

Performance database technology for SciDAC applications

D Gunter¹, K Huck², K Karavanic³, J May⁴, A Malony², K Mohror³, S Moore⁵, A Morris², S Shende², V Taylor⁶, X Wu⁶, and Y Zhang⁷

¹Lawrence Berkeley National Laboratory, Berkeley CA 94720 USA

²University of Oregon, Eugene OR 97403 USA

³Portland State University, Portland, OR 97207 USA

⁴Lawrence Livermore National Laboratory, Livermore CA 94550 USA

⁵University of Tennessee, Knoxville TN 37996 USA

⁶Texas A&M University, College Station TX 77843 USA

⁷Renaissance Computing Institute, Chapel Hill NC 27517 USA

Corresponding author e-mail: shirley@cs.utk.edu

Abstract. As part of the Performance Engineering Research Institute (PERI) effort, the Performance Database Working Group, which involves PERI researchers as well as outside researchers at the University of Oregon, Portland State University, and Texas A&M University, has developed technology for storing performance data collected by a number of performance measurement and analysis tools, including TAU, PerfTrack, Prophecy, and SvPablo. In addition to the performance data, metadata capturing the experimental setup and conditions (e.g., source code version; input data; platform, compiler, library, and operating system versions and configurations; runtime environment) are exported to a common metadata schema, along with some basic performance information. The exported information can be viewed from a common web interface, and a link or contact information is provided for accessing the original performance data in its home database. Analysis tools provided by the individual databases support tasks such as parallel profile browsing and analysis, cross-experiment analysis, and scalability studies. Performance data are currently being collected and analyzed for the GTC and MILC SciDAC applications. The tools are being installed on machines used by SciDAC researchers so that they can easily collect data and upload it to an associated performance database. Work on a deeper level of interoperability that will allow exchange of actual performance data between databases is underway.

1. Introduction

Empirical performance evaluation of parallel applications can generate significant amounts of performance data and analysis results from multiple experiments. To better manage performance information, several performance database systems have been developed that support storage and retrieval of both performance data and accompanying metadata that may be collected by various

⁵ Author to whom correspondence should be directed

performance measurement tools. The tools and database systems have begun to be used by Performance Engineering Research Institute (PERI) researchers as well as SciDAC applications developers. As part of the PERI effort, a working group was formed to achieve interoperability between the different performance database systems. The remainder of this paper describes the interoperability goals and approach, as well as the participating databases and examples of their use with applications.

2. Interoperability approach

The first interoperability goal was to allow users to search and retrieve datasets from any of the participating performance databases through a common interface. This goal has been met by defining a common export format for the performance *metadata* and displaying the metadata for each data set, along with a pointer to the host database, in a format accessible by a web browser.

2.1 Run rules (*metadata*)

The performance metadata describe the context of how the code was run, or the "run rules". Several categories of information are included, as described below:

- *Source code.* Exact version of all components of the source code. This may be done by storing the source files themselves, storing a secure hash of each source file, or tying the versions used to a version control system.
- *Automatic transformations.* Automated transformations that are applied to the source code or intermediate representation, including source-to-source preprocessing and replacement of portions of the source code with automatically generated code.
- *Compiler environment.* The exact version of the compiler and the compiler options, as well as any configuration information for the compiler environment.
- *Libraries.* The exact versions of all libraries linked with the application code, including dynamically linked shared libraries.
- *Input sets.* The exact versions of all input files used. This may be done by storing the input sets themselves (if they are small), storing a secure hash of each input data file, or tying the versions used to a version control system.
- *Performance instrumentation.* The exact steps carried out to instrument the code. In the case of source code instrumentation, this information may be included in "Automatic transformations" instead.
- *Runtime environment.* Information such as job start time, hardware and OS version, file system version, scheduler queue contents, environment variables, and the nodes assigned to the job.

2.2 Metadata schema

The above run rule metadata specifications are formally defined by means of an XML schema. The schema and example conforming XML instances are available on the PERI wiki pages at http://peri-scidac.org/wiki/index.php/Perfdb_XML_Schema.

2.3 Web interface

The metadata and some basic performance data are exported to a common web interface that can be searched or browsed to locate one or more relevant data sets. The interface provides a tabular display of some selected attributes of matching data sets, including the address of the associated performance database and the associated record key. The full XML metadata can also be browsed. The web interface, or performance database "portal", is available at the PERI main website at <http://www.peri-scidac.org/perci/perfdb/search>.

2.4 Future work

Work is underway on a deeper level of interoperability that will allow actual data to be exchanged between participating performance databases. To that end, the participating tool groups have developed a data dictionary that identifies similar performance data fields from their respective databases. The next step will be to define a common data model to which each database can translate their native data.

3 Performance analysis and database tools

3.1 PerfTrack

PerfTrack is an experiment management tool for collecting and analyzing parallel performance data [1]. PerfTrack comprises a data store and interface for the storage and retrieval of performance data describing the runtime behaviour of large-scale parallel applications. PerfTrack uses an extensible resource type system that allows performance data stored in different formats to be integrated, stored, and used in a single performance analysis session.

3.2 Prophecy

Prophecy [2] is a web-based infrastructure for the performance analysis and modeling of parallel and distributed applications. Prophecy consists of three major components: data collection, data analysis, and performance databases. The data collection component focuses on the automatic instrumentation of codes at the level of basic blocks, procedures, or functions. The entities in the database are organized into four areas: application information, executable information, run information, and performance statistics. The data analysis component can produce an online analytical performance model with coefficients, at the granularity specified by the user via web-based interfaces. Prophecy allows for the development of linear as well as nonlinear models. These models, when combined with data from the system database, can be used by the prediction engine to predict the performance on a different compute platform.

3.3 TAU and PerfDMF

TAU is a portable performance system supporting all DOE HPC platforms (including IBM BG/L and Cray XT3) and all dominant programming languages, compilers, thread libraries, and communications libraries for HPC software development [3]. TAU components include: multi-level performance instrumentation, multi-language automatic source instrumentation, flexible and configurable performance measurement, and a widely-ported parallel profiling and tracing system. TAU's Performance Data Management Framework (PerfDMF) provides an interface to an underlying relational database for storing profile data and a profile query and analysis programming interface. PerfDMF includes importers from other tools and a flexible interface schema to provide a unified performance database system for different input sources.

3.4 SvPablo and HPC Database

RENCI HPC Database is a web-based infrastructure for storing and processing performance data collected by SvPablo [4], a graphical environment for source code instrumentation and performance analysis for scientific applications on various HPC systems. The database provides web interfaces for users to browse performance data, perform statistical analysis, conduct performance comparisons across platforms, and study the different characteristics of various existing HPC systems.

4 Application examples

Project participants have collected performance data for the MILC and GTC applications on a number of platforms. Metadata describing the runs, including contact or link information to the exporting database, are available from the web interface described in section 2.4. Although the metadata are publically available, the performance data are password-protected. Three examples are described below.

4.1 MILC

The MIMD Lattice Computation (MILC) code has been developed to do simulations of four dimensional SU(2) lattice gauge theory on MIMD parallel machines[5]. We used PerfTrack to launch and collect data for executions of MILC on Jacquard, the Opteron cluster at NERSC. PerfTrack automatically collected details about the execution, including information about the build, the execution itself, and the performance data that was generated. The results are shown in Figure 1. The build data contain information about the compiler, compiler flags, static libraries linked into the executable, and details about the build environment such as the build node and the operating system version. The execution data contain environment variables, dynamic libraries, operating system details, information about the job scheduler and jobs in the queue, and usage and version information for file systems. The performance data were gathered from timing information output by the MILC code.

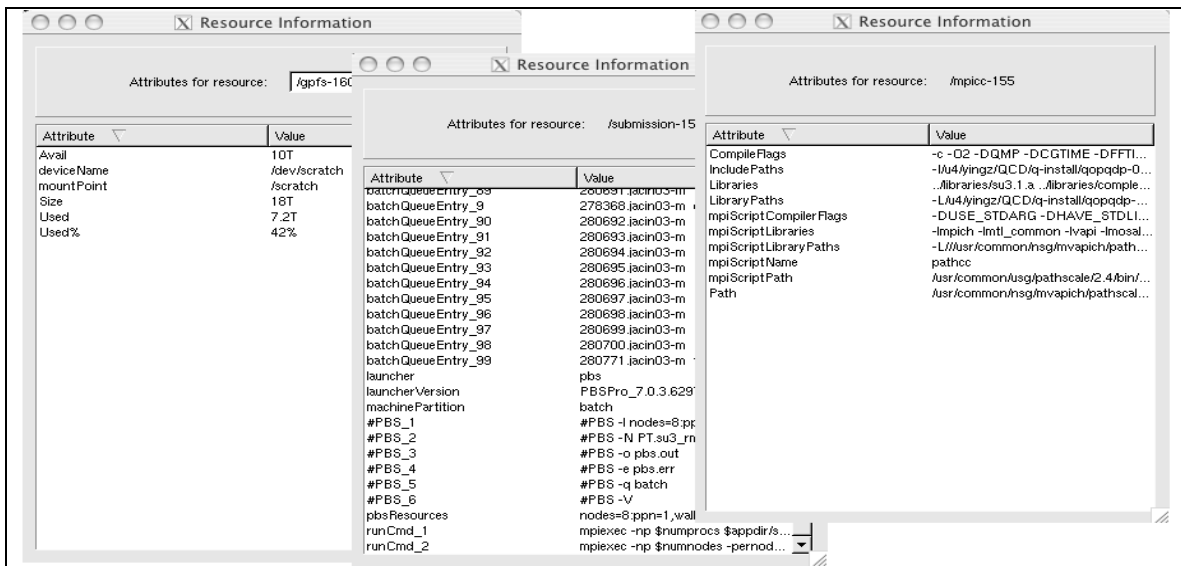


Figure 1. PerfTrack automatically collects details about executions, including attributes of file systems (left), submissions (middle), and compilers (right). Here we show select details for a MILC execution on Jacquard.

4.2 GTC

The Gyrokinetic Turbulence Code (GTC) is a Particle-in-Cell (PIC) code for gyrokinetic simulation of fusion plasmas for studying turbulent transport [6]. GTC developers have the long-term performance goal of scaling the new shaped version of GTC (GTC_s) to tens of thousands of cores so as to be able to simulate ITER-size plasmas. Timing profiles of an initial version of GTC_s at routine-level granularity on the ORNL Cray XT3/4 have been collected for processor counts ranging from 16 to 256 using TAU. A timing profile for a 64-processor run is

shown in Figure 2. Interprocess communication is shown to take a minimal amount of time. The PUSHI and CHARGEI routines implement a scatter-gather algorithm that is a known performance bottleneck.

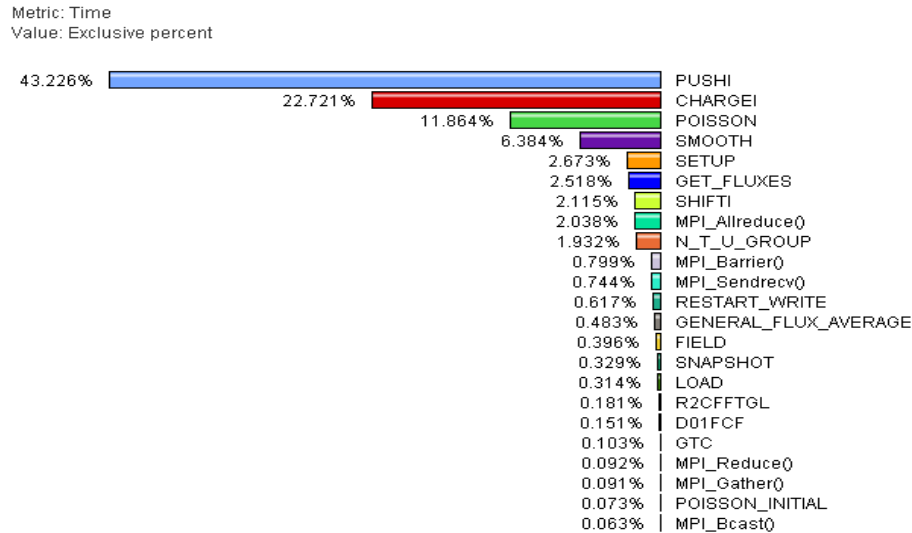


Figure 2. Mean time profile for GTC_s on 64 processors on Jaguar (ORNL Cray XT3/4)

5 Conclusions and future work

Performance database technology provides the means for storing, retrieving, and analyzing performance data from multiple executions of parallel applications. Such data can inform both manual and automated approaches to performance tuning. Interoperability between different performance database systems, especially a deeper level of semantic interoperability, will decouple the analysis from the specific data collection and measurement tools used to collect the performance data

References

- [1] Karavanic K, May J, Mohror K, Miller B, Huck K, Knapp R and Pugh B 2005 Integrating database technology with comparison-based parallel performance diagnosis: the PerfTrack performance experiment management tool *Proc SC2005* Seattle WA
- [2] Wu X, Taylor V and Paris J 2006 The web-based Prophesy automated performance modeling system *The IASTED Int'l Conf. on Web Technologies, Applications and Services (WTAS2006)* Calgary, Canada
- [3] Shende S, and Malony A 2006 The TAU parallel performance system *Int'l J High Performance Computing Applications* **20**(2) 287-331
- [4] DeRose L and Reed D 1999 SvPablo: A multi-language architecture-independent performance analysis system *Proc. Int'l Conf. On Parallel Processing* Fukushima, Japan
- [5] R C Brower *et al* 2006 National software infrastructure for lattice quantum chromodynamics *J. Phys.: Conf. Ser.* **46** 142-146
- [6] Lee W W, Ethier S, Wang W X, Tang W M and Klasky S 2006 Gyrokinetic particle simulation of fusion plasmas: path to petascale computing *J Phys.: Conf. Ser.* **46** 73-81