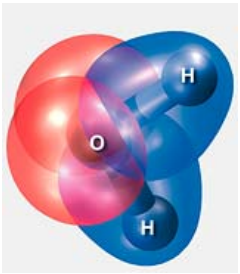
Performance evaluation and optimization of the ls1-MarDyn Molecular Dynamics code on the Cray XE6

Christoph Niethammer

CUG, May 1st, 2012, Stuttgart

Outline

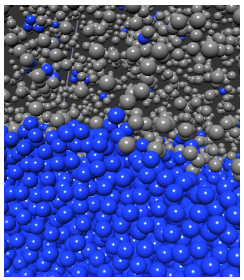
Simulation Methods



[1]

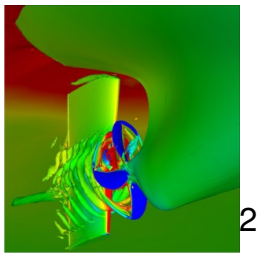
Quantum Mechanics

$$i\hbar\frac{\partial}{\partial t}|\psi(\mathbf{r},t)\rangle = \mathcal{H}|\psi(\mathbf{r},t)\rangle$$



Classical Molecular Dynamcis

...


[2]

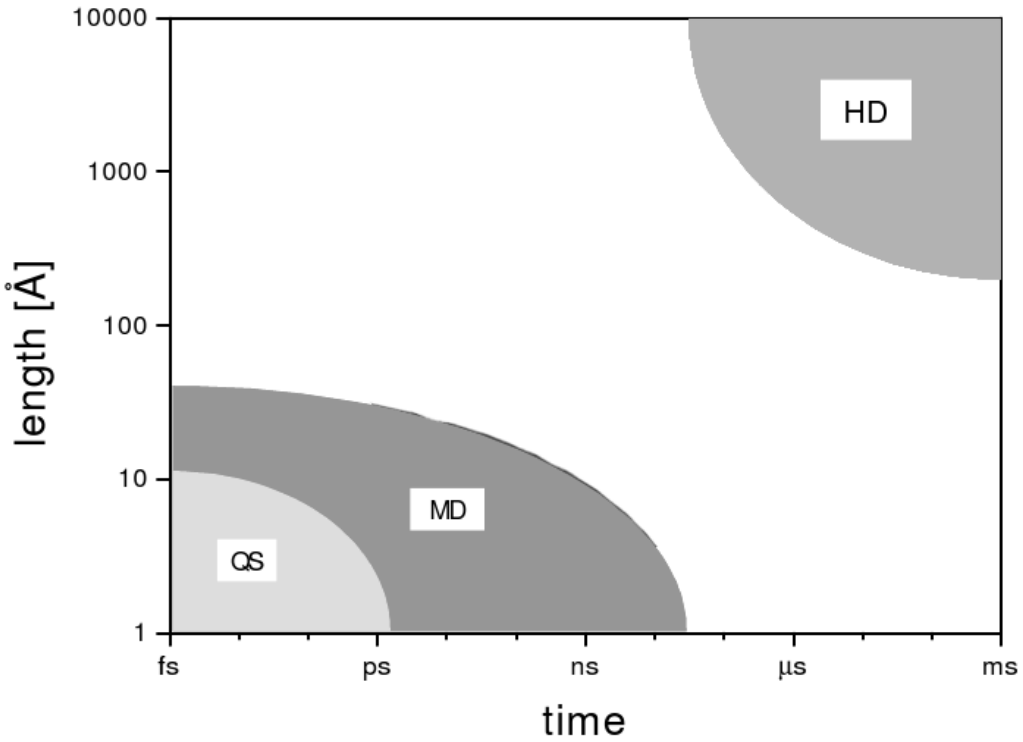
Computational Fluid Dynamics

$$\rho\dot{\mathbf{v}} = -\nabla p + \eta\Delta\mathbf{v} + (\lambda+\eta)\nabla(\nabla\cdot\mathbf{v}) + \mathbf{f}$$

[1] http://www.scinexx.de

[2] http://www.iihr.uiowa.edu/~shiphydro/

Time and Length Scales of Simulation Methods

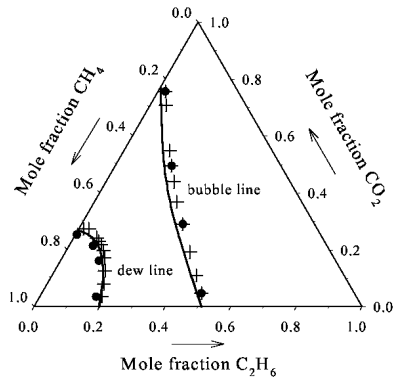


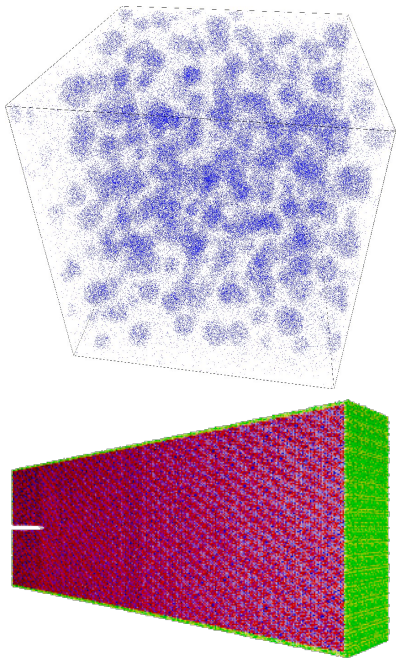
[Godehard Sutmann, 2002]

- Molecular Dynamics (MD) is the link between Quantum simulations (QS) and Hydrodynamics (HD)
- Gap between MD and HD becomes smaller with increasing compute power

Apllication of Molecular Dynamic Simulations

- gas mixtures (MS2):
  thermodynamic properties
- nanofluids (ls1-MarDyn):
  condensation, viscosity
- solid state (IMD):
  crack distribution, diffusion

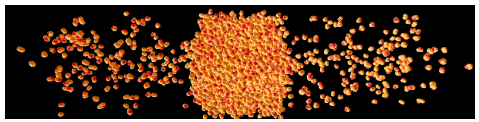




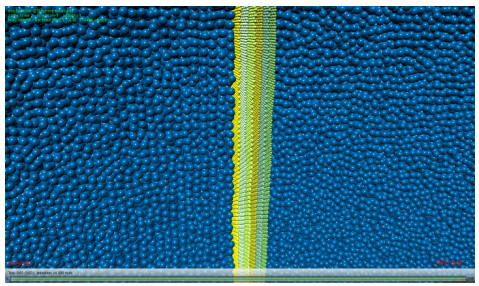
ls1-MarDyn

- Written in C++
- Modular concept:
  - molecule container
  - force adapter
  - event based integrator
  - output plugins
- Works with most C++ compilers:
  GNU, Intel, PGI, Cray, Pathscale, NEC SX, Open64
- Works with any standard compliant MPI implementation

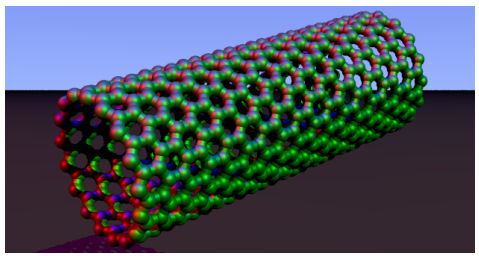
- condensation:



- viscosity



- carbon layers and nanotubes

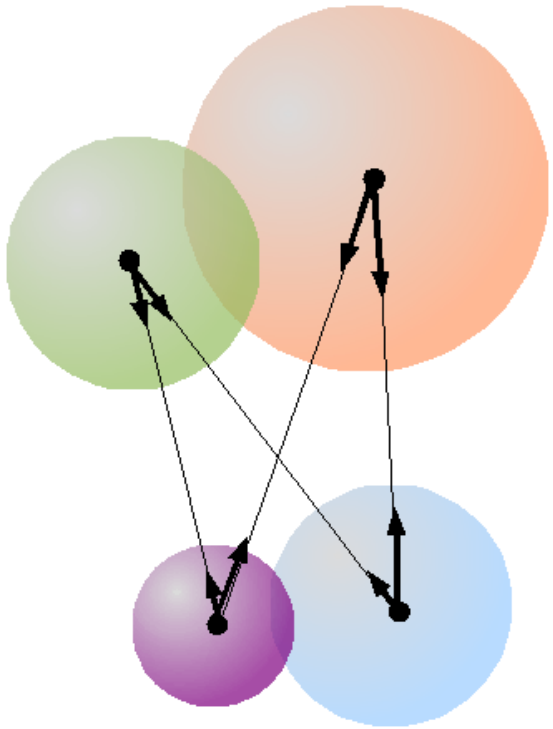


Classical Molecular Dynamics

- Closed system
- Particle model describing molecules $\rightarrow$ point mechanics
- Determination of particle trajectories by integration of NEWTON's equation of motion
  $m\ddot{x}(t) = \vec{F}(t)$
- Interaction between molecules determined from simple potential functions
  $\vec{F}_{ij} = -\nabla\Phi(r_{ij})$
- Potential parameters may be obtained from real experiments or QM simulations

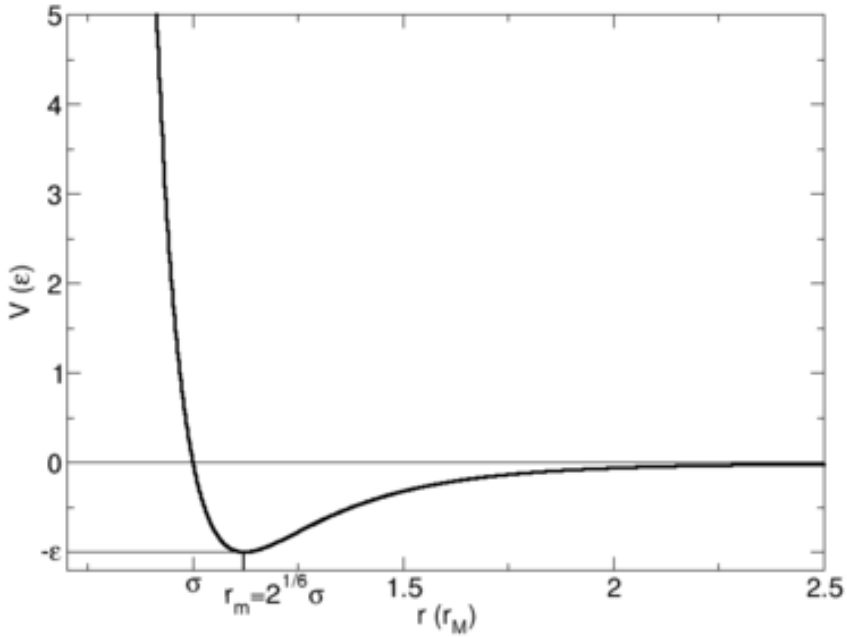
Rigid rotator model

- Molecules described as rigid group of interaction sites
- Combination of all acting forces to one force acting onto the center of mass
- Additional rotational degree of freedom
- Momentum relative to the center of mass



LENNARD-JONES-Potential

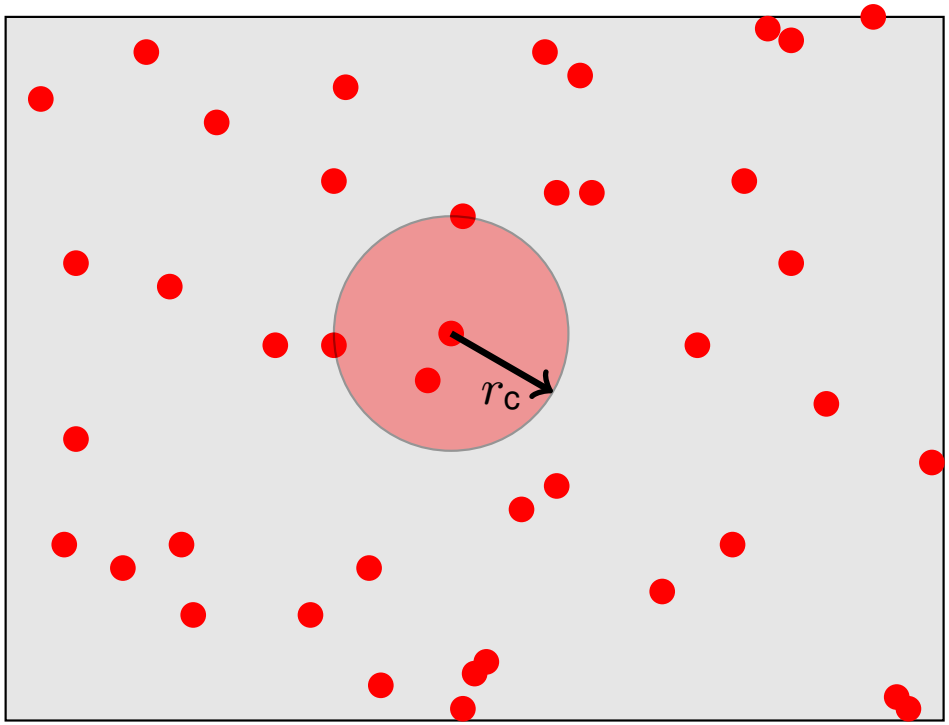
- Well suited to describe noble gases
- Attractive part: $-\frac{C}{r_{ij}^6}$
- Repulsive part: $\begin{cases} \frac{C_n}{r_{ij}^n} \\ \gamma\ \mathrm{e}^{-r/r_0} \end{cases}$



LENNARD-JONES-(12,6)-Potential

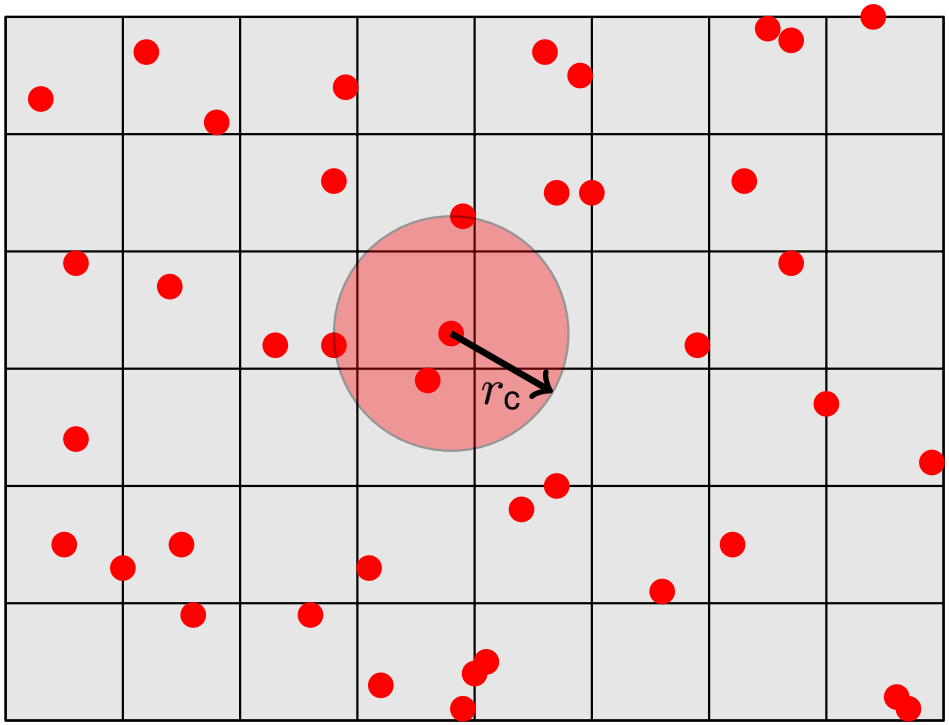
$$\Phi_{\mathrm{LJ}}(r_{ij}) = 4\epsilon \left\{ \left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6} \right\}$$

Link-cell method for short range potentials



introduce **cutoff radius** r_c:

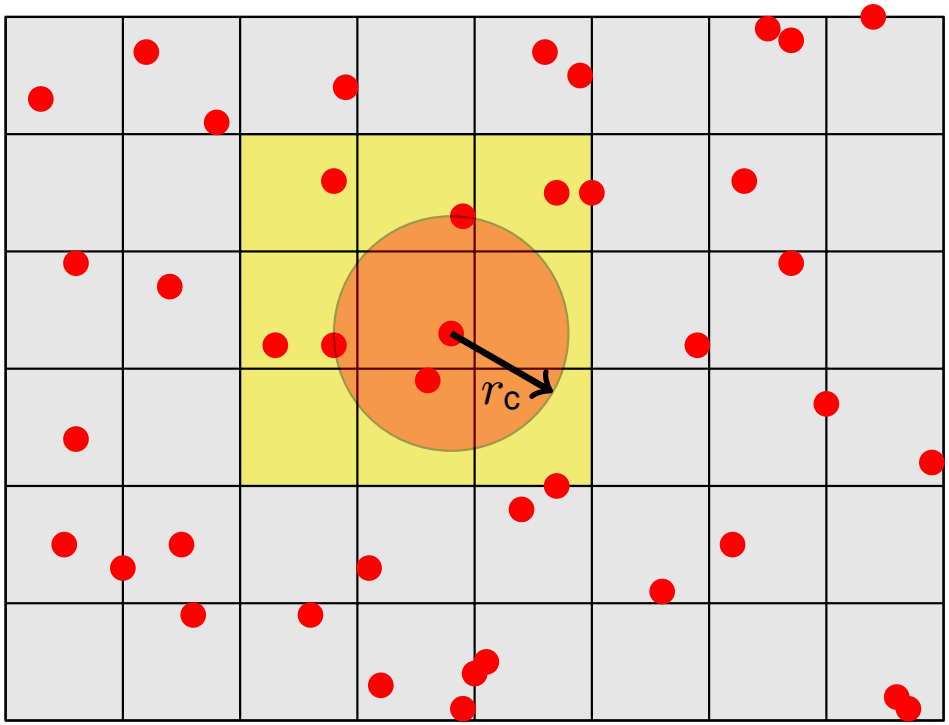
Link-cell method for short range potentials



introduce **cutoff radius** r_c:

- split domain into cells with length of the cutoff radius r_c
- sorting particles into cells takes $\mathcal{O}(n)$ operations

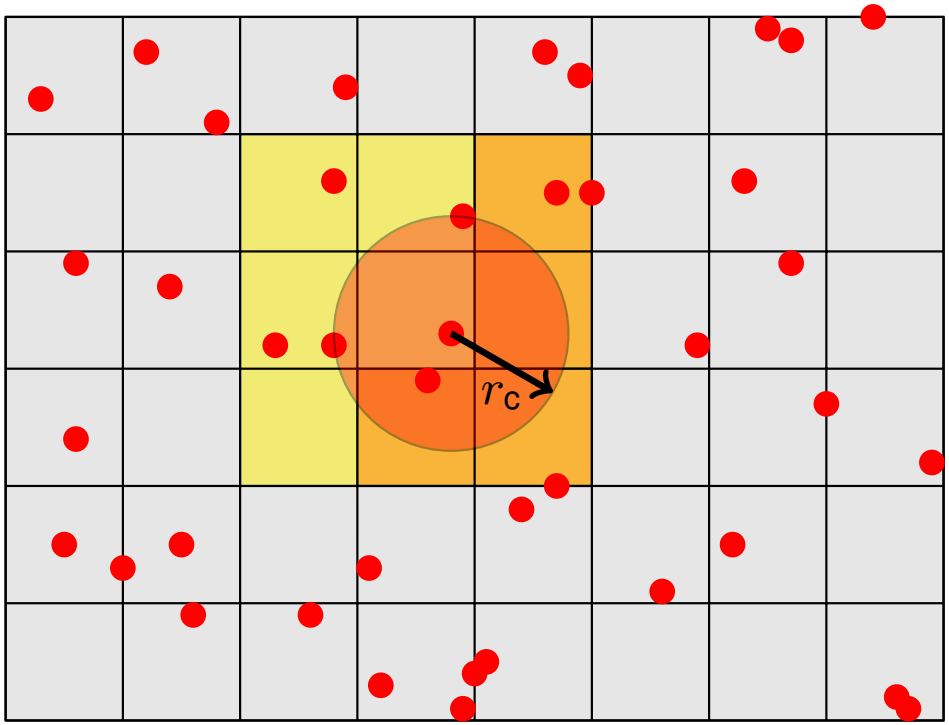
Link-cell method for short range potentials



introduce **cutoff radius** r_c:

- split domain into cells with length of the cutoff radius r_c
- sorting particles into cells takes $\mathcal{O}(n)$ operations
- interacting particles are either in the same or in a neighbour cell

Link-cell method for short range potentials



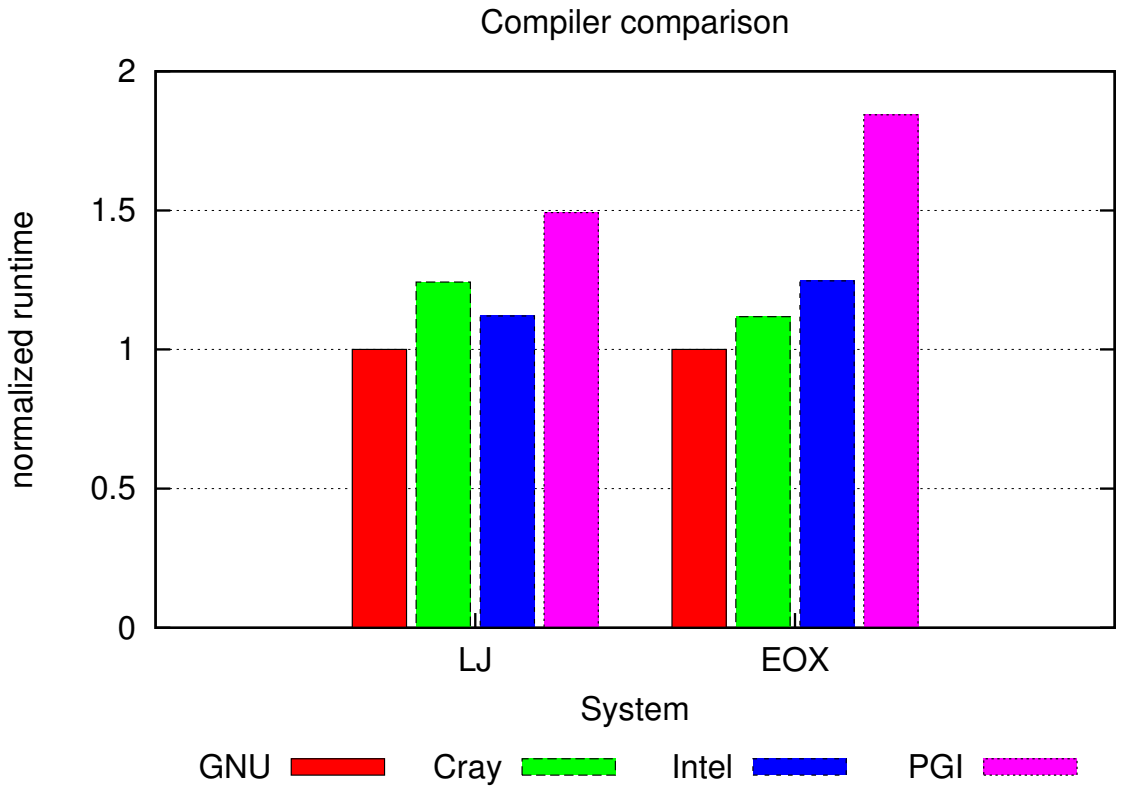
introduce **cutoff radius** r_c:

- split domain into cells with length of the cutoff radius r_c
- sorting particles into cells takes $\mathcal{O}(n)$ operations
- interacting particles are either in the same or in a neighbour cell
- NEWTONs 3rd law halves number of relevant neighbours

The Hermit system at HLRS

- Peak performance: 1.045 PFlops
- Number of compute nodes: 3552
- Number of compute cores: 113,664
- Processor: Dual Socket AMD Interlagos @ 2.3GHz 16 cores each
- Memory/node: 32 GB and 64 GB
- Interconnect: CRAY Gemini, 3D torus

ls1-MarDyn compiler comparison on Hermit



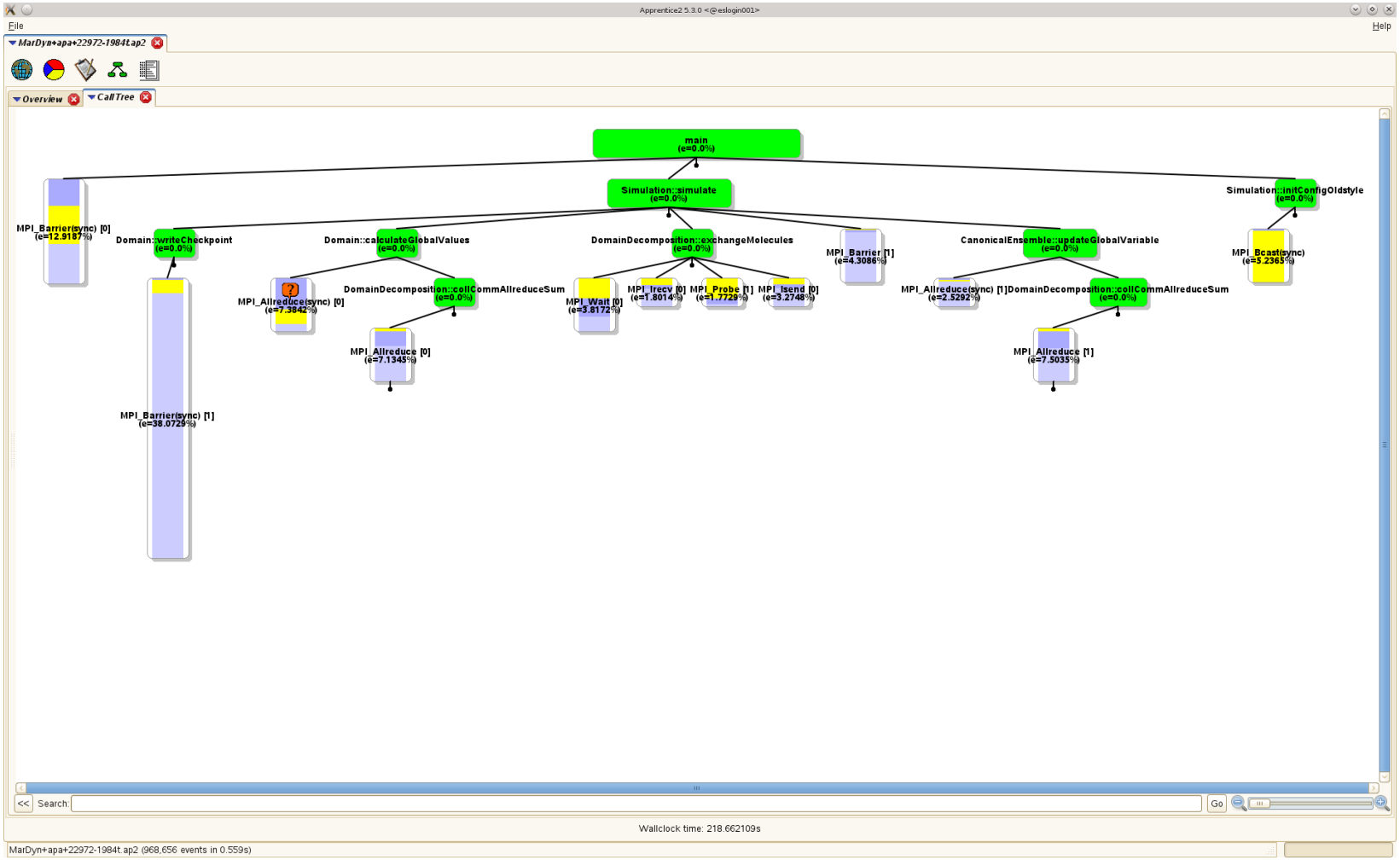
GCC	4.6.2	-O3
Cray	8.0.3	-O3
Intel	12.1.3	-O3
PGI	12.2	-fast -Mipa=fast,inline -Minline=levels:10

ls1-MarDyn PAPI results on Hermit

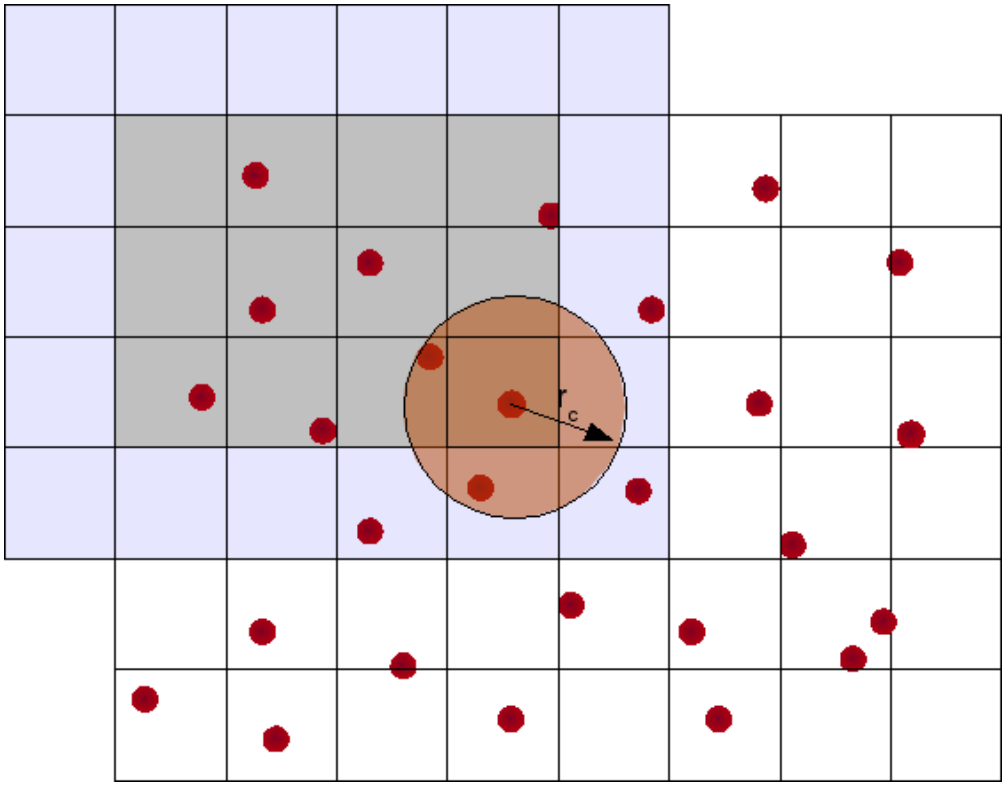
TLB utilization	28441.79	refs/miss	55.550 avg uses
D1 cache hit,miss ratios	99.3%	hits	0.7% misses
D1 cache hit,refill ratio	99.1%	hits	0.9% refills
D1 cache utilization (misses)	144.50	refs/miss	18.063 avg hits
D1 cache utilization (refills)	114.25	refs/refill	14.281 avg uses
D2 cache hit,miss ratio	93.8%	hits	6.2% misses
D1+D2 cache hit,miss ratio	100.0%	hits	0.0% misses
D1+D2 cache utilization	2312.90	refs/miss	289.113 avg hits

- D1 & D2 cache hit rate is very good
- D1 & D2 cache utilization is good (calculations done in double precision)
- Improving TLB utilization hard due to the link cell algorithm

Craypat calltree of ls1-MarDyn



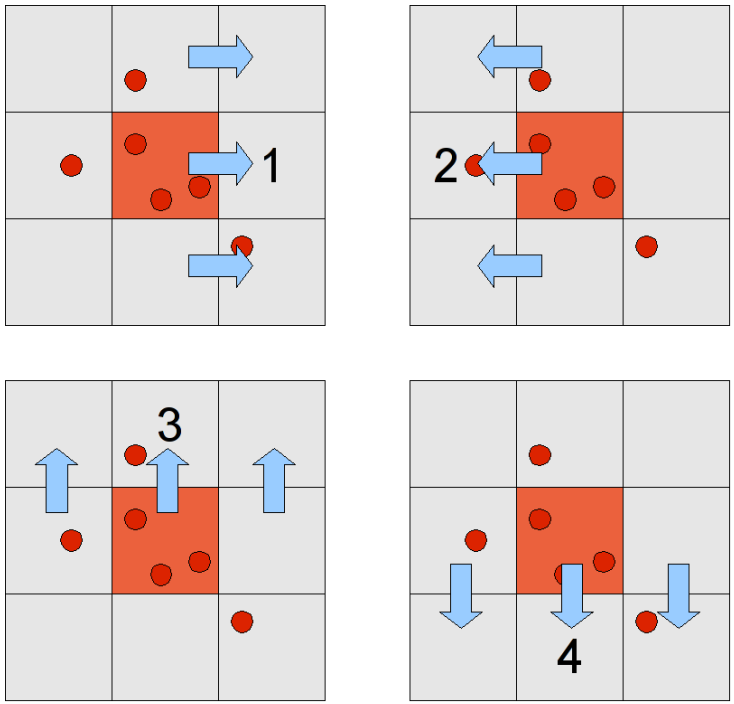
Domain decomposition



- decompose domain into sub-domains
- halos hold molecule data from neighbours' boundary areas
- halo needs updates in each time step

Halo area implementatin in ls1-MarDyn

- Particle exchange done direction wise for x, y and z-direction in upward and downward direction
- Particle exchange in one direction can be performed in parallel (e.g. 1+2 and 3+4)
- Implementation can make use of 3D topology
- ls1-MarDyn uses nonblocking MPI calls to overlap packing and unpacking of messages with communication



Using MPI rank reordering

- Static communication pattern for neighbour particle exchange known
- Hermit has 3D torus network structure
- use grid-order tool to build custom rank file
- craypat provides automated communication pattern detection

RANK_ORDER	4096 PEs
round-robin	15.8(7)
SMP-style	15.5(3)
folded-rank	15.9(7)
custom (grid-order)	15.4(6)
custom (craypat)	15.1(2)

$\Rightarrow$ 3% improvement compared to default (SMP) shows the efficiency of overlapping communication and computation using MPI's nonblocking send and receive calls.

Collectives within ls1-MarDyn

- Computation of global values requires `MPI_Allreduce` which is bad for scalability
- Replace multiple allreduce calls for every variable in the initial code by derived data type and custom reduction function.

```
void allreduceSum() {
    setMPIType();
    MPI_Op reduceOp;
    MPI_Op_create((MPI_User_function *) CollectiveCommunication::add, 1, &reduceOp);
    MPI_Allreduce(_sendValues, _recvValues, 1, _valuesType, reduceOp, _communicator);
    MPI_Op_free(&reduceOp);
    MPI_Type_free(&_valuesType);
}
```

PEs	not-agglomerated	agglomerated	improvement
4096	68.57	61.83	9.83%
8192	45.51	41.34	9.18%
16374	41.17	36.95	11.95%

Tuning parameters for MPI collectives

Cray MPI provides a variety of tuning parameters:

- `MPICH_USE_DMAPP_COLL` enables optimized DMAPP collective algorithms
- `MPICH_COLL_SYNC` performs a barrier before each MPI collective
- `MPICH_REDUCE_NO_SMP` allows to disables smp-aware algorithms

	4096 PEs
none	15.5(0)
`MPICH_COLL_SYNC`	16.9(4)
`MPICH_USE_DMAPP_COLL`	15.(58)
`MPICH_REDUCE_NO_SMP`	15.5(2)

ls1-MarDyn I/O system

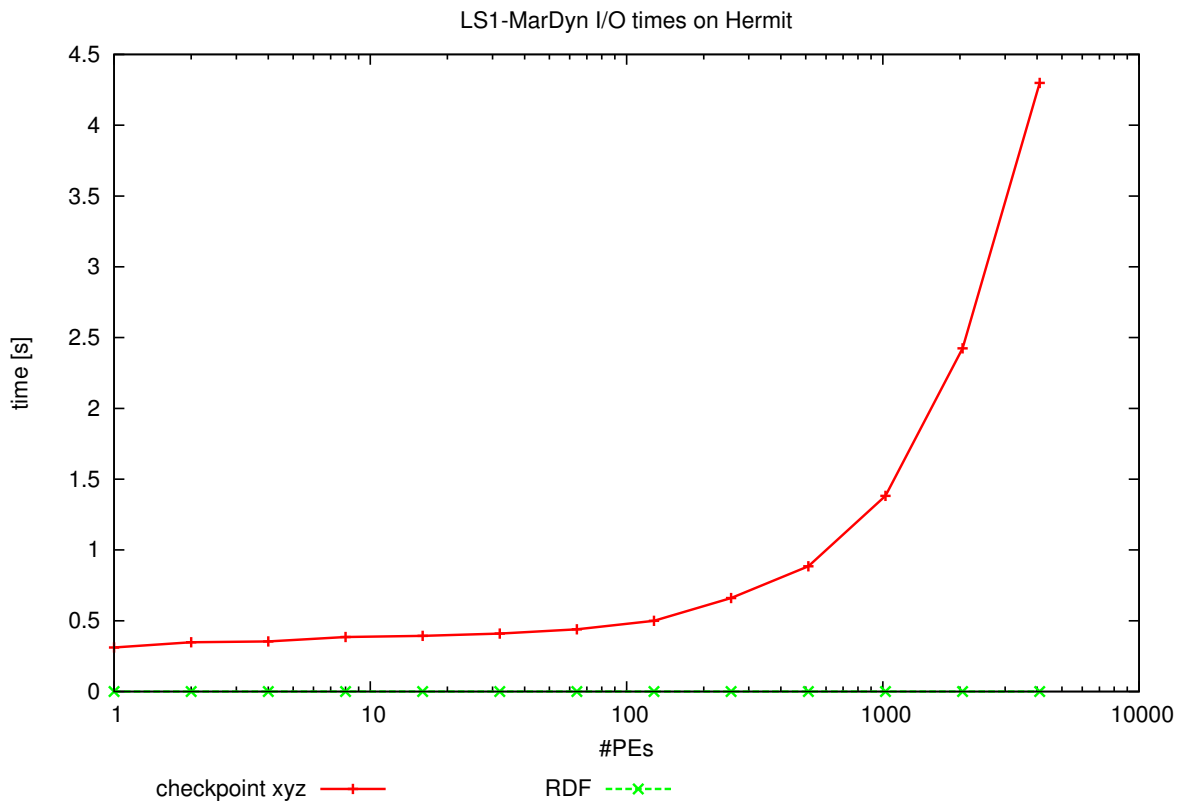
ls1-MarDyn has a pluggable I/O infrastructure which can be used for

- Checkpointing
- Result writing
- Visualization
- Program statistics

ls1-MarDyn I/O system

ls1-MarDyn has a pluggable I/O infrastructure which can be used for

- Checkpointing
- Result writing
- Visualization
- Program statistics



ls1-MarDyn I/O times on Hermit

The *iobuf* report about the input/restart file reading for one of the processes:

```
PE 1086: File "lj40000_t300.inp"
            Calls     Seconds     MB/sec
Read            1    2.185619   0.003748
Open            1    0.278424
Close           1    0.731858
Buffer Read     2    3.685361   0.569049
I/O Wait        2    2.184243   0.960128
Buffers used        2 (2 MB)
Prefetches          1
```

ls1-MarDyn I/O improvement

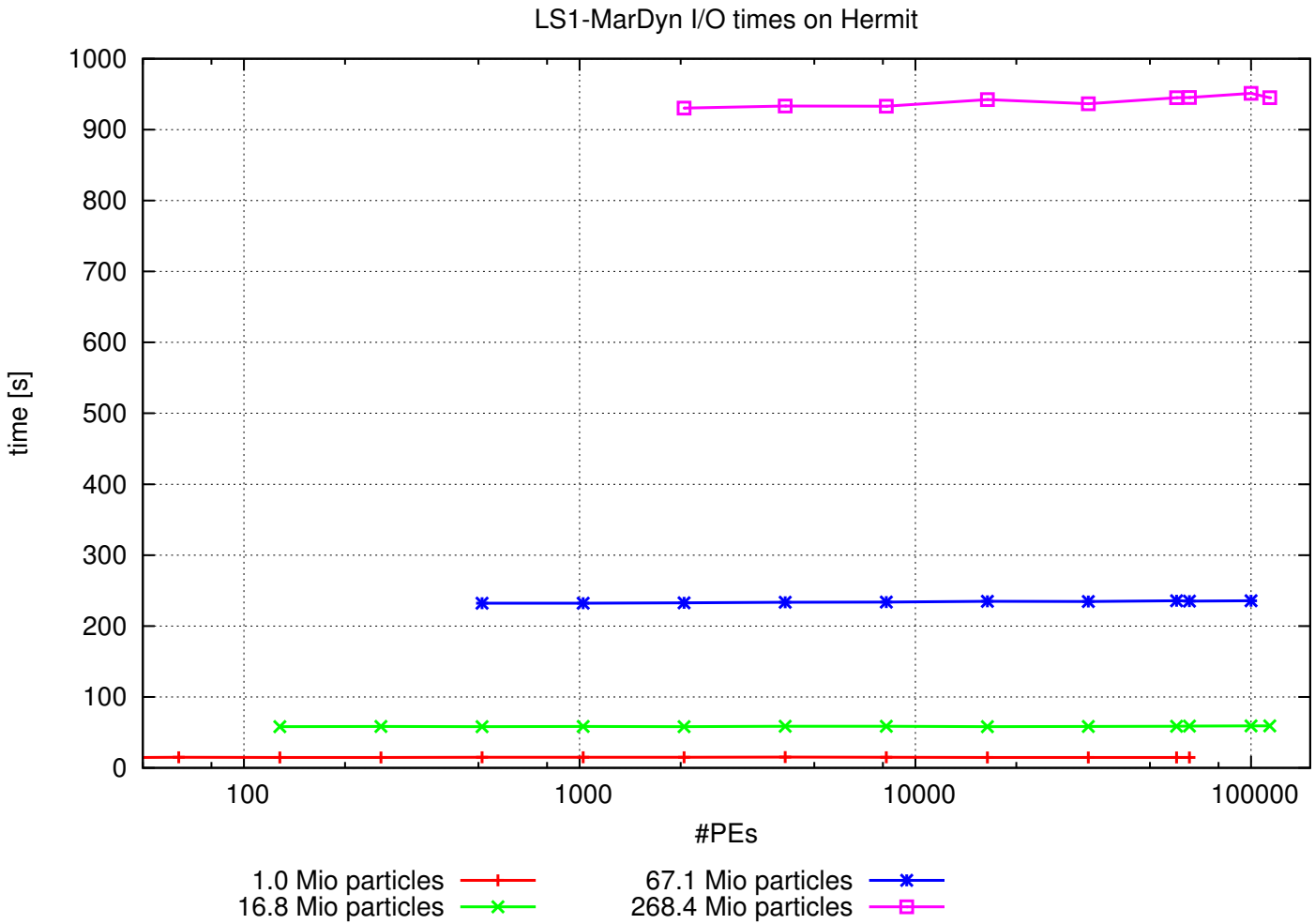
Problem with old version:

- I/O was done by every process
- all processes open the same file

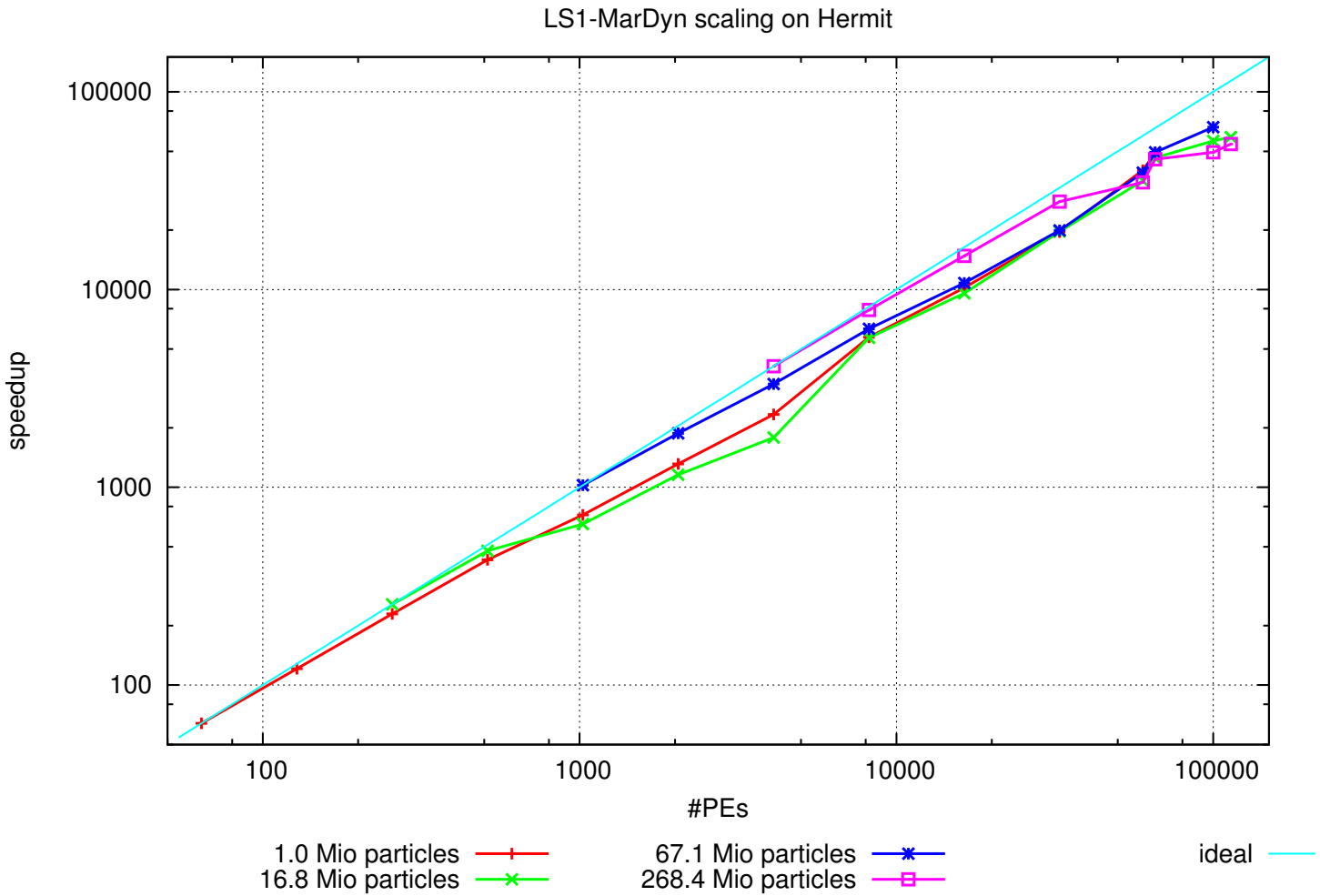
Improvement:

- Only master opens file for reading
- Master broadcasts data block wise to the other processes.

ls1-MarDyn I/O times on Hermit



ls1-MarDyn Scaling on Hermit



Conclusion

- Best compiler for sequential performance of ls1-MarDyn currently GNU 4.6
- Cache usage already good
- MPI neighbour communication good but can profit from rank reordering
- Using derived data types for MPI allreduce operations improves performance by 10% for 16.000 PEs and improves scalability.
- I/O implementation scales now up to 100000 cores using master & broadcast approach.

Questions?

Be welcome to ask questions!

End of

Performance evaluation and optimization of the ls1-MarDyn Molecular Dynamics code on the Cray XE6
Thank you for your attention!

Christoph Niethammer

CUG, May 1st, 2012, Stuttgart