Designing and Auto-Tuning Parallel 3-D FFT for Computation-Communication Overlap

Sukhyun Song, Jeffrey K. Hollingsworth
Department of Computer Science
University of Maryland, College Park
Email: {shsong, hollings}@cs.umd.edu

Abstract
- Design & auto-tune a new parallel 3-D FFT code
- Use non-blocking MPI all-to-all operations
- Maximize computation-communication overlap
- Achieve up to 1.76x speedup over FFTW

Motivation
- The three-dimensional Fast Fourier Transform (3-D FFT) is widely used in many fields of science and engineering.
  - Astronomical N-body simulations
  - Blood flow simulations
  - A parallel 3-D FFT operation is both compute- and communication-intensive.
- It requires all-to-all communication.
- We can increase the 3-D FFT performance by hiding communication behind computation.
- The non-blocking MPI_Ialltoall can be used.

Unique Characteristics
- Optimize computation-communication overlap
  - Overlap all possible computation with communication
  - Design a portable code
    - Fully asynchronous communication without special hardware support or separate threads
  - Optimize local computation
    - Improve cache performance through loop tiling
    - Parameterize and auto-tune our 3-D FFT code
      - Cope with the complex trade-off regarding our optimization techniques

Auto-Tuning with Active Harmony
- Tune parameters in our 3-D FFT code
  - Communication tile size, degree of communication parallelism, loop tiling factors, kidx=1_test call frequencies
  - Active Harmony (AH): A general software framework for auto-tuning
    - We use the Nelder-Mead optimization method to search for the best parameter configuration.
  - Repeat the following steps until finding the best parameter set.
    1) The AH server provides a parameter configuration to be tested.
    2) The AH client executes 3-D FFT with the received configuration.
    3) The client reports the performance back to the server.

Preliminary Results
- Comparison Models
  - FFTW: The MPI-enabled FFTW library
  - NEW: Our new auto-tuned parallel 3-D FFT code
  - TH: Hoeffer et al.’s code auto-tuned by us for fair comparison
- Parallel 3-D FFT Performance
  - Experiments on Hopper at NERSC
    - The speedup of NEW over FFTW: 1.48x – 1.76x

Design of Parallel 3-D FFT
- 3-D FFT is a composition of three sets of 1-D FFTs.
- Assign a data block divided along the x-dim to each process.
- Then each process executes the following steps:
  1) FFTz: 1-D FFTs along the z dimension
  2) Transpose: Change the memory layout for the next step
  3) FFTy: 1-D FFTs along the y dimension
  4) Pack: Data are packed into a communication buffer.
  5) A2A: Perform a non-blocking MPI all-to-all operation
  6) Unpack: The received data are unpacked.
  7) FFTx: 1-D FFTs along the x dimension

Overlap Strategy
- Divide a data block into small communication tiles
- Overlap computation on a tile with communication for other tiles
- Fully asynchronous All-to-All Communication
  - Periodically call kdx=1_test during computation.

Loop Tiling
- Tile the loop inside each communication tile
- Increase cache hits during computation