

CALTECH ASCI TECHNICAL REPORT 044
cit-ascii-tr044
2000

SvPablo: A Multi-Language Performance Analysis System
Luiz De Rose, Ying Zhang, and Daniel A. Reed

SvPablo: A Multi-Language Performance Analysis System

Luiz De Rose, Ying Zhang, and Daniel A. Reed *

Department of Computer Science
University of Illinois
Urbana, Illinois 61801 USA
{derose,zhang8,reed}@cs.uiuc.edu,
WWW home page: <http://www-pablo.cs.uiuc.edu/>

Abstract. SvPablo is a language independent performance analysis and visualization system that supports analysis of applications written in a variety of languages and executing on both sequential and parallel systems. In addition to capturing application data via software instrumentation, SvPablo also exploits hardware performance counters to capture the interaction of software and hardware. Both hardware and software performance data are summarized during program execution, enabling measurement of programs that execute for hours or days on hundreds of processors.

Overview

Emerging parallel systems have multilevel memory hierarchies managed by distributed cache coherence protocols or low latency message passing, all accessed by superscalar processors. To provide a language and architecture independent mechanism for performance analysis, we developed SvPablo (source view Pablo), a graphical environment for instrumenting application source code and browsing dynamic performance data.

SvPablo supports interactive instrumentation of C, Fortran 77, and Fortran 90 and automatic instrumentation of data parallel HPF programs, when compiled with the PGI HPF compiler. Interactive instrumentation provides detailed control, allowing users to specify specific instrumentation points, albeit at the possible expense of excessive perturbation and inhibition of compiler optimizations. In contrast, automatic instrumentation relies on the compiler or runtime system to insert measurement probes in compiler-synthesized code.

During execution of the instrumented code, the SvPablo library captures data and computes performance metrics on the execution dynamics of each instrumented construct on each processor. Because only statistics, rather than detailed

* This work was supported in part by the Defense Advanced Research Projects Agency under DARPA contracts DABT63-94-C0049 (SIO Initiative), F30602-96-C-0161, and DABT63-96-C-0027 by the National Science Foundation under grants NSF CDA 94-01124 and ASC 97-20202, and by the Department of Energy under contracts DOE B-341494, W-7405-ENG-48, and 1-B-333164.

event traces, are maintained, the SvPablo library can capture the execution behavior of codes that execute for hours or days on hundreds of processors.

Instrumentation

The SvPablo instrumentation library includes a standard interface for augmenting software performance data with hardware metrics. We have exploited this interface to capture hardware performance data on the MIPS R10000 [2]. Within SvPablo, a user can select the desired set of hardware counters via a configuration file. During program execution, the SvPablo data capture library queries the counters and records the data with extant application measurements. In addition to presenting the raw counter data, the SvPablo library also computes derived metrics for each source code line (e.g., MFLOPS and branch misprediction percentages).

Taken together, the application and hardware performance measurements provide a rich set of metrics for program analysis. Moreover, the SvPablo interface allows users to identify high-level bottlenecks (e.g., procedures), then explore increasingly levels of detail (e.g., identifying a specific cause of poor performance at a source code line executed on one of many processors).

Following execution, the SvPablo data capture library records its statistical analyses in a set of *summary files*, one for each executing process. A post-mortem utility program then merges the summary files, computing new global statistics and the resulting metrics are correlated with application source code, creating a *performance file* that is represented via the Pablo Self-Describing Data Format (SDDF) [1]. This performance file is the specification used by the SvPablo browser to display application source code and correlated performance metrics.

Analysis

Developing an interface that separates performance data presentation from language and architecture idiosyncrasies requires a flexible specification mechanism for both instrumentation points and performance metrics. Only with this separation can one readily add new metrics and support new languages, compilers, and architectures without requiring extensive modifications to the user interface.

To isolate language differences from the user interface, the performance metrics associated with each procedure and source line are organized as a hierarchy defined by a set of SDDF records. This *meta-meta-format* hierarchy contains three groups of SDDF record descriptors: *mapping*, *configuration* and *statistic*. Mapping records define the set of statistics associated with each instrumentable construct. In turn, configuration records indicate the statistic record names and allow SvPablo to extract the base names of all performance metrics before reading the statistics records, which define the actual performance metrics.

Finally, one of the design goals for SvPablo was creation of an intuitive, cross-architecture, language independent performance analysis interface. Realizing such a design would allow users and performance analysts to learn a single set of software navigation skills and then apply those skills to application codes

written in a variety of languages and executing on a diverse set of sequential and parallel architectures.

Hence, the SvPablo implementation relies on a single interface for performance instrumentation and visualization. If the code was interactively instrumented, the user can refine the performance analysis by re-instrumenting the code while visualizing performance data from earlier executions. Regardless of the instrumentation mode, one can access and load performance data from multiple prior executions, including different numbers of processors and hardware platforms. This allows one to compare executions to understand hardware and software interactions.

Application

As an example, Figure 1 shows the SvPablo interface, together with code and performance data from an HPF program. The SvPablo interface supports a hierarchy of performance displays, ranging from color-coded routine profiles to detailed data on the behavior of a source code line on a single processor.

In the figure, the leftmost scrollbox shows the set of files comprising the HPF code, with all previously measured executions of this code shown in the scrollbox to the right. Here, the user has loaded a performance data context (i.e., a measured execution) for a 32 processors Silicon Graphics Origin 2000. After selecting a performance context, the list of procedures in the application code, together with two color coded metrics, is shown below the performance contexts scrollbox in the area labeled *Routines in Performance Data*.

The two colored columns summarize, over all processes, the mean number of calls and mean cumulative time for the routines. Clicking on a routine name loads the associated source code in the bottom pane of Figure 1, together with color-coded metrics beside each source code line. In addition, pop-up dialogs showing other statistics and detailed information about a particular routine or a particular source code line, including individual processor metrics, can be obtained by clicking the mouse on the routine name or the source code line.

Working with a group of large-scale applications, we observed that SvPablo enabled us to rapidly identify and correct performance bottlenecks. However, the key feature of SvPablo is language and architecture transparency, achieved by representing performance data via a meta-meta-format presentation of different events from different languages using the same graphical interface.

References

1. Reed, D. A., Aydt, R. A., Noe, R. J., Roth, P. C., Shields, K. A., Schwartz, B., Tavera, L. F.: Scalable Performance Analysis: The Pablo Performance Analysis Environment. In *Proceedings of the Scalable Parallel Libraries Conference* (1993), A. Skjellum, Ed., IEEE Computer Society.
2. Zagha, M., Larson, B., Turner, S., Itzkowitz, M.: Performance Analysis Using the MIPS R10000 Performance Counters. In *Proceedings of Supercomputing'96* (November 1996).

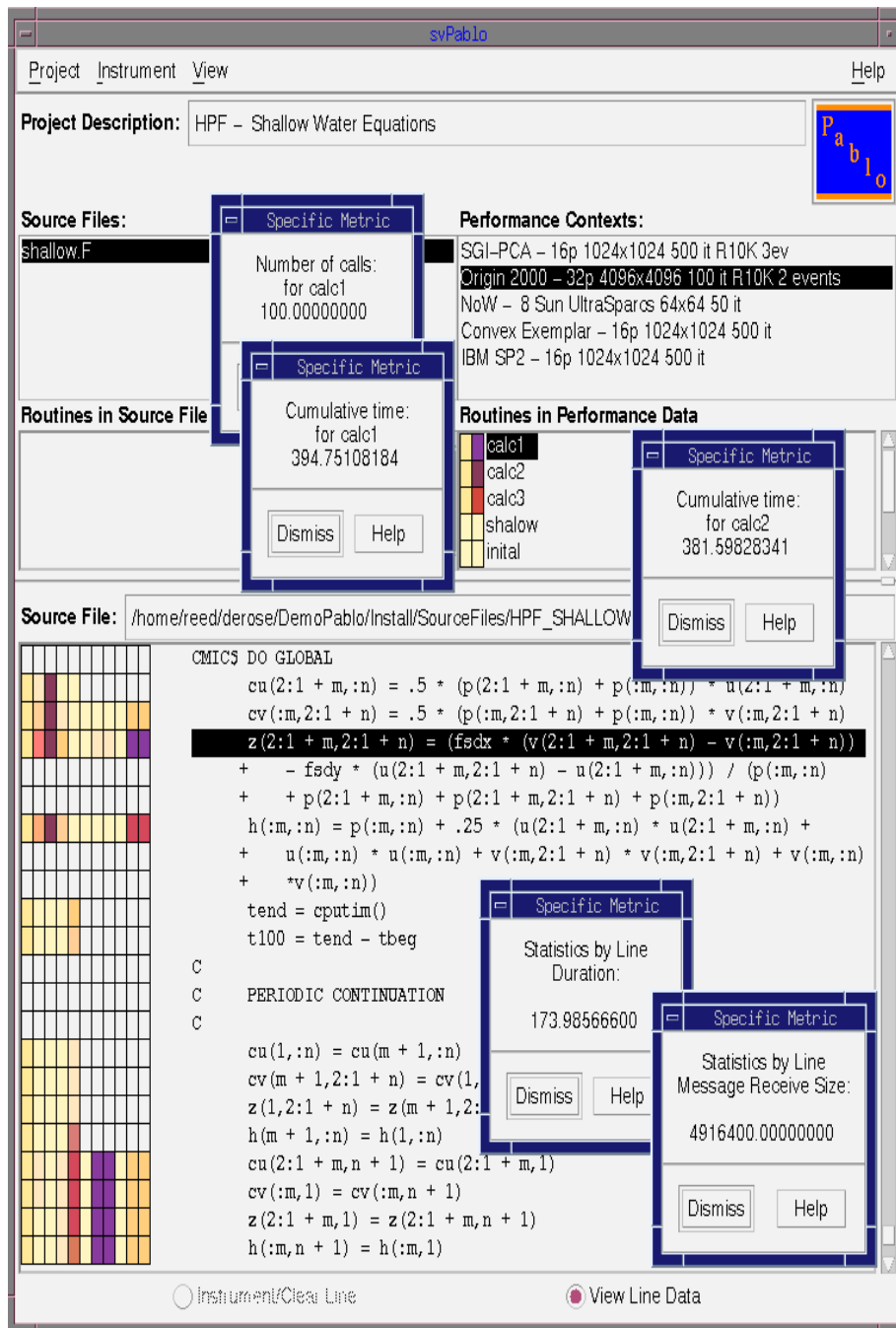


Fig. 1. Shallow Water Code (HPF Performance)