Semi-automatic Assessment of IO Behavior

1st Eugen Betke
DKRZ — Hamburg, Germany
betke@dkrz.de

2nd Julian Kunkel
University of Reading — Reading, UK
j.m.kunkel@reading.ac.uk

***Abstract*—In this work, we develop three IO metrics to automatically detect relevant IO behaviors in timeseries of I/O statistics and evaluate them on raw data of 1,000,000 jobs on the Mistral supercomputer. For that purpose, the job runtime is segmented into fixed size time windows and analyzed. The advantages of this strategy is that temporal information about IO activities during job runtime is preserved and can be used for metric computations. The main contribution of this work is a proposal for mapping of IO statistics into computable metrics: IO-Mean-Score, IO-Problem-Time, and IO-Balance. On two examples, we show how these metrics can be used for identification of problematic and IO intensive jobs.**

I. Introduction

There is a variety of parallel applications that run on HPC systems. In some research areas, like climate simulation, they access a huge amount of data. Each application has its own specific IO behavior. Depending on the application, we can observe typical patterns like parallel/serial IO, bursts, checkpointing, write/read phases, and compute/IO phases. Some patterns are efficient, hence achieving high IO performance and small application runtime. We need to verify that IO is done efficiently. However, checking every application manually is not an option. Profiles often compress the temporal behavior of applications too much and cannot reveal short-term inefficiencies. Our approach is to define meaningful metrics that compute relevant information from segments (timeseries) of the application runtime.

II. Analysis tool

The analysis tool is a product of our continuous research on monitoring data analysis. It requires an initial training, based on a relatively small job dataset. Hence, it downloads job data from monitoring database and creates statistics about IO performance on HPC. In the second step, these statistics are used for computing statistics and assessing individual jobs.

DKRZ's monitoring system provides 5 metadata metrics and 7 data metrics. The source for metric data are counters in Lustre proc-files `stats` and `read_ahead_stats`.

III. Methodology

The concept of this work relies on job runtime segmentation, whereby we choose to split raw metric data into 10 min time windows (segments). Furthermore, for each segment we calculate average performance values. We then map segments into one of three categories (LowIO, HighIO, CriticalIO) based on statistics analysis of a relatively small dataset using the 99- and 99.9-quantiles. IO performance values below q99 are assigned to LowIO, values between q99 and q99.9 are assigned to HighIO and values higher than q99.9 to HighIO. As a positive side effect, this categorization strategy unifies metric units and allows aggregation of different, formerly non-compatible, metrics. We sum up the different metrics to form statistics on node and job level.

Metric segments are basic blocks for the computation of the following derived metrics: *IO-Problem-Time (PT)*: This metric shows the fraction of time a job spends for IO. This is approximated by the fraction of time windows with I/O activity. *Mean-Score (MS)*: This is the average load during IO-relevant phases. While most phases may not do any IO, some might have extraordinary IO activity during such phases. Large jobs can induce slow down on the file system for other jobs. *IO-Balance (B)*: This metric indicates how IO load is distributed between nodes during IO-relevant phases.

IV. Evaluation

For evaluation, we use the statistics of 1,000,000 jobs that we downloaded from DKRZ's monitoring system. These data cover a time period of 99 days. For the initial training, we use data of 30 days.

In our experiments, by ranking the different metrics, we could identify the most I/O crucial jobs. The IO-Balance identifies jobs with a small number of IO-nodes, because it is shrinking with increasing job size. PT and MS can be used as weights in mathematical calculations.

a) Problem-Score $= (1 - B) \cdot PT \cdot MS$*:* shows jobs with most optimization potential. It is a product of all metrics, in a way that bad behaviour increases its value. The top 5000 jobs ranked by Problem-Score, and job runtime $>$ 30 min, is directly reduced to a few jobs that bear most optimization potential. A closer look reveals that these are mostly unbalanced but I/O intensive jobs.

b) IO-Intensity $= B \cdot PT \cdot MS \cdot total\_nodes$*:* identifies applications that generate high IO loads. In the calculation, we have also to consider the number of nodes. A top 40 list of jobs with runtime $>$ 30 min, ranked by decreasing IO-Intensity, was occupied mostly by two applications. A closer look a the raw data, verifies high IO load.

V. Conclusion

We introduce HPC relevant metrics based on a uniform segmentation process. The method takes the temporal activities during job execution time into account. It complements the conventional profile-based I/O performance analysis.