Initial Kernel Timing Using a Simple PIM Performance Model

Daniel S. Katz, Gary L. Block, Paul L. Springer, and ***Thomas Sterling***
Jet Propulsion Laboratory
Phone: 818-354-7359
Email Addresses: {Daniel.S.Katz, Gary.L.Block, Paul.L.Springer}@jpl.nasa.gov
Email Address: *tron@cacr.caltech.edu*

Jay B. Brockman
Notre Dame University
Email Address: *jbb@cse.nd.edu*,

David Callahan
Cray, Inc.
Email Address: *david@cray.com*

1

This presentation will describe some initial results of paper-and-pencil studies of 4 or 5 application kernels applied to a processor-in-memory (PIM) system roughly similar to the Cascade Lightweight Processor (LWP). The application kernels are:

- Linked list traversal
- Sun of leaf nodes on a tree
- Bitonic sort
- Vector sum
- Gaussian elimination

The intent of this work is to guide and validate work on the Cascade project in the areas of compilers, simulators, and languages.

We will first discuss the generic PIM structure. Then, we will explain the concepts needed to program a parallel PIM system (locality, threads, parcels). Next, we will present a simple PIM performance model that will be used in the remainder of the presentation.

For each kernel, we will then present a set of codes, including codes for a single PIM node, and codes for multiple PIM nodes that move data to threads and move threads to data. These codes are written at a fairly low level, between assembly and C, but much closer to C than to assembly. For each code, we will present some hand-drafted timing forecasts, based on the simple PIM performance model.

Finally, we will conclude by discussing what we have learned from this work, including what programming styles seem to work best, from the point-of-view of both expressiveness and performance.

---

[1] This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under its Contract No. NBCH3039003.