Dynamic I/O Model Recommendation System With Machine Learning

*Note: Sub-titles are not captured in Xplore and should not be used

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—In a typical database and file system, using asynchronous I/O is generally a good way to optimize processing efficiency. However, each process and application mainly focus on its own performance, rarely consider the global performance of the system. In this work, in order to balance the I/O performance and resources in a system, we use Machine Learning(ML) techniques to learn I/O model's performance, and set up a client/server system based on gRPC to recommend the more efficient I/O model under different system loads. Which is high performance, scalable, cross-platform and easy to adapt. The experimental result shows that our system has a 15% performance improvement compared to using asynchronous I/O alone and only cost little system resources.**

***Keywords*—asynchronous I/O, synchronous I/O, Machine Learning, performance prediction, gRPC**

I. Introduction

With revolution of "Big Data" and "Cloud Computing", data center has expended to a large scale and data has grown exponentially. Therefore, data center has to process hundreds of millions of pictures and hundreds of billions of messages each day. How to improve processing efficiency is a hot issue of a data center. Due to the huge speed gap between CPU and I/O device, I/O is one of the bottlenecks of the issue. Using asynchronous I/O is a common way to boost I/O speed. Synchronous and asynchronous I/O are two types of I/O synchronizations as ***figure 1*** shows. In a synchronous I/O job, it starts a thread for I/O operation, and it would hang immediately until the operation is finished. While in an asynchronous I/O job, it would start a thread to send a I/O request to Kernel by calling a function, if the request is accepted successfully, it continues to process other jobs. The kernel signals the calling thread when the operation is finished, then the thread interrupts its current job and processes the data from the I/O operation as soon as possible. However, using asynchronous I/O frequently requires much CPU resources and the I/O's latency would become higher when the I/O's depth is longer.

To help improve system I/O and processing efficiency, we purpose to combine I/O with Machine Learning(ML).

Identify applicable funding agency here. If none, delete this.

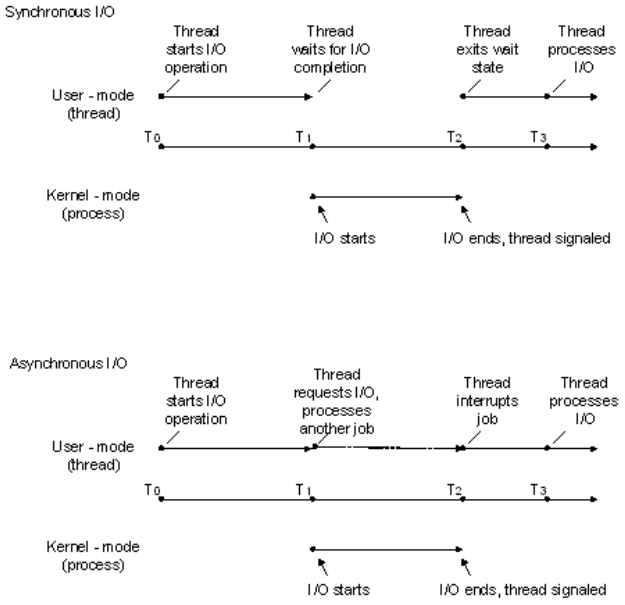

Fig. 1. Asynchronous and Synchronous I/O

The I/O synchronizations of each I/O's engine have different performance when it is facing different kinds of I/O job and different system load. Collecting these I/O data and training into a Decision Tree model which can predict the most suitable I/O method in current situation. And setting up a Client/Server-based recommendation system using gRPC framework to decide using which kind of I/O for each I/O job.

The main challenge of our system is performance, since each I/O job have to call the server for the better I/O method which cost much time on process communication and prediction. To solve the related problems, we use memory cache and gRPC framework. The data is serialized to protocol buffers and passed by stream, meanwhile, store the hot data into cache. Therefore, the communication's time will be shortened and reuse the prediction result. After the improvements applied in our system, Running the same task has an efficiency increase

of nearly 15%.

In summary, we have made the following contributions in our paper:

1) Train a Decision Tree model which can predict the best I/O method base on the current system load.
2) We build a Client/Server-based, lightweight and scalable system using gRPC to help each I/O job improve efficiency.
3) Run a script to make the system self-optimization daily.

The rest of the paper is organized as follow.

II. Design

In this section we talk about the goals of our system first and then show the overview architecture.

A. Goals

- High-performance.
- Scalable.
- Self-optimization.

B. Overview

figure.

C. Techniques

- Machine Learning
- Decision Tree
- gRPC
- atomic
- redis

III. Implementation

- collect data
- train data
- build the system
- test the system

IV. Evaluation

- io-uring performance
- compare single I/O work performance between used and none-used our system
- compare multi I/O works performance between used and none-used our system

V. Related Work

- hot issue in I/O

VI. Conclusion

improvement of our system and future usage scenario

Acknowledgment

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.