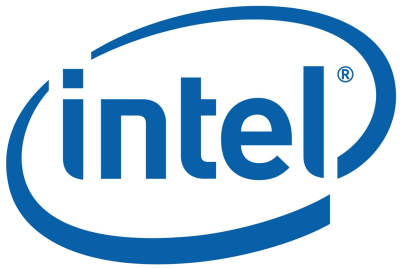
NWChem and Global Arrays Applications using MPI-3 RMA

Jeff Hammond

Extreme Scalability Group & Parallel Computing Lab
Intel Corporation (Portland, OR)

27 August 2014



Abstract (for posterity)

NWChem is a well-known quantum chemistry package designed for massively parallel supercomputers. The basis for NWChem's parallelism is the Global Arrays programming model, which supports distributed arrays, dense linear algebra, flexible one-sided communication and dynamic load-balancing. The low-level communication runtime of Global Arrays is called ARMCI. Dinan and coworkers first mapped ARMCI to MPI-2 remote memory access (RMA), which helped drive the development of the MPI-3 standard. We will describe our implementation of ARMCI using MPI-3 RMA and performance results showing the scalability of NWChem on multiple platforms. In particular, the MVAPICH2 implementation of MPI-3 delivers excellent performance and scalability on InfiniBand systems.

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Extreme Scalability Group Disclaimer

- I work in Intel Labs and therefore don't know anything about Intel products.
- I work for Intel, but I am not an official spokesman for Intel. Hence anything I say are my words, not Intel's. Furthermore, I do not speak for my collaborators, whether they be inside or outside Intel.
- You may or may not be able to reproduce any performance numbers I report.
- Performance numbers for non-Intel platforms were obtained by non-Intel people.
- Hanlon's Razor.

Collaborators

Jim Dinan (Intel) wrote the original version of ARMCI-MPI targeting MPI-2 RMA while at Argonne.

Pavan Balaji (Argonne) is the MPICH team lead and has been overseeing ARMCI-MPI development throughout.

Overview of Computational Chemistry

Atomistic simulation in chemistry

1. *classical* molecular dynamics (MD) with empirical potentials
2. *quantum* molecular dynamics based upon **density**-function theory (DFT)
3. *quantum* chemistry with **wavefunctions** e.g. perturbation theory (PT), coupled-cluster (CC) or quantum monte carlo (QMC).

Classical molecular dynamics

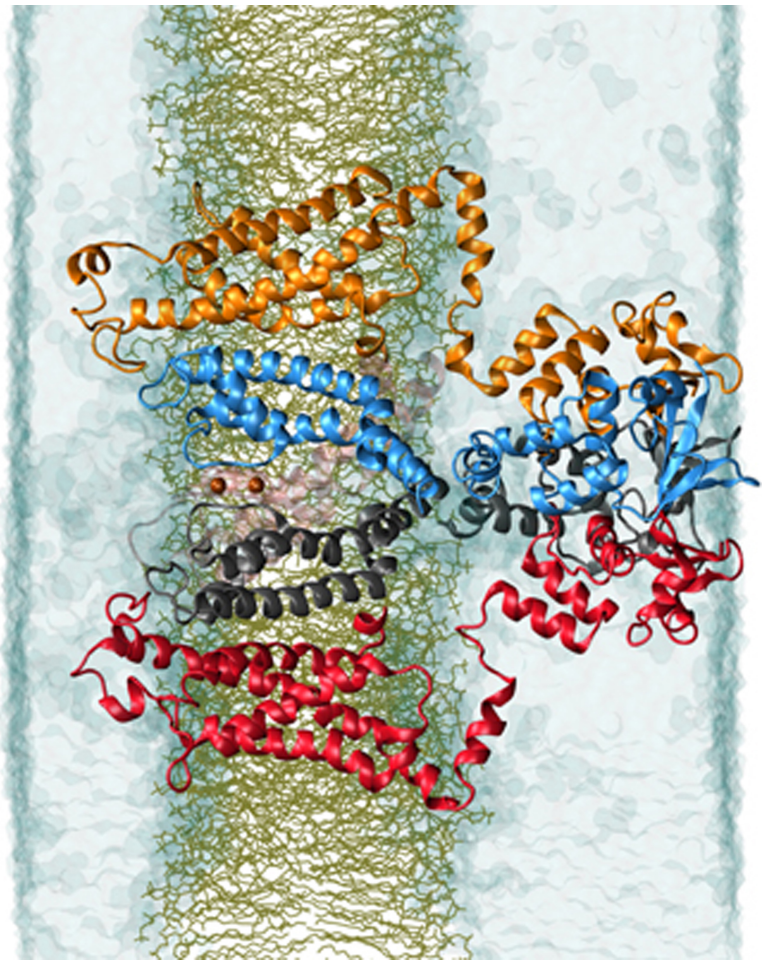


Image courtesy of Benoît Roux via ALCF.

- Solves Newton's equations of motion with empirical terms and classical electrostatics.
- Math: *N*-body
- Programming model needs: Small data, load-imbalanced, latency-sensitive.
- Software: NAMD (Charm++), LAMMPS (MPI+X), Gromacs (MPI+X).

Car-Parrinello molecular dynamics

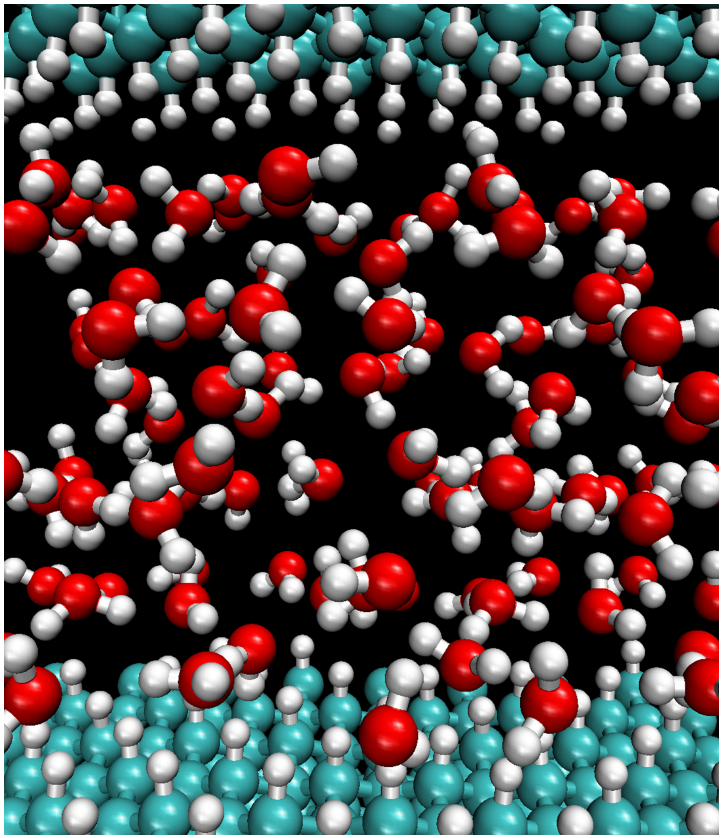


Image courtesy of Giulia Galli via ALCF.

- Forces obtained from solving an approximate single-particle Schrödinger equation.
- Math: 3D FFT, dense linear algebra.
- Programming model needs: Medium data, load-balanced, bandwidth-intensive.

Quantum chemistry

Method Property	Hartree-Fock	Coupled Cluster	Quantum Monte Carlo
Physics	Mean-field	Many-body	Diffusion Eqn.
Math	Eigensolver, Sparse, Matrix-free	Tensor contractions	Interpolation, Monte Carlo
Data	Modest	Very large	Large
Compute	Irregular, Dynamic	Static, Block-sparse	Regular
Comm.	Small Msg.	Big Msg.	Negligible

Dynamic, irregular computations and data in excess of one process/node motivates the use of Global Arrays (more later).

Overview of NWChem

NWChem Overview

- **Open Source License:** ECL* 2.0 — Apache* 2.0 with patent modifications for academic users (see Wikipedia* for details).
- **Wiki:** http://www.nwchem-sw.org
- **Capability:** Very diverse collection of quantum chemical methodology and QM/MM.
- **Portability:** Runs on laptops/workstations (Linux*, Mac* and Cygwin*), clusters (e.g. InfiniBand*) and supercomputers (e.g. Cray* and IBM* Blue Gene*).

NWChem History and Design

- Began at the dawn of the MPP age, before MPI*.
- First MPP code in quantum chemistry; almost every code imitates it now.
- Designed to be object-oriented but had to use Fortran* 77.
- Global Arrays programming model abstracted away explicit communication, was data-centric (i.e. what do you need to do with that matrix?).
- Uses its own memory allocator, IO layer, runtime database, hooks resource managers, low-level timers, and *communication runtime* (ARMCI).

Devil's Advocate

- Object-oriented Fortran 77? Are you insane?
- There was a time before MPI-1? And we had computers then?
- It can't be that hard to rewrite the code.

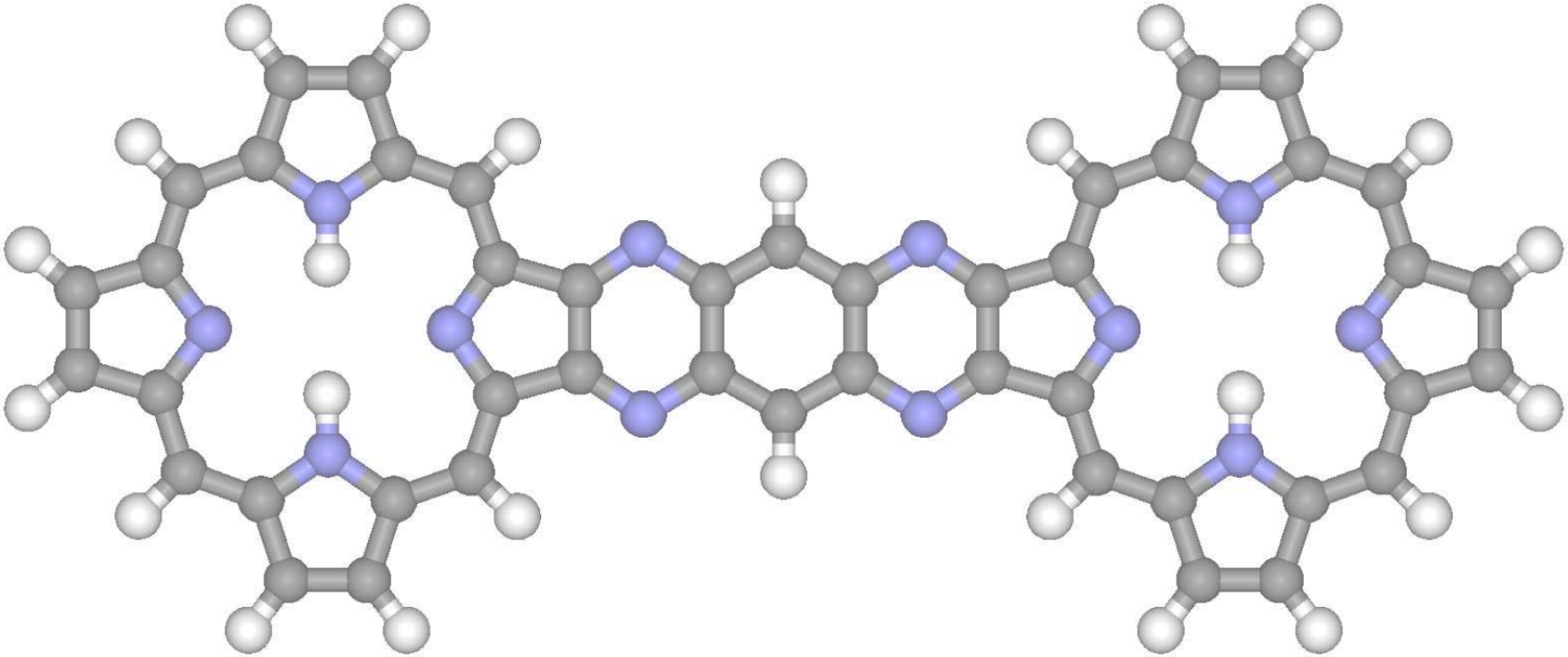
To which I say:

- Yes, it would be lovely to rewrite NWChem in C++.
- Since NWChem abstracts away communication in GA, you shouldn't see MPI either.
- Rewriting 1+ MLOC is highly nontrivial.
- New codes have to be validated.

And then there is the science...

Large coupled-cluster excited-state calculation

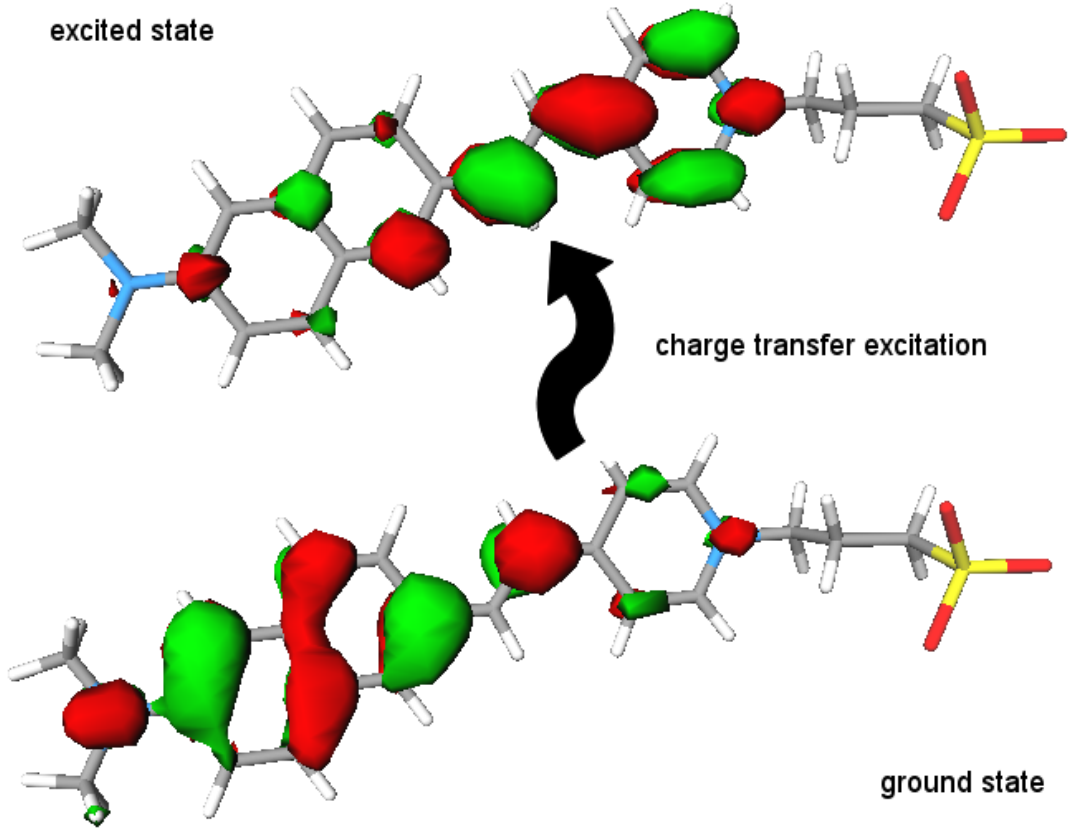
Systems with hundreds of electrons can be modeled using CR-EOMCCSD(T).



J. Chem. Phys. **132**, 154103 (2010).

Charge-transfer excited-states of biomolecules

CR-EOMCCSD(T)/6-31G* — 1 hour on 256 cores (2009/2010)

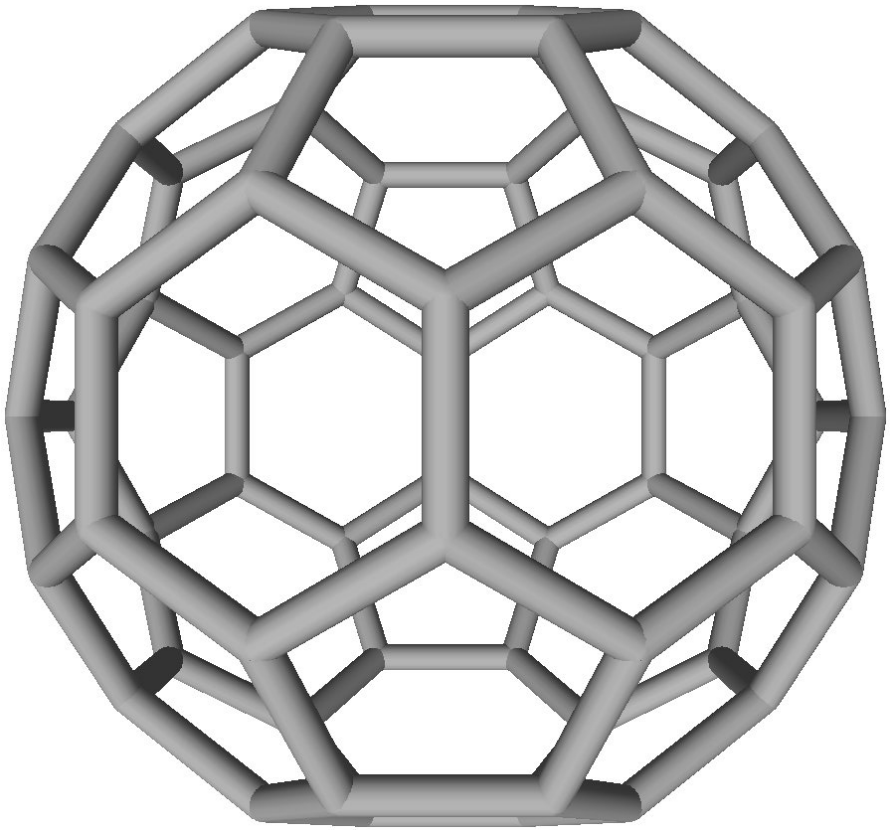


Lower levels of theory are not reliable.

Joint work with Karol Kowalski (PNNL) and Benoît Roux (UC/Argonne).

CCSD-LR Dynamic Polarizability

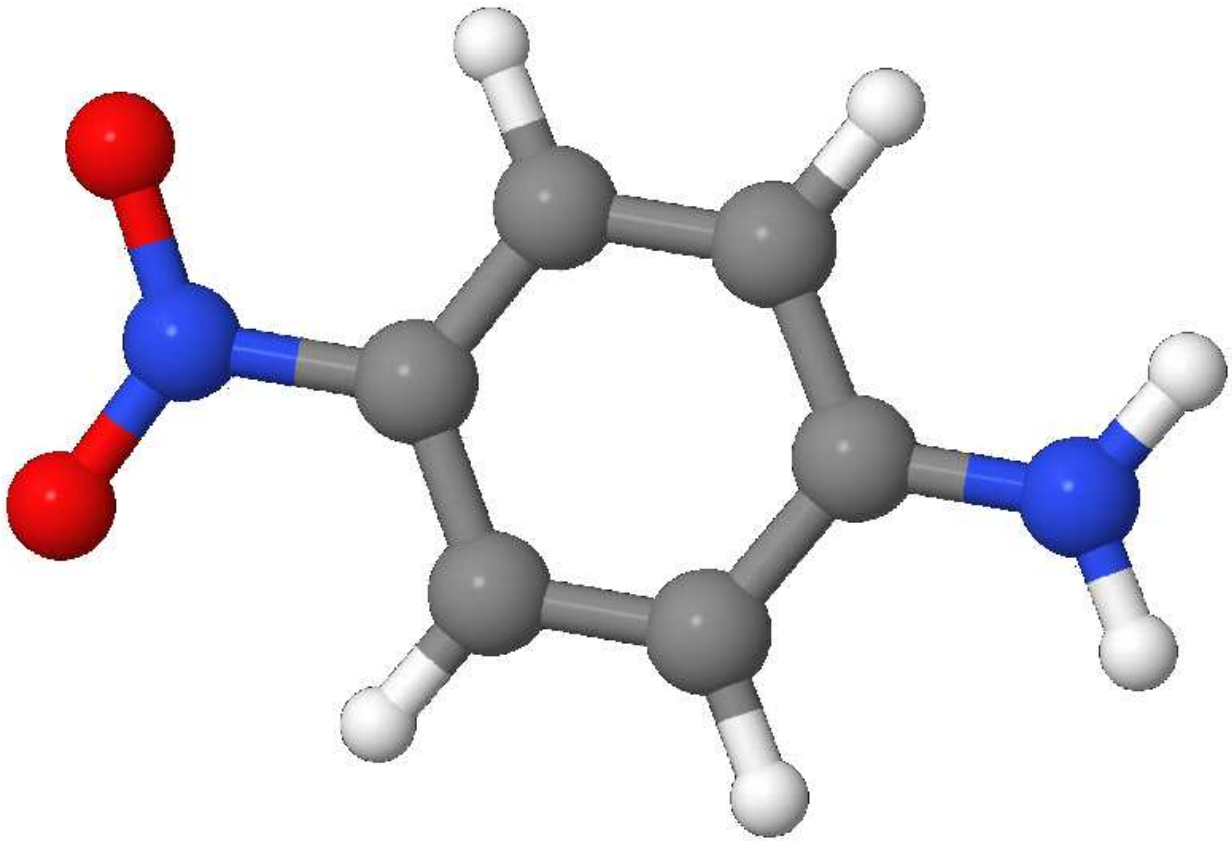
1080 b.f. — 40 hours on 1024 processors (2007)



J. Chem. Phys. **129**, 226101 (2008).

Quadratic response hyperpolarizability

CCSD/d-aug-cc-pVTZ – 812 b.f. – 20 hours on 1024 processors (2008)

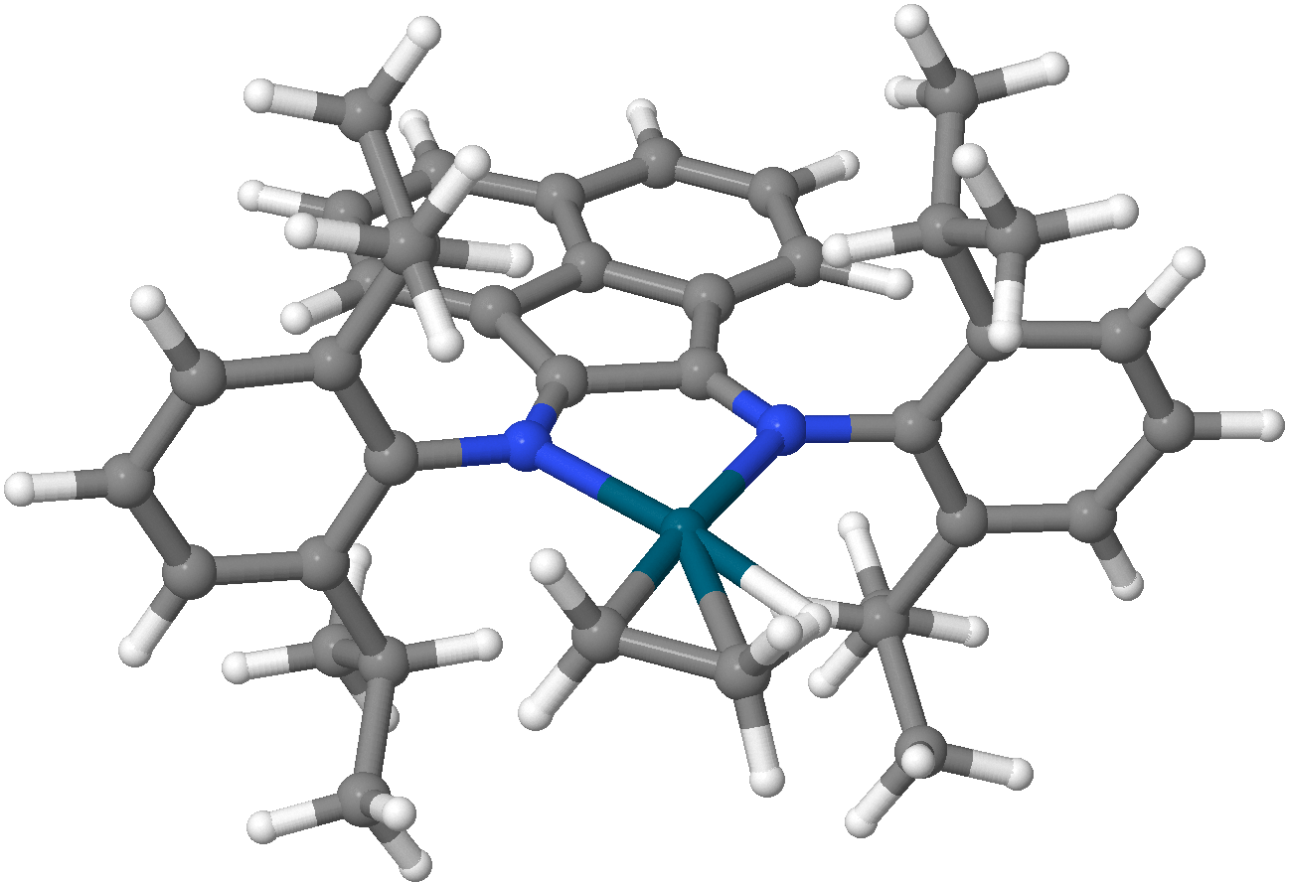


Lower levels of theory are not reliable for this system.

J. Chem. Phys. **130**, 194108 (2009).

Large Fifth-rung DFT (B2PLYP)

2154 b.f. — 7 hours on 256 cores (2008/2009)



Organometallics **29**, 1750-1760 2010.

Overview of Global Arrays

Data model

GA supports N-dimensional ($N < 8$) distributed arrays with regular (blocked and block-cyclic) and irregular (user-defined) distributions in all dimensions.

Direct local access is permitted but the primary programming model is **Get-Compute-Update**.

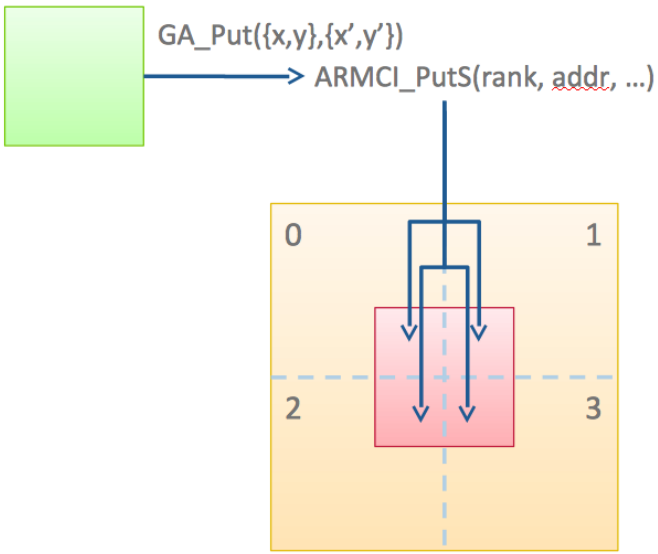
GA Template for Matrix Multiplication

Pseudocode for $C^i_j = A^i_k * B^k_j$:

```
for i in I blocks:
   for j in J blocks:
      for k in K blocks:
          if NXTVAL(me):
             Get block a(i,k) from A
             Get block b(k,j) from B
             Compute:  c(i,j) += a(i,k) * b(k,j)
      Accumulate c(i,j) block to C
```

GA default template is dynamically load-balanced effectively and weak-scales, but ignores locality and topology, may communicate excessively and ignores higher-level structure.

GA to ARMCI



- GA operations act on handle, global indices.
- ARMCI operations act on rank, virtual addresses, size.
- MPI P2P ops act on rank, virtual address, size, datatype.
- MPI RMA operations on handle, offset, datatype.

Not all GA calls map to ARMCI. Math routines call ScaLAPACK and use collective or two-sided comm.

The ARMCI Problem

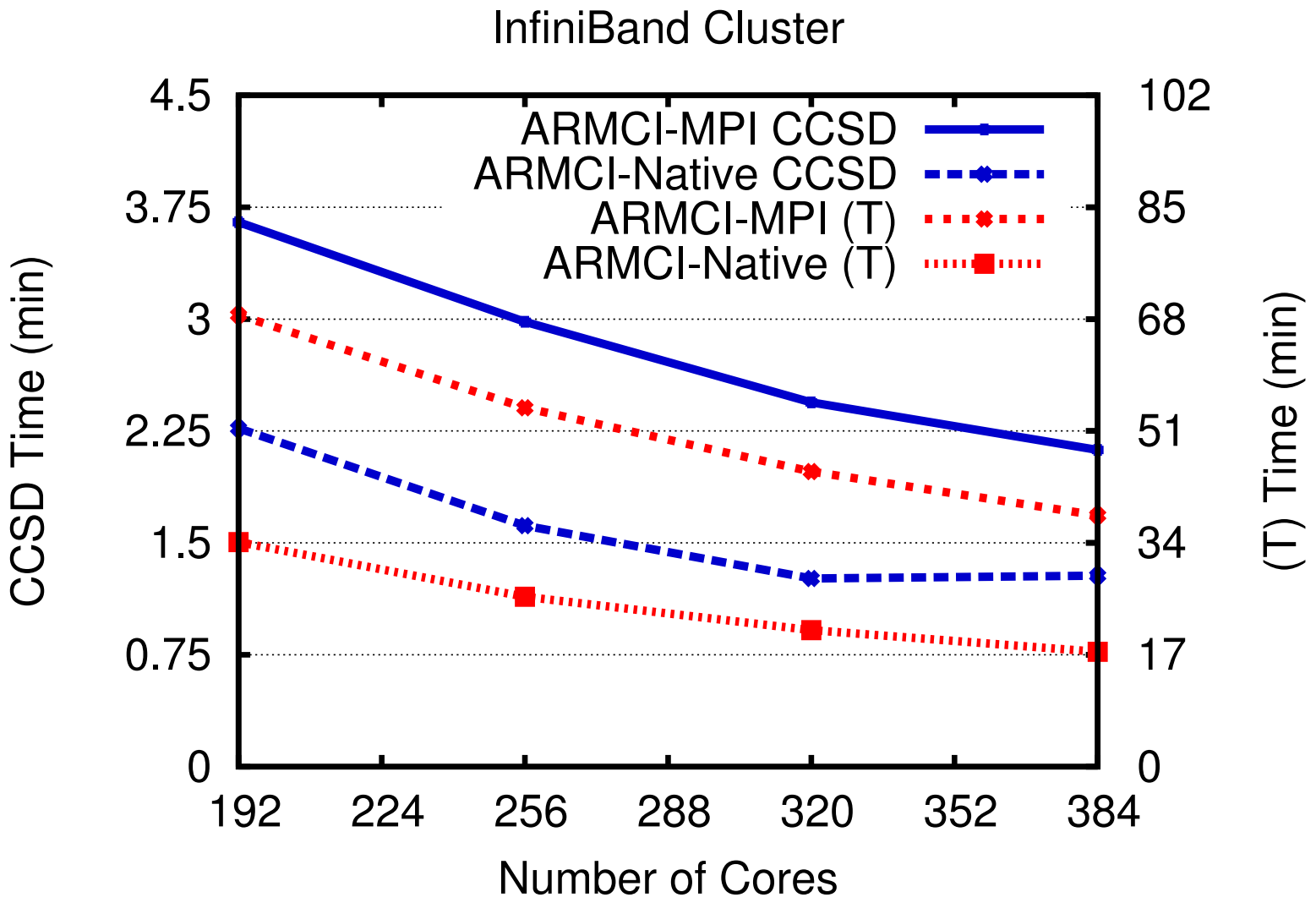
Attempts at portability of ARMCI

ARMCI is the bottleneck to porting NWChem to any new platform. Nothing else comes close anymore (thanks to Linux*, GCC*, etc.).

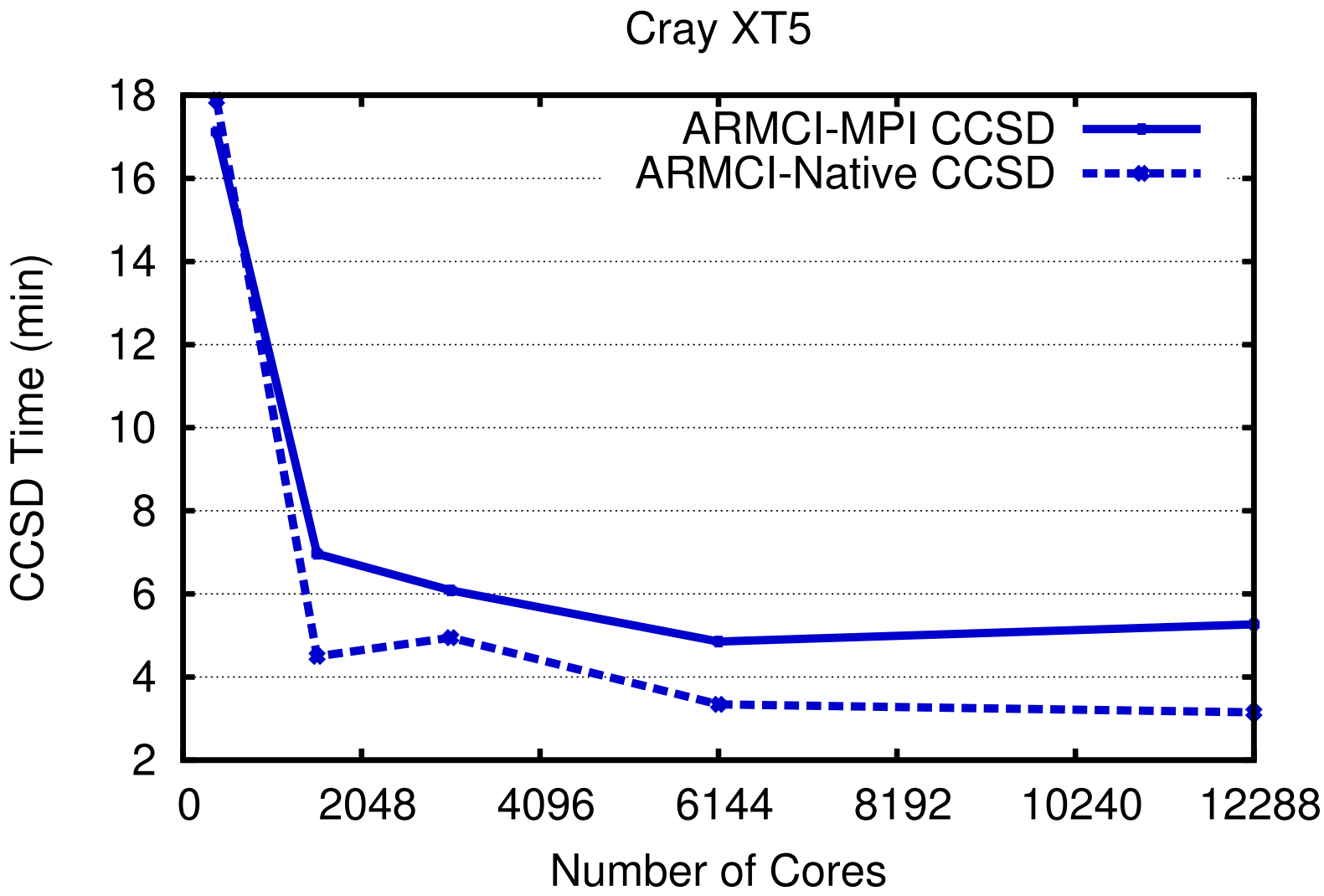
- TCP/IP performs poorly and isn't available on some supercomputers.
- Frantically writing a native port for every network - expensive.
- Cray*-oriented MPI Send+Spawn implementation of ARMCI - fragile.
- Cluster-oriented MPI Send+Threads implementation of ARMCI - slow.
- ARMCI-MPI - requires MPI-RMA to function and perform.

ARMCI-MPI with MPI-2

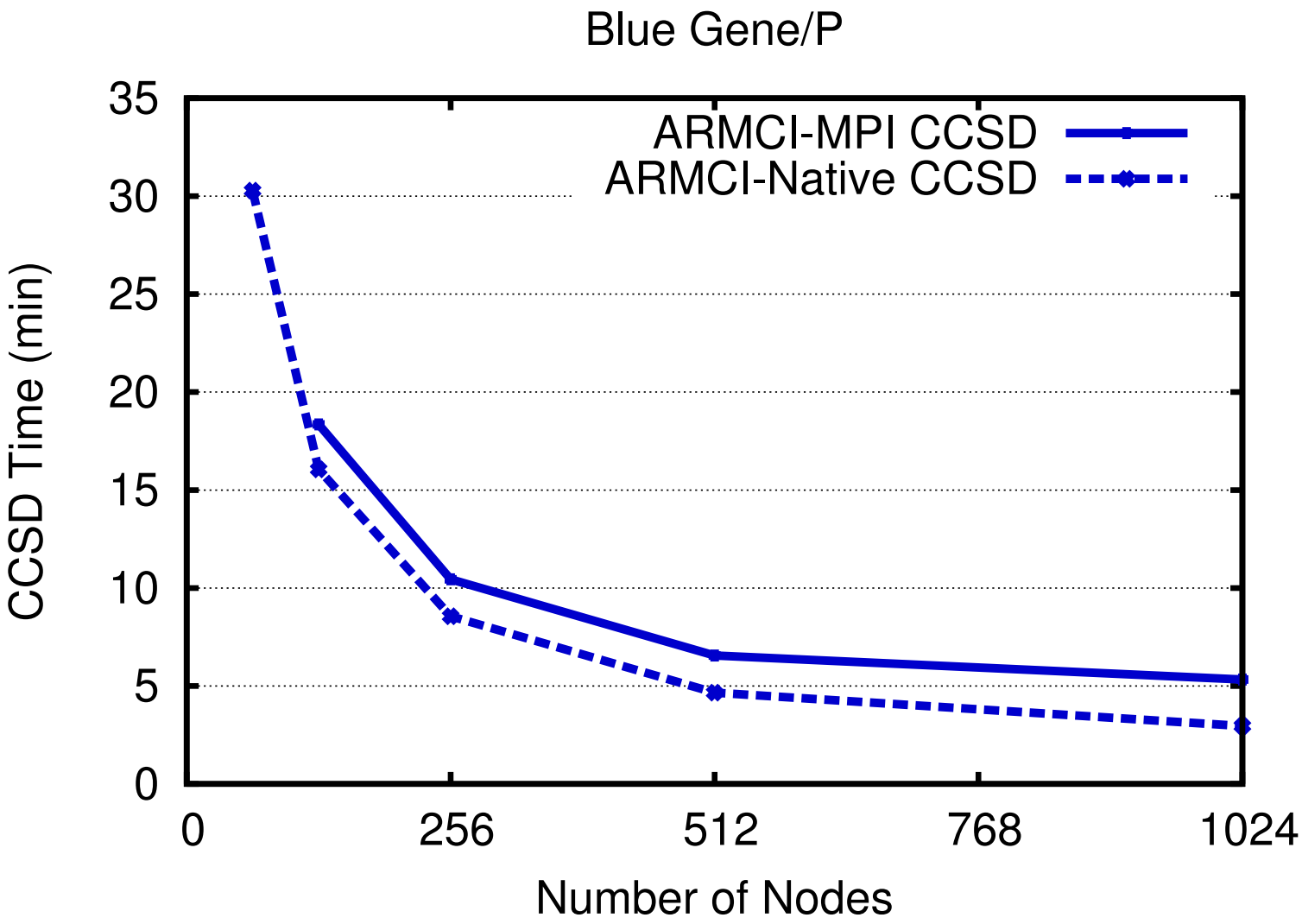
NWChem with ARMCI-MPI (MPI-2)



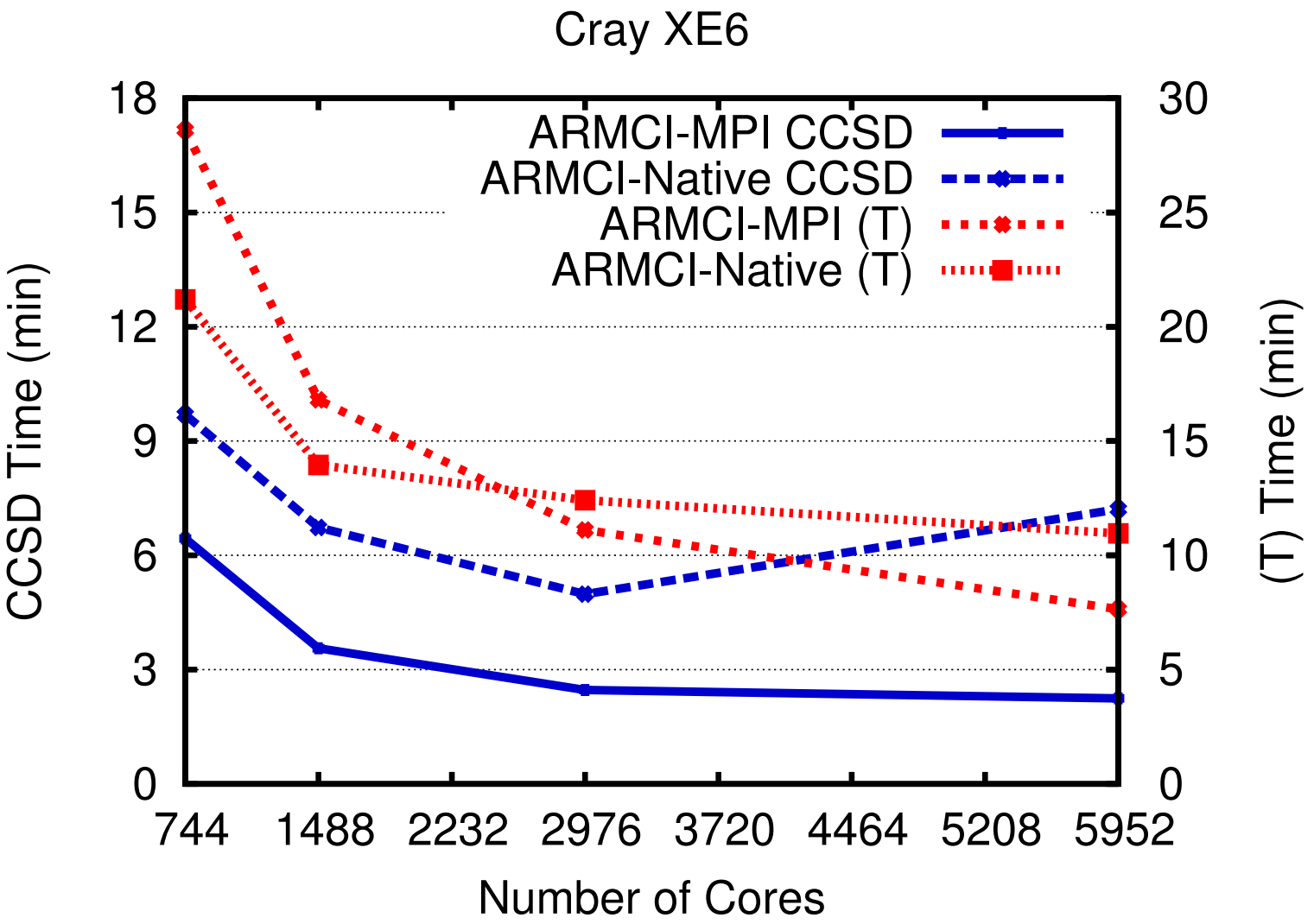
NWChem with ARMCI-MPI (MPI-2)



NWChem with ARMCI-MPI (MPI-2)



NWChem with ARMCI-MPI (MPI-2)



Issues with MPI-2

- Lacked atomics - NXTVAL is RMW(long) many-to-one.
- Nonblocking impossible.
- Local and remote completion must be combined.
- No ability to exploit symmetric or other special memory.
- Separate memory model constrains usage, is rarely necessary.
- No way to aggregate synchronization except for Win_fence.

MPI-3 RMA in a nutshell

- Added RMW and CAS.
- Request-based local completion.
- Flush_local, Flush, Flush_all, Flush_all_local.
- Win_allocate (and Win_allocate_shared).
- New memory model, shared-memory can be used.
- Win_lock_all (shared).

ARMCI-MPI with MPI-3

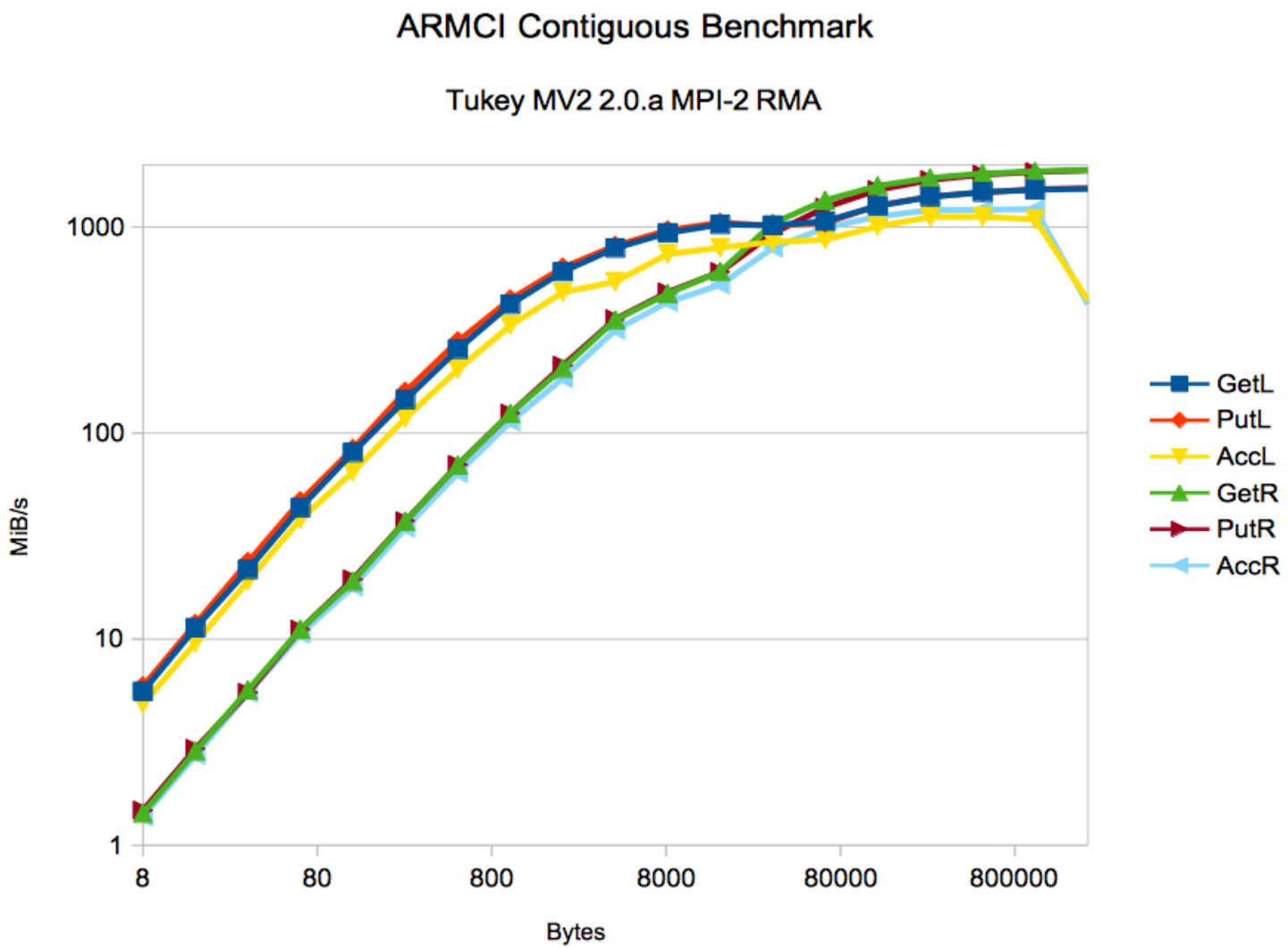
Expected changes

- RMA should improve NXTVAL latency; however, MCS mutex-based implementation was fair, did not overwhelm target.
- Separation of local and remote completion should improve bandwidth on some networks.
- Nonblocking support should improve performance when accessing multiple ranks at once.
- Allocating window memory should allow for better intranode performance.

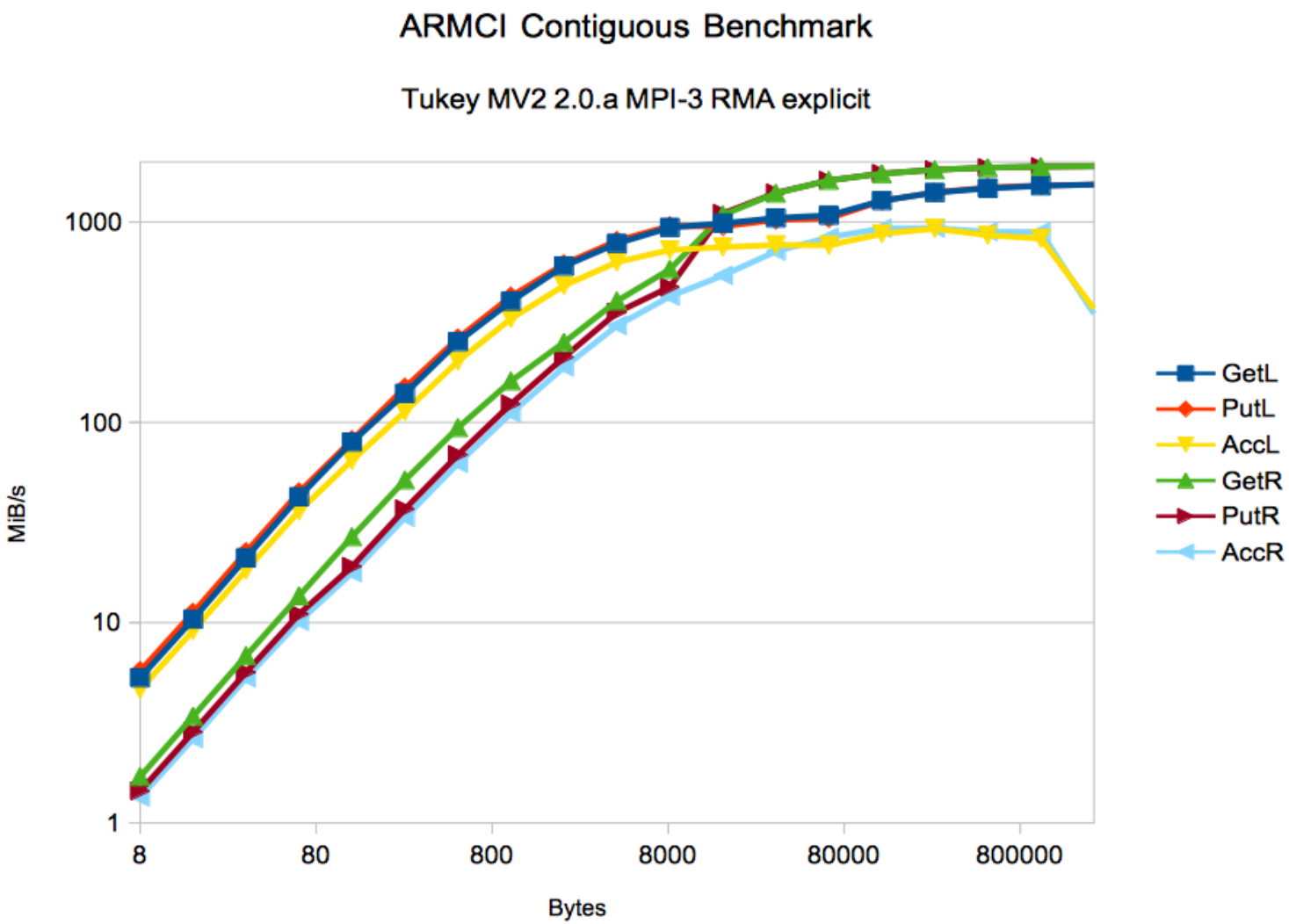
Observed changes

- NWChem is not particularly sensitive to NXTVAL except negative effects from overwhelming target as a consequence of strong-scaling, rapid injection.
- IB and Cray* networks are end-to-end complete; no obvious win from local completion.
- Approximately 4x reduction in comm time on Tianhe-2 (Galaxy Express*) at scale with the George Tech quantum chemistry code.
- Shared-memory optimizations hit numerous bugs and had to be disabled by default in ARMCI-MPI.
- MPI-3 does not change asynchronous progress situation, which remains problematic.

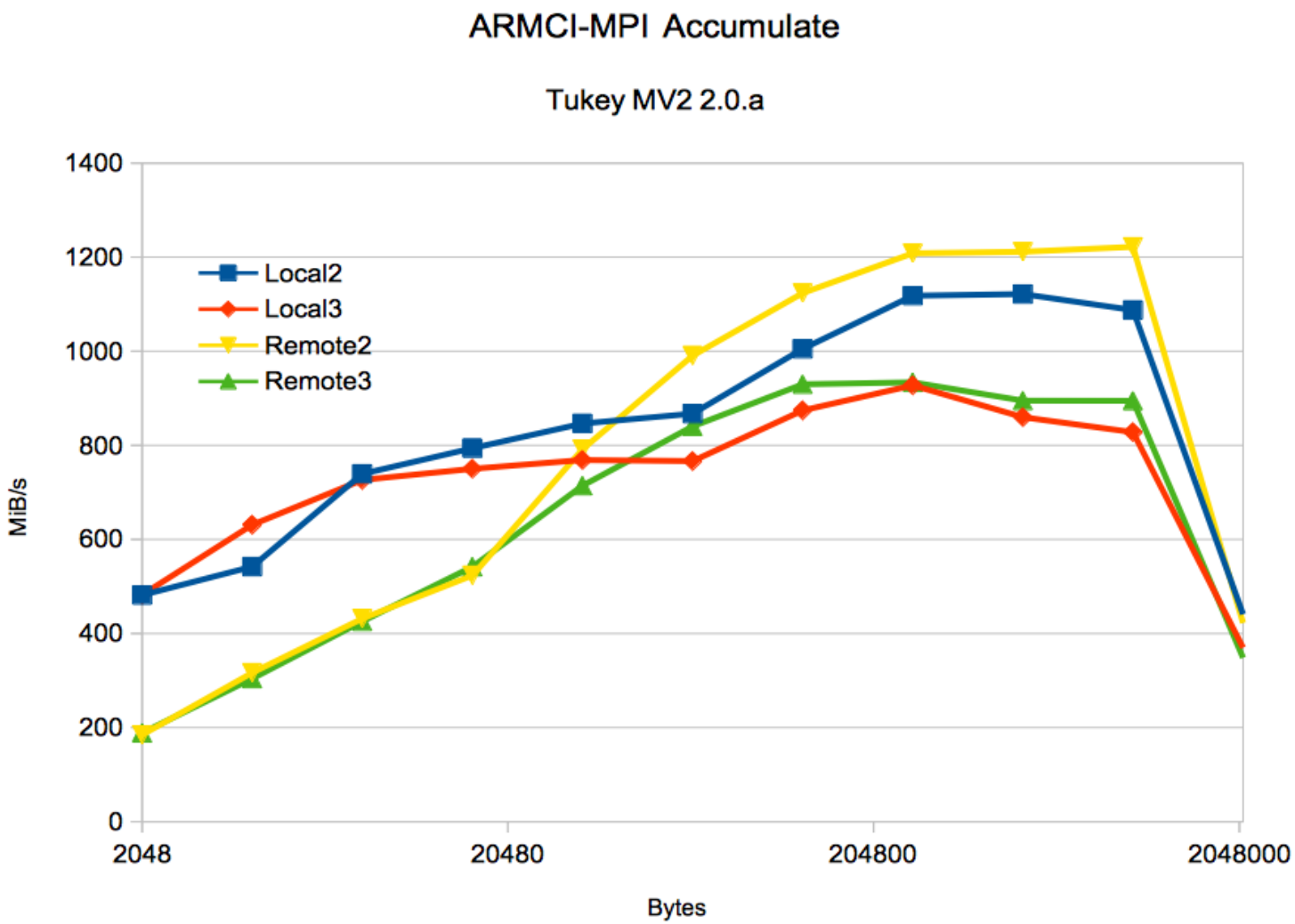
ARMCI-MPI over MPI-2



ARMCI-MPI over MPI-3



ARMCI-MPI over MPI-3



"The best performance improvement is the transition from the nonworking state to the working state." – John Osterhout

ARMCI-MPI Osterhout Performance

- No native ARMCI port on Tianhe-2; ARMCI-MPI3 scaled to 8000 nodes.
- No native ARMCI port on Blue Gene/Q*; ARMCI-MPI2 scaled to thousands of nodes.
- ARMCI native InfiniBand* port extremely unstable; segfaults nearly all the time for large-memory jobs. ARMCI-MPI able to run near the memory limit without crashing.
- ARMCI native DMAPP* port extremely unstable; fails every time in TCE. ARMCI-MPI running to 80K cores without crashing.

Final remarks

- MVAPICH2* has supported MPI-3 for a long time and enabled otherwise impossible science with NWChem.
- ARMCI-MPI enabled portability of NWChem and other quantum chemistry codes to new platforms without delay.
- MPI-3 is not yet a clear win over MPI-2 because of implementation issues (bugs).
- When ARMCI native port exists and works, it's usually faster.

Please see http://wiki.mpich.org/armci-mpi/index.php/Main_Page for details.