Paper preparation

First of all thank you very much for listening to my paper report

Our paper is Distributed machine learning task scheduling based on Lyapunov optimization

The paper focuses on a solution about how to schedule machine learning tasks reasonably in a distributed environment.

first，we need to talk about

Cloud cluster environment

In recent years, big data technology has developed fast, and application data in various industries has shown explosive growth. Cloud cluster environments provided by enterprise are constantly emerging in our industrial market. These clusters have the characteristics of public availability, service orientation, and resource limitiations. For our service providers, what they pursue is how to maximize their own benefits while meeting the needs of users. This is also the driving force for our experiments, how to use limited resources to gain benefits.

then we will talk about

Features of machine learning

Machine learning has been a very fast development direction in recent years, and the scientific community is also increasingly investing in this area. Most of our regular machine learning tasks contain the features of repetitive computing, large data scale, and independent calculation.

Take our most commonly used CNN model as an example. In the calculation process, a large amount of data needs to be repeatedly calculated to obtain the final results. Most of our subsequent experiments are also discussed based on the CNN model.

so this is

Scheduling strategy

The current distributed scheduling strategy generally has two concepts: data parallelism and model parallelism

In our research, we chose to apply the idea of data **parallelism.**

Data parallelism means to divide data into different computing resources (nodes) so that different nodes can process different data: Since we have many nodes processing data in parallel, we should be able to process more at the same time. So that we achieves the purpose of parallel computing.

But we will also meet some problems, such as whether for any machine learning task, distribution is the best solution strategy, and whether we can make dynamic decisions based on the environment, whether the current cluster environment allows Distribution

Problems we solve

In our paper, we focused on solving a few problems

first,How to determine when a machine learning task is used for distributed

second,In an environment where cluster resources are limited, how do we solve the relationship between the optimal scheduling strategy and the quality of service?

third,How to apply the Lyapunov optimization problem to machine learning tasks for effective attempts

there are some graphs about

The relationship between the calculation time of the general model and its batchsize

In our experiment, in order to measure the calculation time of the machine learning task, we will estimate it based on the calculation parameters provided by the user.

We found in experiments that on the GPU hardware devices, because the calculation time of the model has a strong relationship with the batchsize set by the user.So we estimate the model first, and calculate its value when the batchsize is 1.

In this way, we can estimate the tasks' real time based on the formula we fit.

The result in the figure is a graph of our theoretical calculation time and actual calculation time. It can be seen that the two curves are basically consistent.

this page is about

Machine learning task parameter calculation

These are what we need to consider when performing distributed calculation, because in theory, distributed machine learning will greatly reduce the time due to the distrubution. but the additional overhead in the calculation process is also we have to consider, so we have to consider some parameters that we need to send in the calculation process.

It is in our analysis of Nvidia's paper on FLOPS calculation and found that there are basic estimates for the model of machine learning tasks. we take the CNN model as an example， and we combined the conclusions in the paper and our own real machine test, , the task of calculation is mainly reflected in the convolutional layer and the fully connected layer. Therefore, our experiment believes that only a few parameters of the model need to be considered, and the details of each layer of the model are not considered, which will not have much impact on the results.

In the figure we show the total parameters of some models

We can see that different models will have different parameters, and then under the conditions of different sizes of our data sets, it may have a certain impact on whether we make distributed decisions.

and we need to think about

The balance of communication and calculation

For each task, if you want to perform distributed computing, we need to consider communication time while considering computing time. For stand-alone computing, we don't need to consider it.

Through the previous experiment, we can obtain the calculation time required by the task in the case of the batchsize set by the user, according to the estimated situation when the batchsize is 1.

Later, we can estimate the size of the model parameters on the previous page. Combine the communication bandwidth of the cluster to obtain the communication overhead of the task in the process of distributed computing.

Then we sum the two to get the total time required by the task when using different GPU numbers. In this way, we can know whether the specified task is suitable for distributed computing.

then we will talk about our experiment,first is

Task queue design

In our scheduling process, our task queue backlog is the most important part for every decision we make.

In our experiment, we want to use the Lyapunov optimization model to solve the problem of machine learning scheduling. But in general research, this optimization method is used to deal with data flow problems.

The difference between the two is that the transmission of the data flow can be interrupted at any time, and once the machine learning task is decided to enter the cluster, the task as a whole needs to be put in at one time.

In our design, we set up two queues, one is used to represent the actual task queue, and another is the virtual queue. we will explain the two queues next.

The first is the actual task queue. Since machine learning tasks are not like data flow, the size of the task can be simply measured by the size of the data. Machine learning tasks have very big differences in many aspects due to their models, data sets, and setting parameters.

Therefore, we use the tasks' estimated computing time as a measure of the task queue length.

In this way, the length of each task in the queue will be determined by its calculation time. How to obtain the estimated calculation time for each task is also mentioned in the previous page.

With regard to the design of the virtual queue, the idea is as follows.

We consider a situation, that is, the task calculation time is not very long, and after the calculation, it is found that the distributed task can obtain greater benefits. But assuming that the resources have not been released at this time, would we be willing to spend more waiting time to make the task be the most optimal Scheduling?

So our virtual queue is used to measure the waiting time of tasks in the cluster.

From this point of view, because the estimated calculation time of the task does not change over time, in our scheduling process, the length of the task in the queue is constant, that is, static. Since our virtual queue is related to the tasks' waiting time. each time we make a decision, the virtual queue will also change, which is dynamic.

by analyzing the situation of the two queues at each scheduling moment,we will made decision.

then is our Formula analysis

Our optimization goal is to minimize the tasks' total time.

Through the previous analysis, we can obtain the tasks' calculation time and the basis of the tasks' communication time, so we can calculate whether different machine learning tasks are suitable for distributed computing.

The second is about the design of our two queues. we need to make decisions at every scheduling moment:

For the actual task queue, we use the parameter k to indicate whether the task is scheduled at this moment. Therefore, the values of k are only 0 or 1. Then in the queue, we not only have to consider the tasks that can be scheduled at this moment, but also the tasks that arrive at the cluster in this moment, so we also need to think subsequent tasks.

For virtual queue, our idea is that this queue will represent the tasks' waiting time in the cluster. The longer the virtual queue length, the longer the tasks' waiting time in the queue. In this case, in order to meet the requirements of service quality as much as possible, we will try our best to put tasks into the cluster as early as possible to improve the accumulation of queues.

Since the two queues are critical to our scheduling analysis, according to Lyapunov's optimization ideas, we merge the queue formula.

Second page

Scheduling is carried out at every moment. **In the process,** we also made some modifications to the Lyapunov optimized formula, and the rest of the derivation steps are no different from conventional ideas.

The final decision formula is as follows, except that some variables are modified based on the our machine learning model's feature , and the rest have not changed.

For our final decision result, we need to explain here

First, we believe that when there is no scheduling, we cannot obtain benefit from scheduling, and it will not affect the queue at this moment, so we believe that the total benefit at this time is 0.

based on this, our scheduling will have two situations. One is that after scheduling, we can get benefits from it. In this case, we will make scheduling decision at this moment. The other is that after scheduling, we cannot obtain benefit but may even get additional overhead. Then in this case, we may tend to do nothing at this moment, so that our benefit is 0. After that, we can postpone the decision to the next moment.

What needs to be pointed out here is that **postponement** means that the virtual queue becomes longer. In this case, we will have a greater chance of scheduling tasks in the next scheduling moment. This is also our solution strategy to prevent tasks from being unable to be scheduled due to some special situation.

the last is our Experimental results

This set of experiments is based on our previous small-scale calculations on real computers, and then used simulated scheduling experiments to complete.

The experiment settings are divided into four groups,

The proportion of small data tasks is more, and the arrival is scattered .and the second is the same task,but be centralized arrival.

another test group is that large data tasks have a large proportion .

The comparative experiment we took included fixed single GPU calculations and calculated with random GPUs.

From the experimental results, we can see that, except for the last experiment group, there is no obvious difference between the three scheduling methods. In the other three cases, our scheduling strategy can produce good results. The results of the experiment is also proves that in most cases, our scheduling strategy can produce better results.

We also **analyzed** the last experiment whose results were not good. We think because we consider a more extreme situation.In our normal scheduling process, it is actually difficult to meet a situation where a large number of big data computing tasks arrive in a short time.

In this case, due to the limitation of resources, the tasks in the calculation are not released due to the long-term occupation of resources. It leads to the continuous increase of waiting tasks. And because of the features of tasks, most of them have long computing time, so our two queues will be easy in a bad situation. At this time, our scheduling strategy will tend to schedule tasks as early as possible to reduce the length of the queue.

In this case, after a long time scheduling, the task scheduling will approach this behavior. Once the resources are released, then if scheduling at this moment, regardless of whether distributed computing is performed, The behavior will always get benefits, so it will directly schedule the tasks .

Generally speaking, in the later stage of scheduling, our scheduling strategy will inevitably tend to be completed by one GPU for one task. This also reasonably explains why in the last case, the time spent on the three strategies is almost the same.

Finally, thank you very much for listening. This is the end of my report