Initial Kernel Timing Using a Simple PIM Performance Model

Daniel S. Katz\textsuperscript{1*}, Gary L. Block\textsuperscript{1}, Jay B. Brockman\textsuperscript{2}, David Callahan\textsuperscript{3}, Paul L. Springer\textsuperscript{1}, Thomas Sterling\textsuperscript{1,4}

\textsuperscript{1}Jet Propulsion Laboratory, California Institute of Technology, USA
\textsuperscript{2}University of Notre Dame, USA
\textsuperscript{3}Cray Inc., USA
\textsuperscript{4}California Institute of Technology, USA

*Technical Group Supervisor
Parallel Applications Technologies Group
http://pat.jpl.nasa.gov/
Daniel.S.Katz@jpl.nasa.gov

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under its Contract No. NBCH3039003.
Purpose of this Poster

- Discuss initial results of paper-and-pencil studies of 4 application kernels applied to a processor-in-memory (PIM) system roughly similar to the Cascade Lightweight Processor (LWP)

- Application kernels:
  - Linked list traversal
  - Vector sum
  - Bitonic sort

- Intent of work is to guide and validate work on Cascade in the areas of compilers, simulators, and languages
Poster Topics

• Generic PIM structure
• Concepts needed to program a parallel PIM system
  • Locality
  • Threads
  • Parcels
• Simple PIM performance model
• For each kernel:
  • Code(s) for a single PIM node
  • Code(s) for multiple PIM nodes that move data to threads
  • Code(s) for multiple PIM nodes that move threads to data
  • Hand-drafted timing forecasts, based on the simple PIM performance model
• Lessons learned
  • What programming styles seem to work best
    • Looking at both expressiveness and performance