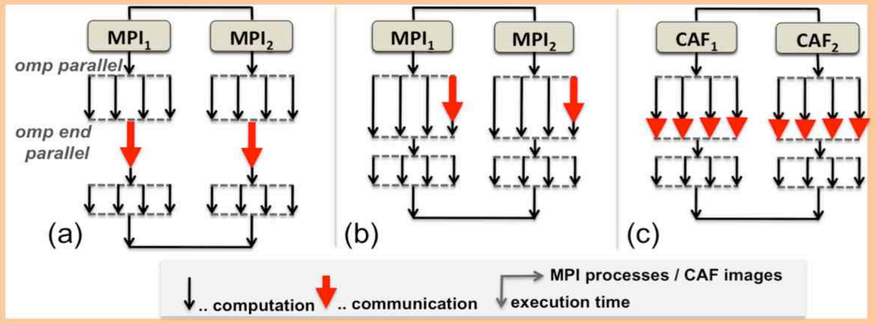
Global Address Space Communication Techniques for Gyrokinetic Fusion Applications on Ultra-Scale Platforms, *SC'11 best paper finalist*

R Preissl[1], N Wichmann[2], B Long[2], J Shalf[1], S Ethier[3] and A Koniges[1]

[1]Berkeley Lab (USA), [2]CRAY Inc. (USA), [3]Princeton Plasma Physics Laboratory (USA)

A majority of the work was supported by the Petascale Initiative in Computational Science at NERSC with additional support by the Cray Center of Excellence at NERSC. Additionally, we are grateful for interactions with Nicholas Wright (NERSC) and for the extended computer time as well as the valuable support from NERSC. This work was supported by the Director, Office of Science, Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.



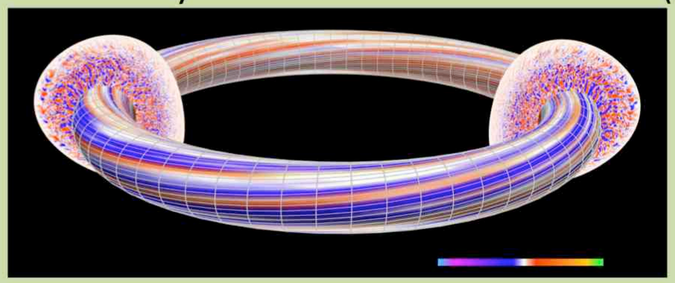
Programming Models:

(a) Classical hybrid MPI/OpenMP programming model

(b) Extension of classical hybrid model where MPI thread teams for work distribution and collective MPI function calls

(c) The advanced hybrid PGAS/OpenMP algorithm builds on the strategy of communicating threads, but allows ALL OpenMP threads per team to make communication calls to the thread-safe PGAS communication layer

Application: Gyrokinetic Tokamak Simulation (GTS)



- A global 3D Particle-In-Cell (PIC) code with MPI & OpenMP support
- Developed to study plasma microturbulence in toroidal, magnetic confinement devices called tokamaks
- Microturbulence is a complex, nonlinear phenomenon that is believed to play a key role in the confinement of energy and particles in fusion plasmas
- Due to one of the levels implemented for parallelism in GTS, particles from one toroidal domain to another while they travel around the torus
- **Shift phase** represents the most communication intense routine in GTS and will gain in importance when scaling GTS to Petascale or even Exascale supercomputers.

Shift Algorithms: Two-Sided (MPI-1) vs. One-Sided (PGAS)

MPI

```
!(1) Prepost receive requests
do i=1,nr_dests
  MPI_IRECV(recv_buf(i),i,req(i),tor_comm,..)
enddo

!(2) compute shifted particles and fill buffer
!$omp parallel
pack(p_array,shift,holes,send_buf)

!(3) Send of particles to destination process
do j=1,nr_dests
  MPI_ISEND(send_buf(j),j,req(j+i),tor_comm,..)
enddo
MPI_WAITALL(2*nr_dests,req,..)
```

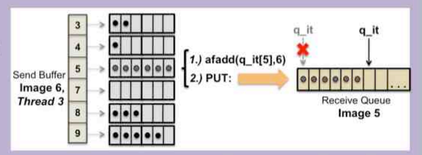
- MPI algorithms have been extensively researched and optimized in the past
- Multithreaded MPI algorithms exploiting shared memory work-sharing constructs + novel overlapping techniques, e.g., using OpenMP task, have been developed
- All optimized MPI algorithms share the large bulk data transfers to exchange moving particles following rule that performance is optimized by sending fewer & larger messages
- Vendor-supplied MPI library that has been highly tuned for the XE6/Gemini interconnect

Coarray Fortran

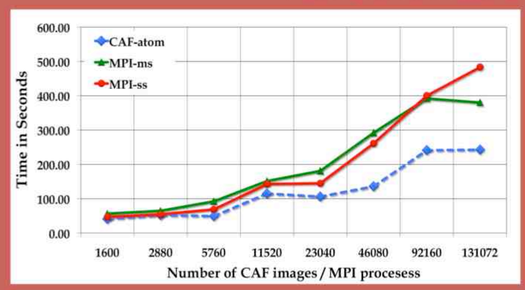
```
!(1) compute shifted particles and fill the
!  receiving queues on destination images
!$omp parallel do schedule(dynamic,p_size/100)&
!$omp private(s_buf,buf_cnt) shared(recvQ,q_it)
do i=1,p_size
  dest=compute_destination(p_array(i))
  if(dest.ne.local_toroidal_domain) {
    holes(shift++)=i
    s_buf(dest,buf_cnt(dest)++)=p_array(i)
    if(buf_cnt(dest).eq.sb_size) {
      q_start=afadd(q_it[dest],sb_size)
      recvQ(q_start:q_start+sb_size-1)[dest] &
        =s_buf(dest,1:sb_size)
      buf_cnt(dest)=0  }  }
enddo

!(2) shift remaining particles
empty_s_buffers(s_buf)
!$omp end parallel
```

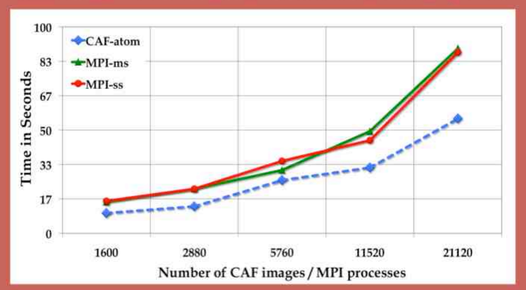
- Exploit the one-sided nature of the PGAS model
- Lightweight data transfers without synchronization between the sending & receiving images
- Sending more frequent smaller messages enables the Coarray approach to outperform the MPI-1 implementations due to enhanced communication and computation overlap as well as the better network bandwidth utilization
- A similar strategy in MPI-1 cannot be implemented and very hard in MPI-2
- One-sided messaging semantics as language constructs (Coarrays in Fortran 2008) allow compilers to directly reference remote memory and to apply communication optimizations
- HW support for PGAS, as provided by the recent Cray XE6 Gemini interconnect, is essential to realize the performance potential; but, still good performance improvements on XT4!



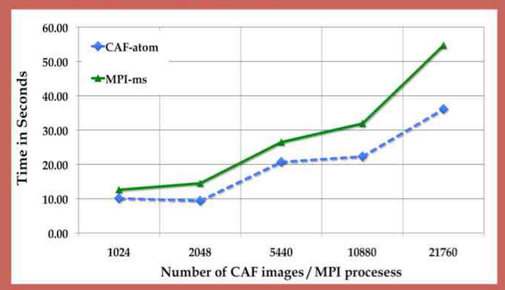
Performance: Evaluated in a benchmark suite & in the real application



Single-Threaded (Benchmark Suite)



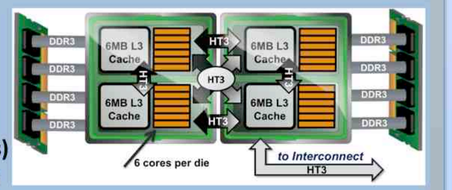
Multi-Threaded (Benchmark Suite)



Multi-Threaded (GTS)

Architecture: NERSC CRAY XE6 "Hopper"

Compute Node Configuration:



- 6384 nodes
- 2 twelve-core AMD 'MagnyCours' 2.1 GHz processors per node (NUMA)
- 24 cores per node (153,216 total cores)
- 1.28 Peta-flops for the entire machine
- Each core has their own L1 and L2 caches, with 64 KB and 512KB respectively
- 6 MB L3 cache shared between 6 cores on the Magny-Cours processor
- 4 DDR3 1333 MHz memory channels per twelve-core 'MagnyCours' processor

Gemini Characteristics:

- 168 GB/sec routing capacity
- Scales to over 100,000 network endpoints
- Link-level reliability and adaptive routing
- Provides global address space
- Efficiently supports MPI, one-sided MPI, Shmem, UPC, Co-Array Fortran
- Supports millions of MPI messages per second
- Internode latency on the order of 1 microsecond
- Bandwidth of 9.8 GB/sec per Gemini chip