On Providing Useful Information for Analyzing and Tuning Applications

John Mellor-Crummey[1], Robert Fowler[1], and David Whalley[2]
[1] Department of Computer Science, Rice University, 6100 Main Street, Houston, TX 77005
[2]Department of Computer Science, Florida State University, Tallahassee, FL 32306
johnmc@rice.edu, rjf@rice.edu, whalley@cs.fsu.edu

Application performance tuning is a complex process that requires correlating many types of information with source code to locate and analyze performance problems bottlenecks. Existing performance tools don't adequately support this process in one or more dimensions. We describe two performance tools, *MHSim* and *HPCView*, that we built to support our own work on data layout and optimizing compilers. Both tools report their results in scope-hierarchy views of the corresponding source code and produce their output as HTML databases that can be analyzed portably and collaboratively using a commodity browser.

MHSim is an integrated simulator and instrumentation tool designed to investigate problems with poor memory hierarchy utilization in Fortran programs. *MHSim* quantifies cache conflicts, temporal reuse, and spatial reuse, and it correlates simulation results to individual references and loops in the target program.

When browsing *MHSim* output, as shown in Figure 1, the upper right pane displays source code annotated with hyperlinks. Clicking on a # hyperlink that precedes an array reference scrolls each of the panes below to display the simulation results associated with that reference. To the left of each loop header (scope) are two hyperlinks. Clicking the 'S' hyperlink displays loop summary information in the panes below rather than the reference-level information shown in the figure. Clicking the 'A' hyperlink displays loop-level summary information for each array referenced in the loop. The simulation results for each level of the memory hierarchy are displayed in panes below the source pane, one line per array reference in the source code. The bottom pane in the window shows evictor information. For each reference in the source program, the evictor pane presents a sorted list of source-code references that cause interference (cache evictions, including self-interference) with this reference. Clicking on the hyperlink associated with an evicting reference will auto-navigate all of the panes to display the source code and simulation output for that reference. The evictor information is an effective, intuitive mechanism for identifying and quantifying patterns of cache interference.

HPCView produces a navigable HTML document that combines performance metrics from diverse sources, synthsizes new metrics, and correlates them with program source code. *HPCView* inputs are "profiles" in a standardized format. Simple scripts convert vendor- or tool- specific data into the *HPCView* format. To date, the principal source of input data have been profiles generated by sampling the program counter at events such as timers or the overflow of hardware performance counters tracking interesting events, *e.g.* cache misses. Other "profile-like" data sources include simulators and static analysis tools.

Figure 2 shows an *HPCView* screenshot. From left to right, the data columns are: **cycles**, measured with a hardware counter; **ideal cycles**, computed by *pixie* using a combination of static analysis and loop counts; *stalls*, computed by *HPCView* by subtracting ideal cycles from cycles; and **FLOPS**, from a hardware counter. Each line in the source pane with associated performance information is marked with an 'L' hyperlink that navigates and highlights the other panes. Below the source pane is a flat, line-oriented performance data table. Each line is labelled a hyperlink containing file name and line number. The table is sorted in descending order by the performance metric column is selected by clicking on the 'sort' link in the column header. Sorting affects both the flat and hierarchical displays. The lower three panes of window are a hierarchical display of data aggregated by program scopes: program, source file, procedure, loop, and source line. In general, data sources provide data for only individual source lines. We combine that data with scope information obtained through static analysis to aggregate the data at the higher levels. The three panes show the data, from top to bottom, for a parent scope, the currently selected scope, and its immediate children. Links navigate up and down the hierarchy.

MHSim and *HPCView* have proven themselves to be extremely useful for analysis and tuning for two general reasons. First, they explicitly present information that is useful for tuning. Of special utility are the evictor information of *MHSim* and the combination of data, including computed metrics, from multiple sources in *HPCView*. Second, by combining multiple views of performance data with a source browser under a familiar, browsable interface, these tools dramatically reduce the effort needed to do performance analysis. This reduction of effort is of particular importance in repeated cycles of measurement, analysis, and tuning.

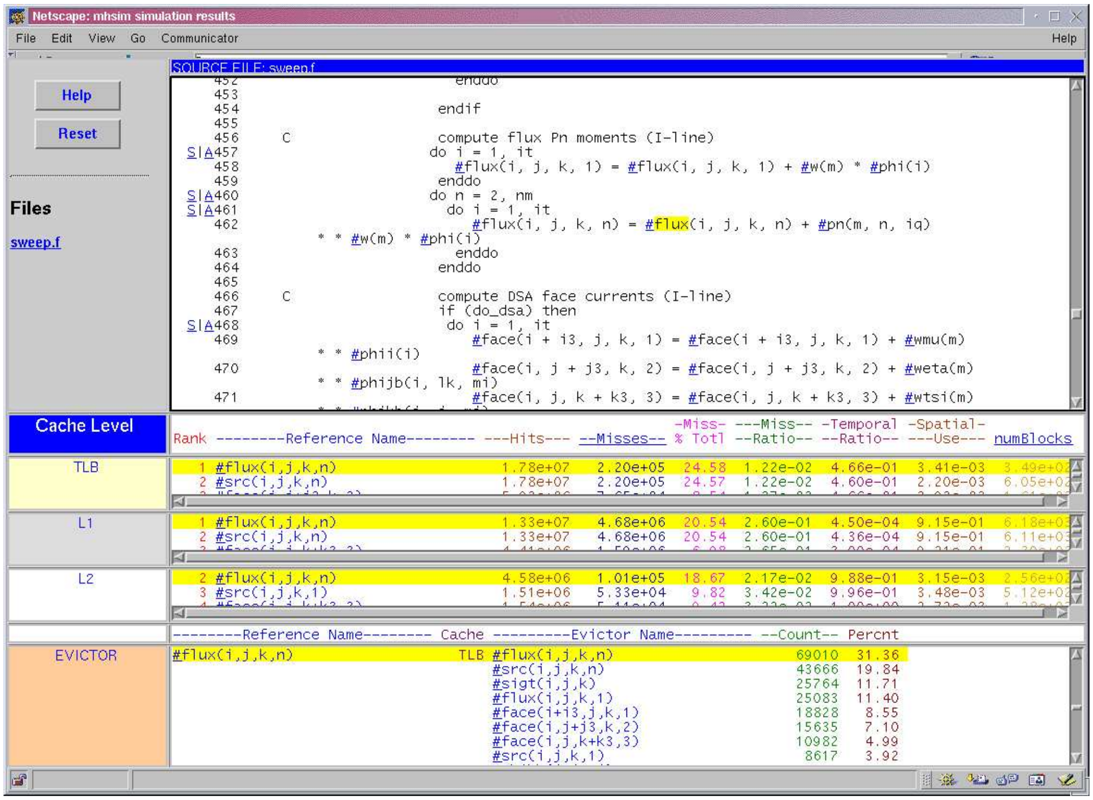

Figure 1: The *MHSim* user interface.

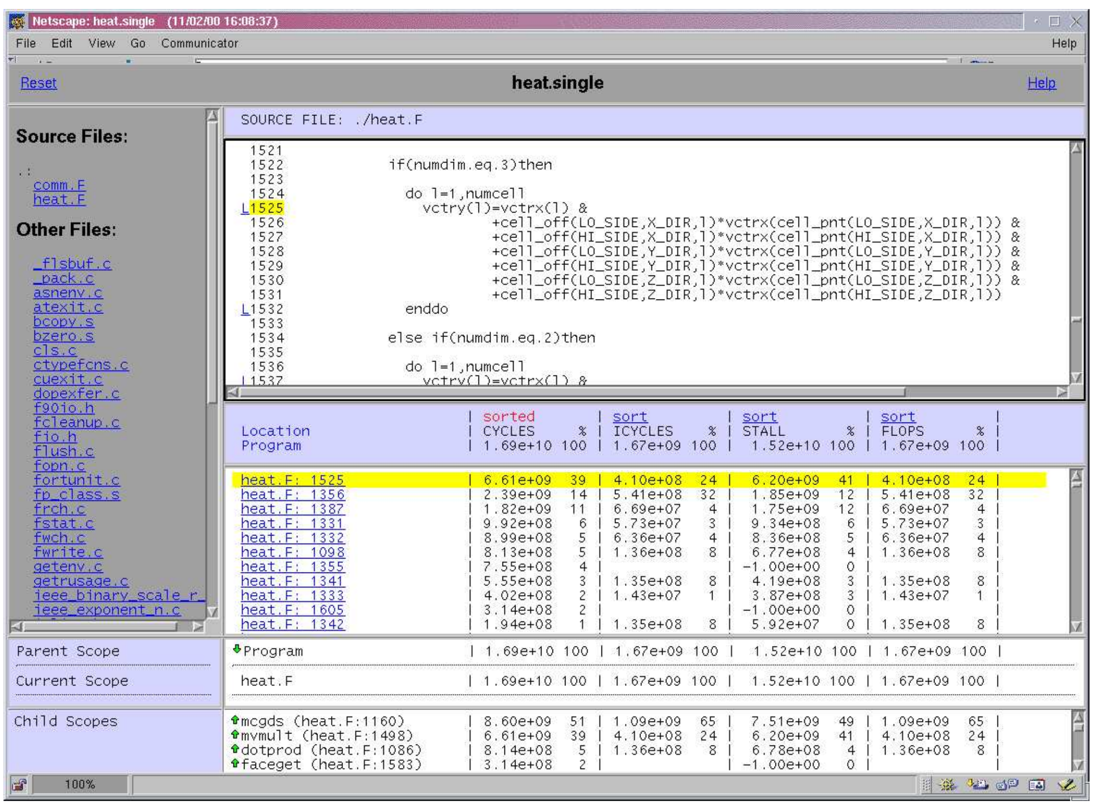

Figure 2: HPCView displaying both measured and computed metrics.