Predicting Performance of Non-Contiguous I/O with Machine Learning

Julian M. Kunkel
DKRZ
kunkel@dkrz.de

Michaela Zimmer
University of Hamburg

Eugen Betke
University of Hamburg
eugen.betke@gmail.com

ABSTRACT

Data-sieving in ROMIO promises to optimize individual non-contiguous I/O. However, making the right choice and parameterize its buffer size are non-trivial, since performance prediction is difficult. Moreover, since many performance factors are not taken into account by data-sieving, the optimization towards access pattern and system is often not possible.

In this work, we 1) discuss limitations of data-sieving, 2) apply machine learning techniques to build a performance predictor and 3) introduce a workflow to learn and apply optimal parameter settings. Even though this initial research is based on decision trees, with sparse training data the algorithm can predict many cases sufficiently. Applying the scheme to a set of experimental data improved throughput, consistently achieving between 25% and 50% of the the best parameterization's gain. Additionally, we demonstrate the versatility of this approach for creating expert knowledge by extracting rules from the learned tree.

With MPI 2, an I/O interface has been standardized which promises to improve performance for parallel applications. Among the supported features, it explicitly supports non-contiguous I/O – one API call accesses multiple file regions, and, with collective I/O, multiple processes can coordinate their file access. The standard explicitly allows an implementation to exploit its knowledge about concurrent operations; for example, by scheduling the I/O calls intelligently. Since there are many factors influencing performance in a supercomputer, extracting the best performance is anything but trivial. The available optimizations offer a selection of parameters to be adapted to target machine and specific workload, and through a wrong choice, performance may be degraded.

To alleviate this, our research strives to provide a tool that will be aware of system capabilities as well as its performance history, using all to suggest the best parameter set for the task at hand. Our main contributions are: 1) A workflow developed for the SIOX framework which can monitor and learn performance and apply optimization parameters. 2) The evaluation of decision trees to capture and predict non-contiguous performance behavior.

The SIOX framework allows for an integrated approach which extracts a performance model and applies it to set appropriate MPI hints. As Figure 1 shows, applications instrumented for SIOX will deliver access and performance data, to be processed by SIOX proper and archived to a trace file or a data warehouse. From there, a trace reader will feed them into machine learning plug-ins, which in turn will write their findings to files or a knowledge base. These are then used to control the behavior of the optimization plug-ins.

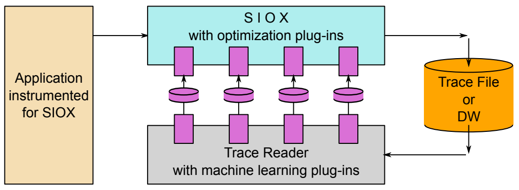
Application instrumented for SIOX
SIOX
with optimization plug-ins
Trace File or DW
Trace Reader
with machine learning plug-ins

Figure 1: Optimization workflow with SIOX. The purple boxes represent plug-ins communicating via files or entries in a knowledge base

This processing cycle permits automatic supervision, analysis and optimization of processes. But by its power to set parameters during operations and its knowledge of the data available, this system is even capable of active learning, conducting automatic explorations of the parameter space to improve its efficiency.

Previous attempts at HPC self-optimization all required human intervention at some point. The SIOX framework offers the first fully autonomic comprehensive cycle, encompassing collection, analysis and application of optimizations found. Our experiments demonstrate the workflow necessary and prove the validity of the approach: Even a simple plug-in relying on stock ML techniques can serve as a fairly accurate performance predictor. The decision trees learned in the process reproduced known expert knowledge correctly but also yielded new and interesting insights into best practices for data-sieving on our system. By automatically generating simple rules-of-thumb from the extensive decision trees, expert knowledge can be generated. Applying the results during online optimization finally increases the average I/O performance by several MiB/s. Active learning during phases of low utilization can supplement sparse training data to greatly improve predictor accuracy and overall effectiveness.

We are currently working on an adaptive data-sieving algorithm relying on the additional parameters discussed in this work. Once implemented, more accurate representations and characteristics of the resulting parameter spaces will be researched. Future efforts will further explore ML techniques and their applicability, as well as the effects of selective data acquisition and active learning.