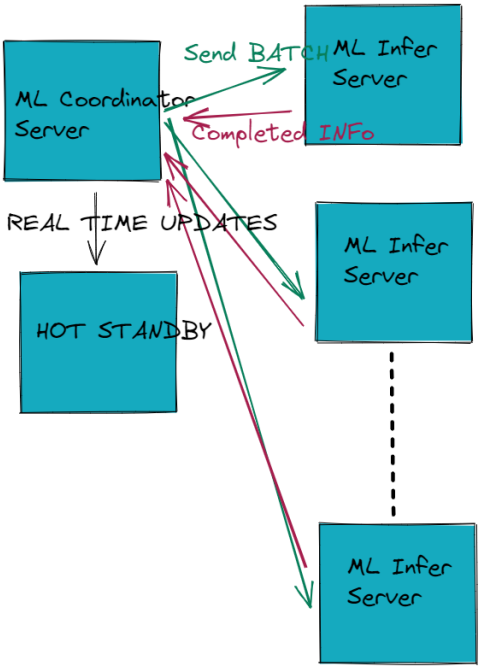
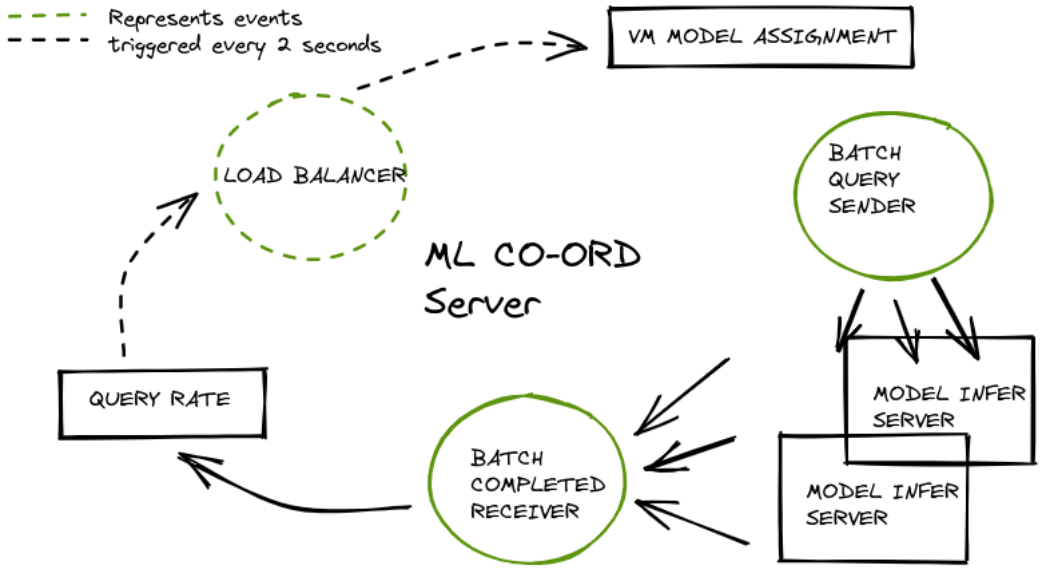
A Distributed Learning Cluster CS 425 MP4

SIMRAN SANDHU (sandhu5) Uday Kanth Reddy Kakarla(uk3)

Overall view:



Our design of IDunno cluster uses an ML Coordinator which starts the VMs in Inference phase in order to send query batches of images to ML Infer Model Server. We can schedule two kinds of Model Jobs Alexnet and ResNet(101) on the IDunno Cluster. The cluster can be in two phases: 'Training Phase' and 'Inference Phase'. The training phase is scheduled on dedicated VMs for each VM and the trained models are shared via Remote Procedure Calls (scp) to each Cluster Member (as detected by MP2). For the 'Inference Phase' the ML Coordinator Server schedules jobs on each of the Model Infer Servers with load balancing (as described below). The output is stored in SDFS(MP3) after each batch of query is processed, which ensures that even after failure of Model Infer Servers, the output of queries is not lost. Incase of failure of one of the Model Infer Servers, the MP2 detects it and informs ML Coordinator Server which is responsible for reassigning those queries and changing future job assignments. The Hot Standby gets live updates from ML Coordinator Server, and takes up the responsibilities of selecting a new ML Coordinator Server as soon as its failure is detected (with MP2). The Train Dataset was obtained from ILSVRC 2012 ImageNet Competition (https://image-net.org/), we used a portion of the training set (500 MB) and the Test Dataset was obtained from ILSVRC 2012 ImageNet Competition (https://image-net.org/), we used a portion of the testing set (310 MB). We used pretrained models (AlexNet and ResNet) from the torchvision models set https://pytorch.org/, and trained it using the train dataset. The average processing time per image for AlexNet and ResNet is 0.07 secs and 0.49 secs respectively.



Design Details The VM Model Assignment is started at the default predefined ratio 1:1. When a batch query is assigned to a VM the Coordinator keeps a track of it in the 'VM MODEL ASSIGNMENT' object. Once the Batch Queries are processed by the Model Infer Servers and the Model Coordinator Server receives the 'Completed' status from them, the query processing time is calculated with the help of:- ((send_time-recieve_time) / BATCH_SIZE). The load balancer is triggered every two seconds (to ensure we have collected sufficient data to make a reassignment decision) which modifies the VM Model Assignment to ensure that the query processing rate of the two models is within 20% of each other.

Programming Framework

We use an event driven framework with RPC calls to handle those events. There are broadly 3 events:-

1) Load balancer event -It is triggered every 2 seconds and modifies the VM Job Assignments to meet fair time inference.
2) ML Infer Server Fail Event - It is triggered every time ML Infer Server fails, the corresponding batch query logs are rescheduled and VM Job assignments are handled such that the ratio of query rates of ML jobs are within 20% of each other.
3) ML Coordinator Server Fail Event - The ML Coordinator Server's failure triggers the Hot Standby to multicast across the IDunno Cluster members to inform them of the new ML Coordinator Server ip address. The new Coordinator server takes up the ML Coordinator Server's responsibilities after initializing with the parameters (such as query rate and VM job assignment) which it received from the last update of ML Coordinator Server.

Observations

The following observations are made with 8 ML Infer Model Servers, 1 ML Coordinator Server and 1 Hot Standby.

1) Fair-Time Inference:

1) When a job is added to the cluster (1 → 2 jobs) :

1a) What is the ratio of resources that IDunno decides on across jobs to meet fair-time inference

	1	2	3	4	5	average	Std dev
Assignment Ratio (Calculated after 20 secs after starting both models)	0.143	0.143	0.143	0.143	0.143	0.143	0

The resources is assigned dynamically by IDunno in the ratio (1:7)

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?

	1	2	3	4	5	average	Std dev
Time (sec) taken by model to start executing queries	1.113	1.153	0.134	1.701	1.243	1.069	0.512

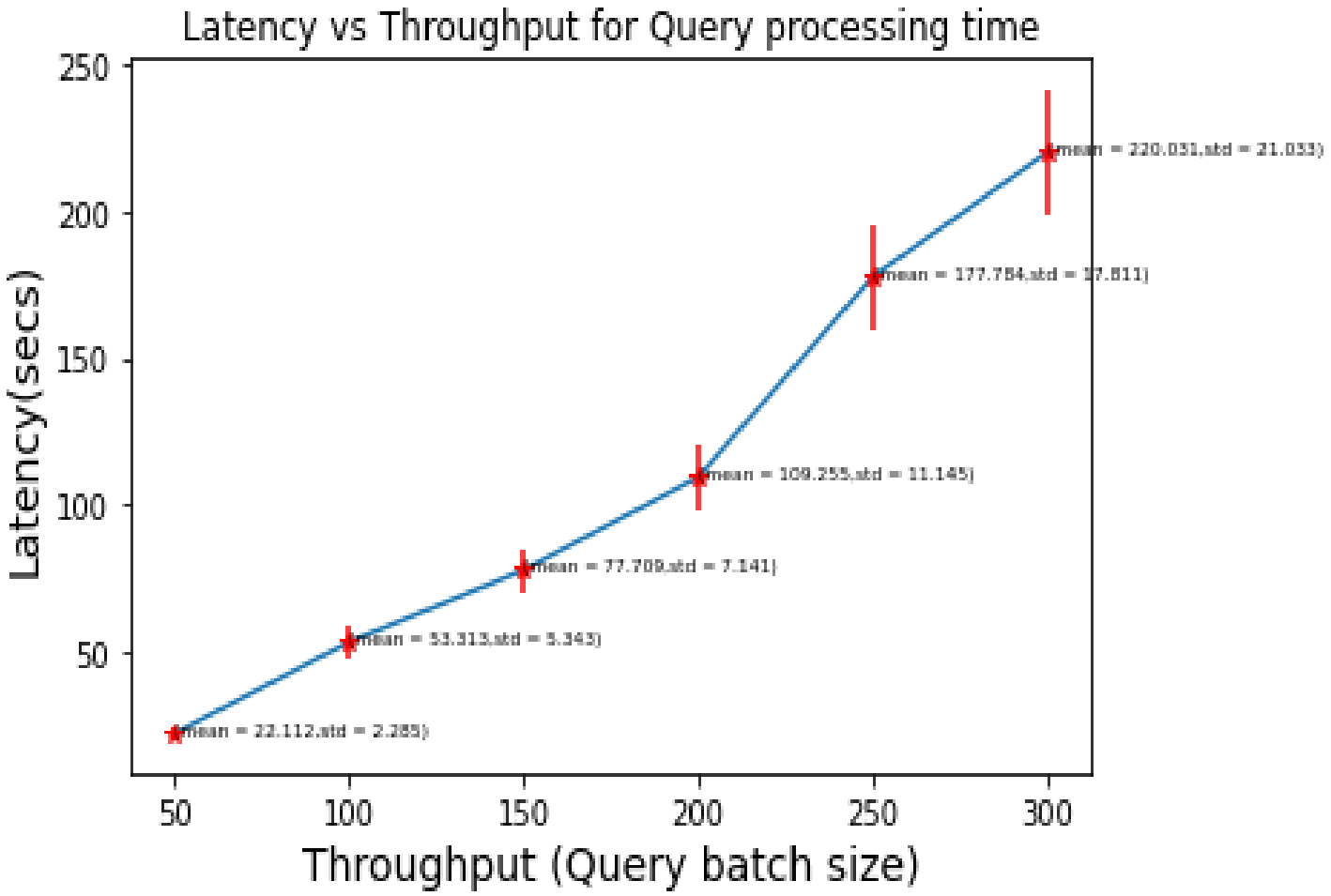
2) After failure of one (non-coordinator) VM, how long does the cluster take to resume "normal" operation (for inference case)

	1	2	3	4	5	average	Std dev
Time (sec) taken by model to start executing queries	3.442	3.411	3.515	3.496	3.379	3.449	0.051

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

These values include time to detect failure of coordinator and fetch parameters of the last update received.

	1	2	3	4	5	average	Std dev
Time (sec) taken by hot standby to take up role of coordinator	3.142	2.921	2.925	3.197	2.988	3.035	0.114



Applications:

The most Common application of IDunno Cluster is real-time inference of Models (especially Ensemble models which are very common to boost accuracy) in fields such as self-driving cars or medical surgeries. Since these medical surgeries and driving require the model to take very risky and instantaneous decisions, fair time inference is very important. An Ensemble model inference will require the models to be queried with roughly equal processing rates. Also fault tolerance is very important since any of the VMs doing inference could fail and it will be very expensive to lose the information already computed.

Conclusion

For fair time inference, the IDunno cluster behaves consistently after initializing both the jobs. For fault tolerance we observe that the cluster behaves normally after approximately 3.5 secs after failure of non-coord and 3.00 seconds after failure of co-ordinator server. For latency vs throughput we observe that up-to batch batch size 200 of queries the latency varies linearly with the throughput, however, as we increase the batch size beyond that, the latency increases much faster which is most likely because of high network traffic.