Performance Analysis and Optimization of a Hybrid Distributed Reverse Time Migration Application

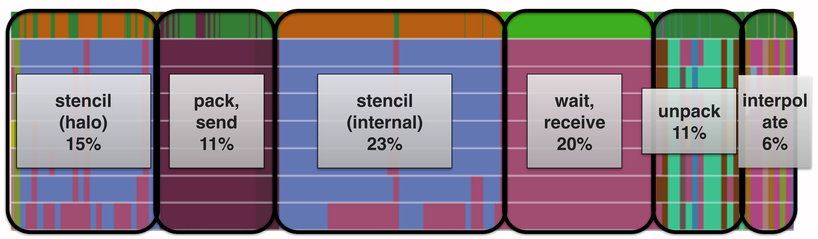
Sri Raj Paul, John Mellor-Crummey (Advisor)
Rice University, Houston, TX, USA

Mauricio Araya-Polo (Advisor), Detlef Hohl (Advisor)
Shell International Exploration & Production Inc., Houston, TX, USA

Distributed RTM and Optimization Challenges

- RTM is a computationally intensive method that employ a high order stencil to create images of subsurface structures
- Distributed RTM employs MPI+OpenMP to perform RTM calculations on a cluster
- Analysis of MPI+OpenMP applications is challenging
  - — Performance analysis tools can help
- Tuning MPI+OpenMP applications requires understanding at multiple levels
  - — Work partitioning and interprocess communication
  - — Threading within each process
  - — Functional unit and cache utilization within a core
- Rice University's HPCToolkit supports analysis at each of these levels

HPCToolkit Visualization of a Forward Timestep



A complete execution includes other activities such as reading data from input files and hence reported percentages do not add to 100%

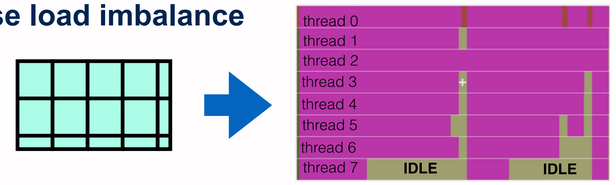
- Calculate stencil for halo regions
- Pack halo data and send to neighbors
- Calculate stencil for internal regions
- Wait for data to arrive from neighbors
- Unpack data received from neighbors
- Interpolate as necessary

References

1.L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent. HPCToolkit: Tools for performance analysis of optimized parallel programs. Concurr. Comput. : Pract. Exper., 22(6):685–701, Apr. 2010

Remove Thread Level Load Imbalance

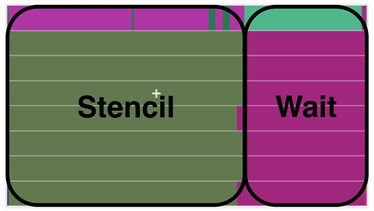
- Smaller tiles towards the high end of each dimension cause load imbalance



- HPCToolkit's sample traces show thread idleness
- Static scheduling of tiles with unequal size causes load imbalance
- Dynamic scheduling reduces load imbalance
  - — Improves overall performance by roughly 5%

Computation-Communicaton Overlap

- HPCToolkit shows waiting after the stencil computation
  - — waiting for communication to finish



- Communication is initiated before and completed after stencil computation but delays indicate little overlap
- Reserving one thread to progress communication improves performance by about 10%

Improve Data Reuse in Cache

- Measurements with PAPI using hardware performance counters show a high L2 cache miss rate
- DRTM's 3D stencil involves more points in the X-Y plane than the X-Z plane
- Interchanging loops so that X-Y planes are innermost improves cache reuse, improving performance by roughly 5%

	Original	Interchange	Reduction
L1 DCM	3.37E+11 (12.5%)	3.19E+11 (11.8%)	5%
L2 DCM	1.17E+11 (35%)	6.67E+10 (21%)	43%

Hardware performance counters showing cache misses for 4 processes (with cache miss rate shown in parentheses)

Experimental Results

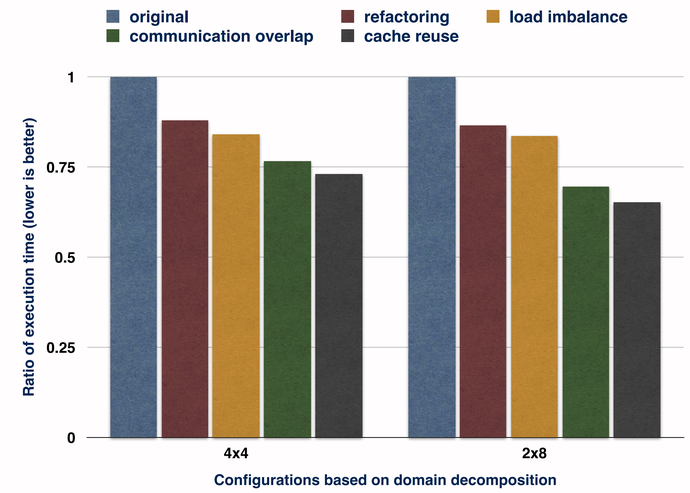


Figure 1: Improvement with respect to each optimization for two configurations: 4x4 and 2x8 domain decompositions of the X-Y plane

Refactoring an abstraction layer to remove unnecessary data copies used to support different programming models avoided memory copies that were unnecessary for OpenMP.

Conclusions and Future Directions

- A complex interplay of factors affects the performance of MPI+OpenMP programs
- Tuning such applications requires understanding performance at multiple levels
- HPCToolkit enables analysis of parallelism across nodes, within a node, and performance of individual cores
- Insights from HPCToolkit and hardware performance counters helped us improve DRTM's performance by roughly 30%
- Ongoing work
  - — Improving register reuse and vectorization of higher order stencils
  - — Improving load balance by using a work-aware nonuniform domain decomposition

This research was partially supported by Shell International Exploration & Production Inc. under research agreement PT46021.