Improving sample based cluster scheduler with learning based private-cluster-state

Chunliang Hao†, Yinrun Lyu†, Jie Shen‡, Celia Chen§, Jian Zhai†, Mingshu Li†

† Institute of Software, Chinese Academy of Sciences, Beijing, China
‡ Department of Computing, Imperial College London, London, UK
§ Center for Systems and Software Engineering, University of Southern California, California, USA
Email:{chunliang, yinrun}@nfs.iscas.ac.cn, js1907@imperial.ac.uk, qiangqiac@usc.edu, {zhaijian, mingshu}@iscas.ac.cn

Abstract

One major limitation of sample based scheduler is lacking global knowledge of the cluster, which leads to precision problems and incapability towards some important scheduling concerns. Instead of using a centralized component/scheduler to collect global state and push to distribute schedulers, this paper suggest that each scheduler could independently predict the real-time cluster state simply by accumulate, organize and learn its own process data. For which purpose we introduce the LPCS technique. Preliminary experiments on Google trace shows that with LPCS, the scheduler is able to generate a precise approximation of real-time cluster state for decisioning support.

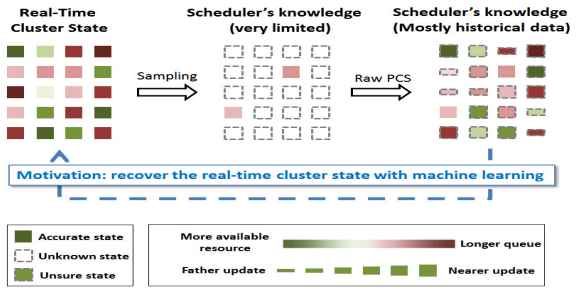

Figure 1. A demonstration of precision problem in sample based cluster scheduling and LPCS rationale, using 20-node cluster.

1. Introduction

Sample based scheduling is a typical distribute scheduling method that is both low-latency and scalable(4). Its core idea is using a small random subset of cluster (a.k.a sample cluster) instead of the entire cluster for quick decisioning, and redraw the sample for each decision to achieve load balance. This simple decisioning process allows multiple schedulers co-functioning in the same cluster without the need of coordination. However, it suffers from severe precision problem(2). The impreciseness comes from the fact that sample based scheduler does not have any global knowledge about global cluster state, hence sometimes it makes sub-optimal scheduling decisions, and it is incapable of supporting some important scheduling features.

Our previous research, private-cluster-state (PCS) (3) shows that simply accumulating and organizing private process data during sampling could also in some degree mitigate this precision problem at low cost. In this research we take a step forward to introduce LPCS, which generate an precise approximation of real-time cluster state from PCS data with the help of machine learning.

2. LPCS

We introduce a simple example to explain the system context of LPCS and its rationale. As shown in figure 1, the top left cluster state (of a 20-node cluster) is the real-time snapshot, which ideally a scheduler should possess in order to make precise decisions.However for a sample based scheduler in the top-middle, it could only communicate with few randomly sample nodes. In this figure, the top-middle scheduler is about to make sub-optimal decision since out of randomness it samples two busy node for this decision, hence it must choose a busy node while the cluster has plenty of other nodes with available resources. With PCS, as shown in top-right of the figure, scheduler is able to accumulate its own process data and form a imprecise cluster state. Since each node's information in this imprecise state has different time of update (represented by height in this figure), some recently updated data might be quite precise while some father updated data might has already expired (mostly cause by task assignments from other schedulers).

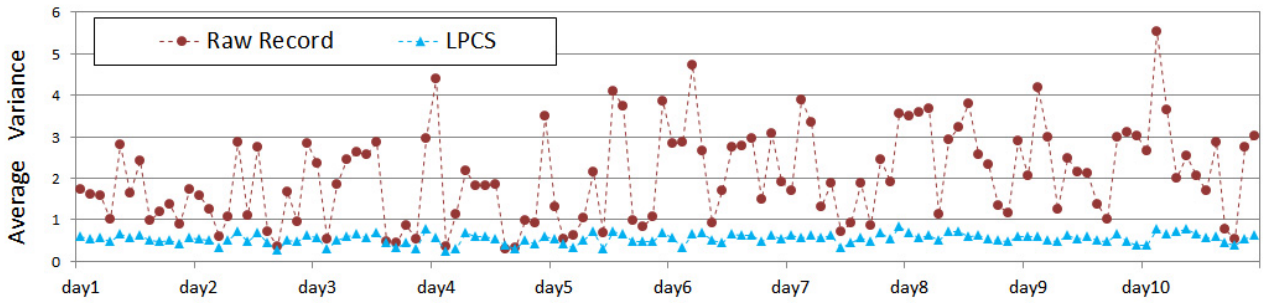

Figure 2. Google trace results, 10 days, evaluating the average variance between LPCS speculation and real-time cluster state. Using scheduler accumulated raw historical data as baseline to show improvements gained from machine learning.

On top of these context, we introduce LPCS, which suggests that each scheduler could learn how worker state changes through time, so that each scheduler could converts its PCS data to a precise speculation of real-time cluster state.

LPCS is implemented in each scheduler as an inner component, it has three major parts: 1. An on-line decision advisor 2. A private state accumulator 3. An off-line model updater. The on-line advisor use prediction model from off-line updater and data from private state accumulator to predict the real time resource state of one or several nodes, then advice the scheduler based on the prediction. The private state accumulator record and organize PCS data, the detail of this process could be found in our PCS research. Off-line model updater is responsible for periodically update the prediction model. Each time scheduler communicate with one worker, it send feedback containing received state of that worker to model updater, these feedback will be used as part of the training data in periodical update.

The learning process inside off-line updater work as follows: LPCS define five cluster context according to utilization, very low, low, medium, high and extreme. For each context LPCS learns a correlation between historical worker resource state and real-time worker state. LPCS use linear regression model and the following features: resource state of worker node n in private state $\vec{r_n}$, update time of $\vec{r_n}$, worker node n's real-time resource state $\vec{R_n}$, estimated cluster utilization calculated from scheduler's private state and task throughput (of the current model update period). Periodically, data of the recent 72 hours in the updater will be used to train new model. New model will be pushed to the on-line advisor to help the next period's scheduling.

3. Preliminary Experiment

We evaluate the performance of LPCS in a simulated cluster, using publicly available Google trace(1). A-1000 node cluster is used, each node is set to have 20 resource units. 5 schedulers with LPCS are deployed in the cluster. We use 3 days of data from Google trace to train the starting model. We use the following 10 day's data to test the performance of LPCS. The off-line model update period is set to 2 hours.

Every time when a scheduler communicate with a worker node, we record two variances: one is between real-time resource state of worker node and the scheduler's stored raw historical record of that node; the other variance is between real-time node state and LPCS speculation results. We display the average of two variance in figure 2. The figure shows that comparing to raw historical data, LPCS speculation reduce most of the variance. With LPCS the average variance in figure is always below 1, in contrast without learning the results are mostly above 1 and could reach as high as 5.6. These results support our claim that through LPCS, the scheduler could effective learn the correlation between historical data and real-time data, and is able to generate a quite precise approximation of real-time cluster state accordingly.

To summarize, with the help of LPCS, a sample based scheduler has a much broader and preciser view of the cluster. We also conduct experiments to verify how such global knowledge would help the final scheduling outcome, preliminary results shows that it could improve about 11% of 90th percentile job runtime in Google trace simulations, related experiments will be introduced in future full paper.

References

[1] CHEN, Y., ALSPAUGH, S., AND KATZ, R. Interactive analytical processing in big data systems. In *VLDB* (2012), pp. 1802–1813.

[2] DELGADO, P., DINU, F., KERMARREC, A.-M., AND ZWAENEPOEL, W. Hawk: Hybrid datacenter scheduling. In *USENIX ATC* (2015), pp. 499–510.

[3] HAO, C., SHEN, J., CHEN, C., ZHANG, H., WU, Y., AND LI, M. Pcssampler: Sample-based, private-state cluster scheduling. In *CCGrid* (2017), pp. 599–608.

[4] OUSTERHOUT, K., WENDELL, P., ZAHARIA, M., AND STOICA, I. Sparrow: distributed, low latency scheduling. In *SOSP* (2013), pp. 69–84.