Optimizing Fine-grained Communication in a Biomolecular Simulation Application on Cray XK6

Yanhua Sun[1] Gengbin Zheng[1] Chao Mei[1] Eric J. Bohm[1]
James C. Phillips[1] Terry Jones[2] Laxmikant(Sanjay) V. Kále[1]

[1]University of Illinois at Urbana-Champaign
[2]Oak Ridge National Lab

sun51@illinois.edu

November 27, 2012

Motivation - Molecular Dynamics Simulation

- Critical in understanding the functioning of biological machinery

Motivation - Molecular Dynamics Simulation

- Critical in understanding the functioning of biological machinery
- Challenging - femto-second timestep to maintain accuracy
- Hundreds of nanoseconds (even microseconds) are needed to observe interesting phenomena

Motivation - Molecular Dynamics Simulation

- Critical in understanding the functioning of biological machinery
- Challenging - femto-second timestep to maintain accuracy
- Hundreds of nanoseconds (even microseconds) are needed to observe interesting phenomena
- 10^8 steps
- *millisecond* timestep simulation

Motivation - Molecular Dynamics Simulation

- Critical in understanding the functioning of biological machinery
- Challenging - femto-second timestep to maintain accuracy
- Hundreds of nanoseconds (even microseconds) are needed to observe interesting phenomena
- 10^8 steps
- *millisecond* timestep simulation
- a single time step of 1 million atom simulation : $20 seconds$
- scale to hundreds of thousands cores
- fine-grained decomposition

Outline

Analyze and optimize the fine-grained molecular dynamics simulation on Cray XK6 supercomputer

Outline

Analyze and optimize the fine-grained molecular dynamics simulation on Cray XK6 supercomputer

Outline

- Background of Cray XK6, CHARM++ and NAMD

Outline

Analyze and optimize the fine-grained molecular dynamics simulation on Cray XK6 supercomputer

Outline

- Background of Cray XK6, CHARM++ and NAMD
- Detect the performance bottleneck

Outline

Analyze and optimize the fine-grained molecular dynamics simulation on Cray XK6 supercomputer

Outline

- Background of Cray XK6, CHARM++ and NAMD
- Detect the performance bottleneck
- Optimization techniques

Outline

Analyze and optimize the fine-grained molecular dynamics simulation on Cray XK6 supercomputer

Outline

- Background of Cray XK6, CHARM++ and NAMD
- Detect the performance bottleneck
- Optimization techniques
- Performance results

Cray XK6 - Jaguar at ORNL

Processors

- 16-core Interlagos processor, GPGPU
- Kepler GPGPU (results are on Fermi)
- CPU set - Jaguar XK6; GPU set - TitanDev

Cray XK6 - Jaguar at ORNL

Processors

- 16-core Interlagos processor, GPGPU
- Kepler GPGPU (results are on Fermi)
- CPU set - Jaguar XK6; GPU set - TitanDev

Network

- 3D torus Gemini Interconnect
- Hardware support for RDMA
- user Generic Network Interface (uGNI)

CHARM++ Runtime System

CHARM++ is a parallel programming model that implements message-driven objects.

CHARM++ Runtime System

CHARM++ is a parallel programming model that implements message-driven objects.

- Improve both the performance and productivity
- Adaptive runtime system - mapping, balancing , etc.
- Multithreading mode (SMP) for multi-core computers

SMP CHARM++ implementation on uGNI

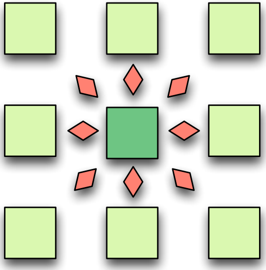
- Worker threads put messages into Comm thread queues
- Medium/Large messages($> 1KB$) - RDMA
- Small messages - SMSG
- Polling network messages

NAMD

A highly scalable molecular dynamics application developed in the mid-1990s, based on CHARM++.

NAMD

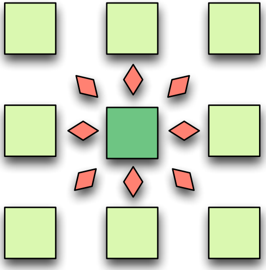
A highly scalable molecular dynamics application developed in the mid-1990s, based on CHARM++.



- Simulation box is spatially divided into "patches"
- Force calculation between two patches is assigned to compute objects

NAMD

A highly scalable molecular dynamics application developed in the mid-1990s, based on CHARM++.



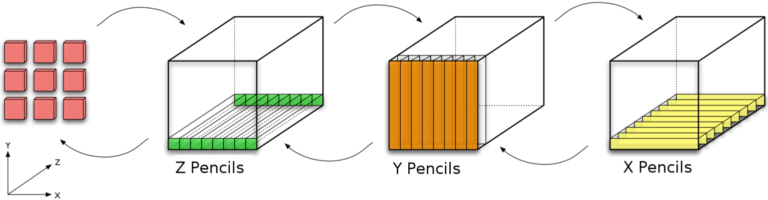
- Simulation box is spatially divided into "patches"
- Force calculation between two patches is assigned to compute objects
- Complexity is $O(NlogN)$ by using short-range and long-range calculation
- GPU case : short-range work on GPUs, long-range work on CPU

PME

Long-range calculation is implemented via particle-mesh Ewald method (PME)

PME

Long-range calculation is implemented via particle-mesh Ewald method (PME)



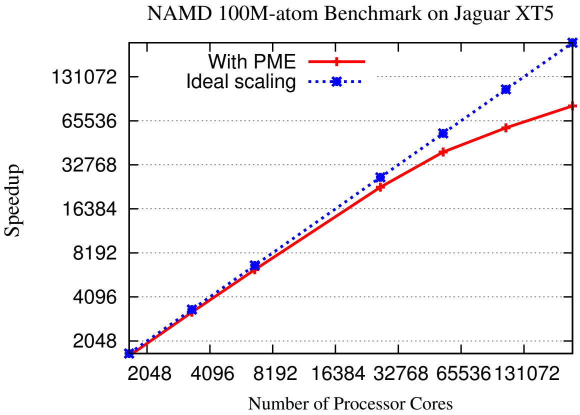
Pencil PME communication pattern

Molecular Systems

Molecule	Atoms	Cutoff($\dot{A}$)	Simulation Box
DHFR	23558	9	62x62x62
Apoa1	92224	12	108x108x77
1M STMV	1066628	12	216x216x216
100M STMV	106662800	12	1084x1084x867

Table: Parameters for four molecular systems

Earlier Work



NAMD scaled up to 224K cores on Cray XT5 for a 100-million atom simulation.
Speedup starts to falter beyond 64K cores.

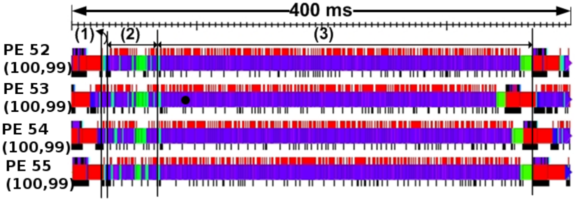
Performance Analysis

Trace-based Performance Analysis Tool – Projections

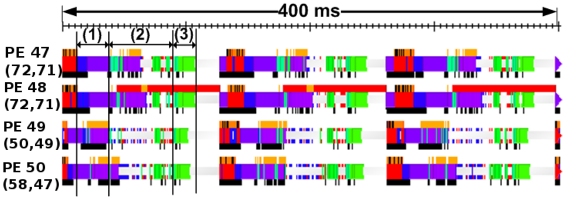
- Automatic runtime instrumentation module
- Java-based GUI program to visualize and analyze the performance data

Timelines for CPU and GPU runs

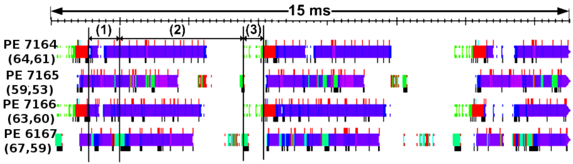
Purple: short-range work; Green: long-range work;
Red: integration; White: idle



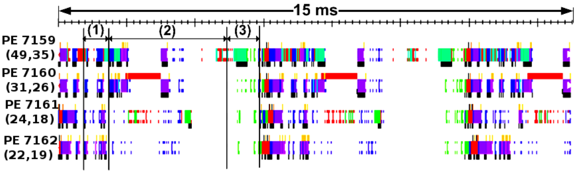
4 nodes with CPU only



4 nodes with GPU

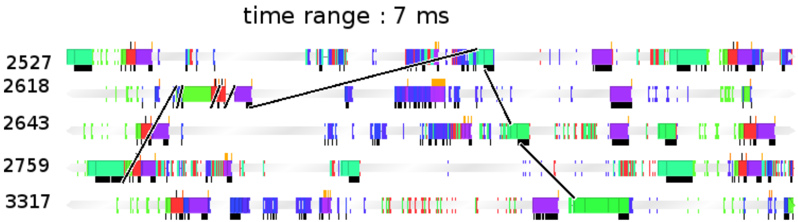


512 nodes with CPU only



512 nodes with GPUs

Trace back Multiple Messages for Critical Path



Speedup Communication on Critical Path - Priority Messages

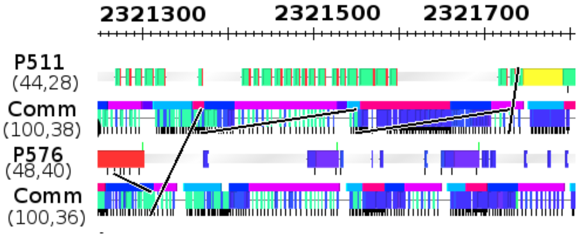
Short-range calculation, PME work driven by different messages

Speedup Communication on Critical Path - Priority Messages

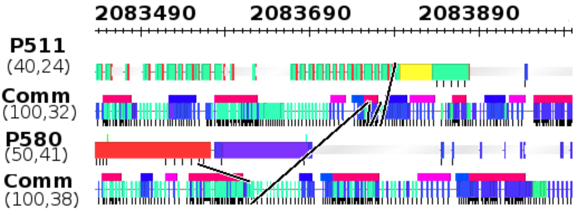
Short-range calculation, PME work driven by different messages

- Sender: Out-of-Band sending
- Receiver: Priority execution
- Increase responsiveness

The message tracing of patch-to-PME in Projections timeline for DHFR running on 1024 cores



Without optimization



With optimization

Persistent Messages

- Persistent channels for FFT are setup at the beginning
- No need to allocate memory
- Direct one-sided put
- 10% overall performance improvement

PME Decomposition

Theorem

$$T = T_{comm} + T_{comp} = \frac{D}{4 * B * \alpha} + \frac{NlogN}{P} \tag{1}$$

PME Decomposition

Theorem

$$T = T_{comm} + T_{comp} = \frac{D}{4 * B * \alpha} + \frac{NlogN}{P} \quad (1)$$

- Reduce the number of messages to utilize network more efficiently (1 pencil per physical node)
- More parallelism to utilize CPU resources (1 pencil per core)
- Tradeoff (1 pencil per CHARM++ process)

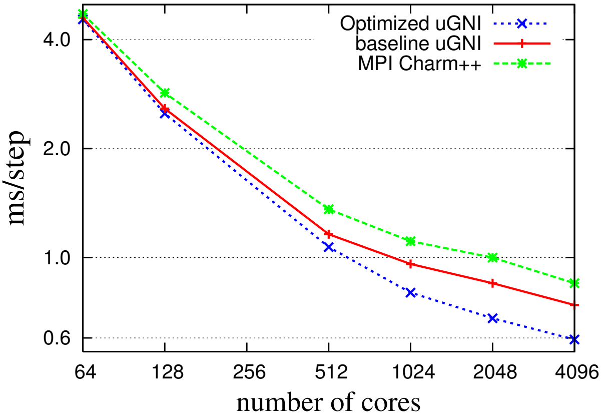
PME Decomposition

Theorem

$$T = T_{comm} + T_{comp} = \frac{D}{4 * B * \alpha} + \frac{NlogN}{P} \tag{1}$$

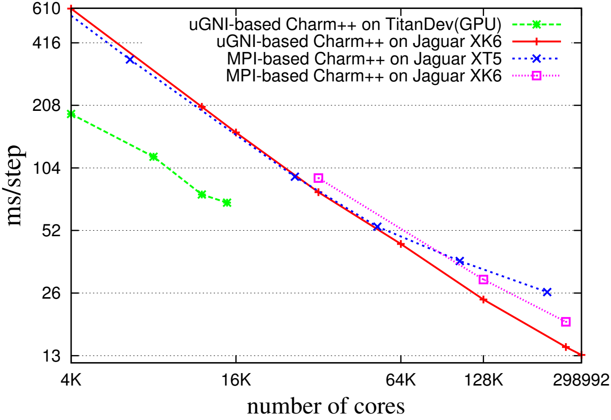
- Reduce the number of messages to utilize network more efficiently (1 pencil per physical node)
- More parallelism to utilize CPU resources (1 pencil per core)
- Tradeoff (1 pencil per CHARM++ process)
- CkLoop library to utilize all cores

Performance Results - DHFR



NAMD DHFR CPU performance on TitanDev

Performance Results - XK6 V.S. XT5 (100 million atoms)



Conclusion and Future Work

Conclusion

- Techniques to analyze and optimize NAMD on both application and runtime system
- Timestep of 100M STMV is improved from 26$ms/step$ on Jaguar XT5 to 13$ms/step$ XK6

Conclusion and Future Work

Conclusion

- Techniques to analyze and optimize NAMD on both application and runtime system
- Timestep of 100M STMV is improved from $26ms/step$ on Jaguar XT5 to $13ms/step$ XK6

Future Work

- Topology-aware PME distribution and communication
- Multi-level summation to replace PME

PPL@SC12

- Tutorial Charm++, 8:30AM-12:00PM on Sunday November 11
- HPC Challenge BoF, 12:15PM-1:15PM on Tuesday November 13, in 255-A
- Sidney Fernbach Award Talk, 11:30AM-12:00PM on Wednesday November 14th, in 155-E
- Dissertation showcase "Saving Energy and Power", 11:15AM - 11:30AM on Wednesday November 14, in 155-F
- Paper talk "Optimizing fine-grained communication in a biomolecular dynamics simulation application on Cray XK6" on Wednesday Nov 14, in 355-EF
- Charm++ BoF, 12:15PM-1:15PM on Thursday November 15, in 255-A
- http://charm.cs.illinois.edu/