordinator send a broadcast to tell all nodes that it is elected as the new coordinator, and then continue to receive and distribute the query .

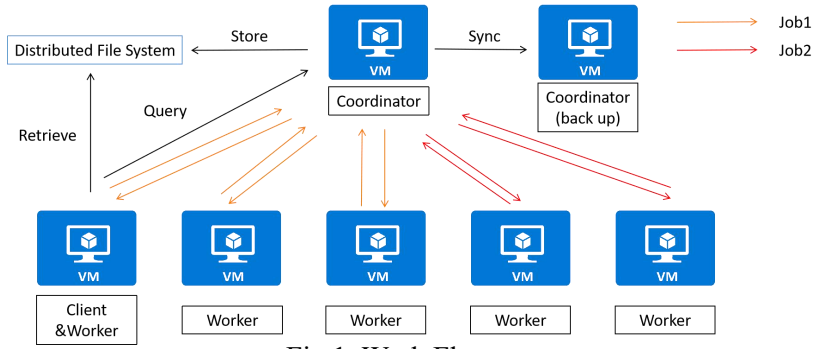


Fig 1. Work Flow

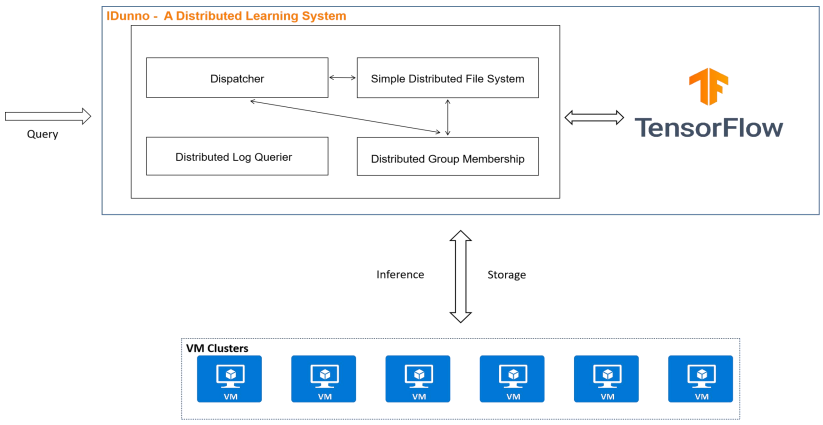


Fig 2. Overall architecture

Data and discussion

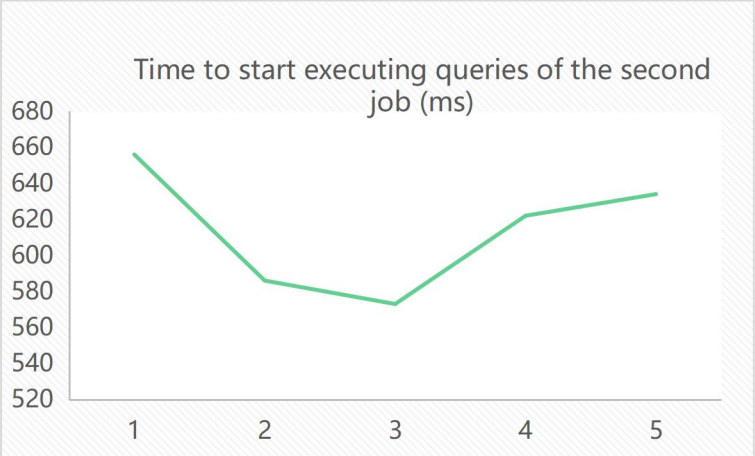
1) Fair-Time Inference:

1a) When a job is added to the cluster (1->2 jobs), what is the ratio of resources that IDunno decides on across jobs to meet fair-time inference?

Ratio: round(1*batch_size_1) : round(1.1*batch_size_2)

Discussion: We choose ResNet50 and Inception_V3 as two jobs. These two jobs are familiar. Through the sample test, we tested and figured out that the ratio of processing rate of two jobs is 1 : 1.1 (batch ratio: 1:1). And the experiment proved that, when batch sizes are same and ratio of VMs is 1:1, the query rate meet fair-time inference (about within 10%).

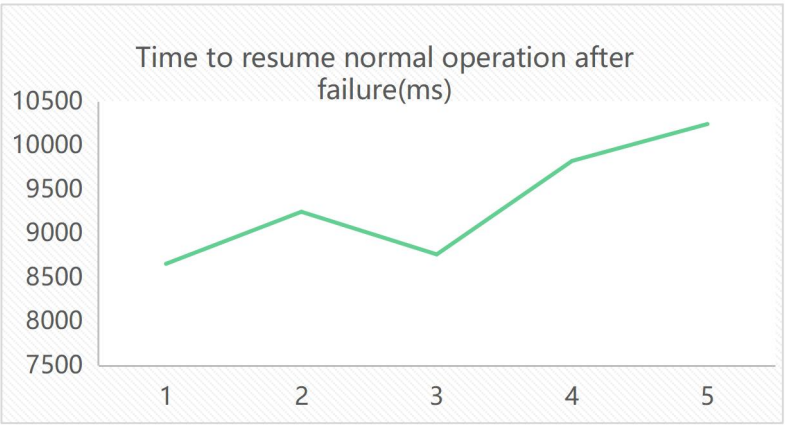
1b) When a job is added to the cluster (1->2 jobs), how much time does the cluster take to start executing queries of the second job?



Average time: 614ms

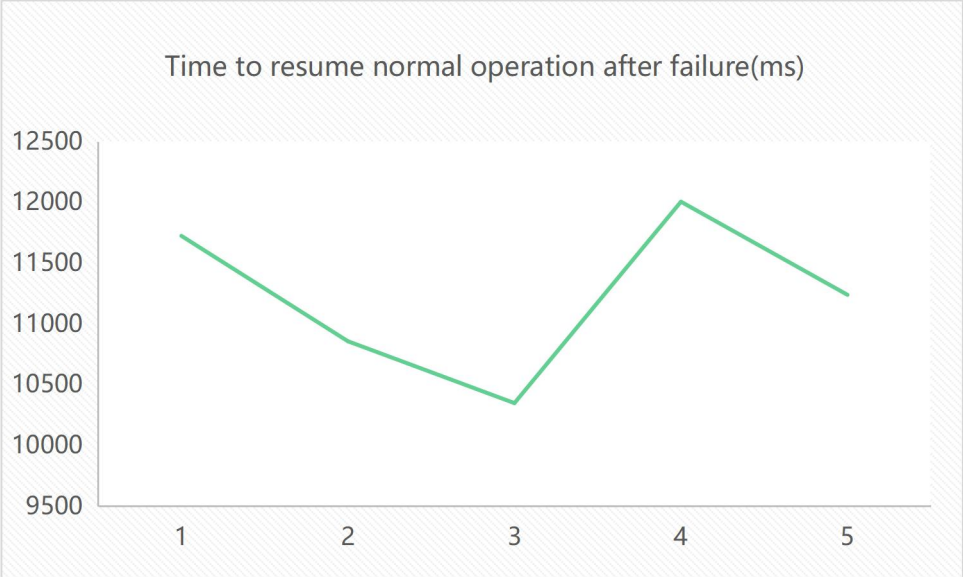
Discussion: The time deviation is low just as we predicted. Because it is a stable process. The process is that when coordinator receives a new job, it used batch size and flops to predict the ratio of processing time of two jobs, then it reallocates the number of VMs for each job. This process has no time-consuming operation. And coordinator will dynamically adjust the allocation of VMs.

2) After failure of one (non-coordinator) VM, how long does the cluster take to resume "normal" operation (for inference case)?



Average time: 9346ms
Discussion: The process is that when a VM crashes, queries which were on this crashed VM and have not finished would be redistributed to other workers. If "fair-time" is violated, the coordinator will dynamically adjust.

3) **After failure of the Coordinator, how long does the cluster take to resume "normal" operation (for inference case)?**



Average time: 11235ms
Discussion: Plots is reasonable. The back_up_coordinator and coordinator keep consensus in the data. The steps to resume "normal" are that: when back_up_coordinator monitors that coordinator crashes, back_up_coordinator would immediately broadcast the "elected" message to all other nodes. Then workers will return the result to new coordinator. The major part of time should be cost in detecting the failure of coordinator, which is about 9 s.