Analyzing the I/O scalability of a parallel Particle-in-Cell code

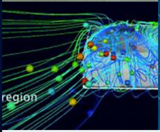
Sandra Mendez, Nicolay Hammer, Anupam Karmakar

Email: sandra.mendez@lrz.de

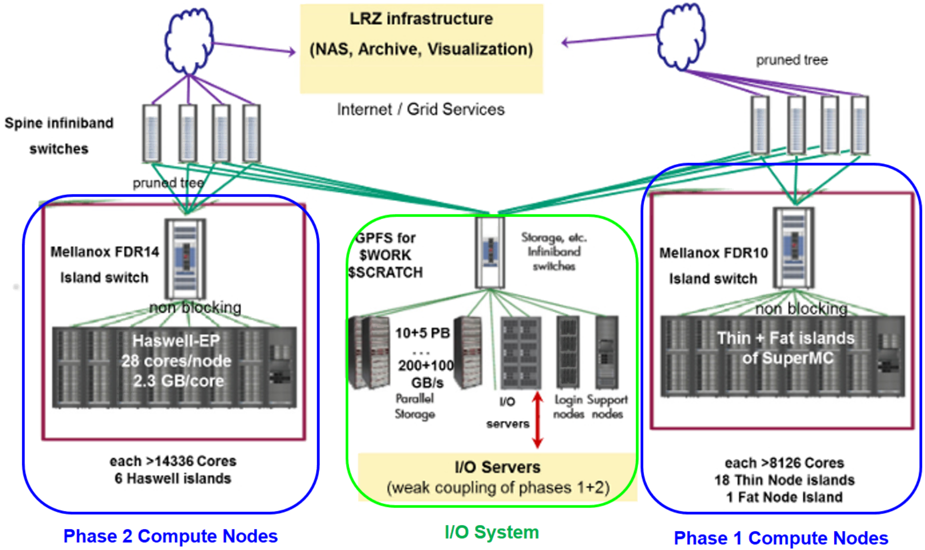
Outline

Introduction

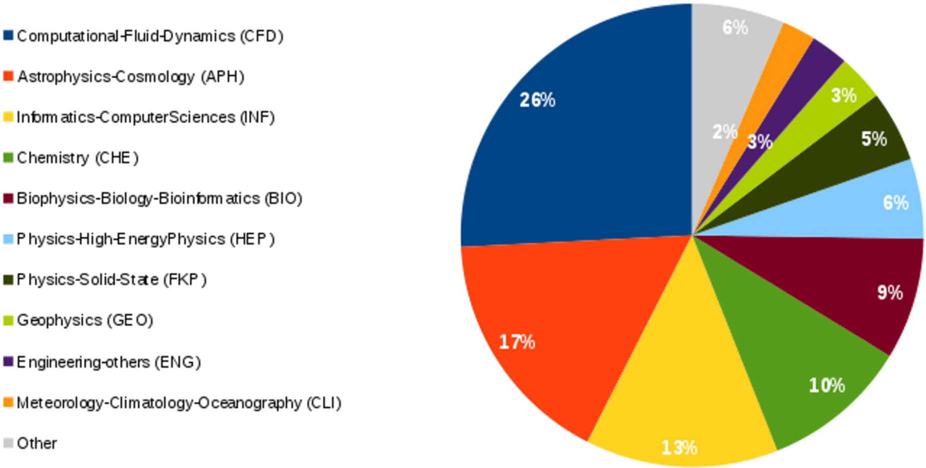
Member of the Gauss Centre for Supercomputing (GCS). Tier-0 centre for PRACE, the Partnership for Advanced Computing in Europe. 2012 SuperMUC Phase 1 and 2015 SuperMUC Phase 2. Total Peak Performance 6.4 PFlop/s.



SuperMUC Phase 1 and Phase 2



Projects by Research Area



Expert Support for Specific Domain

Application Labs

- Astrophysics and Plasma Physics (AstroLab)
- Biology and Life Sciences (BioLab)
- Computational Fluid Dynamics (CFDLab)
- Geosciences (GeoLab)

I/O Support

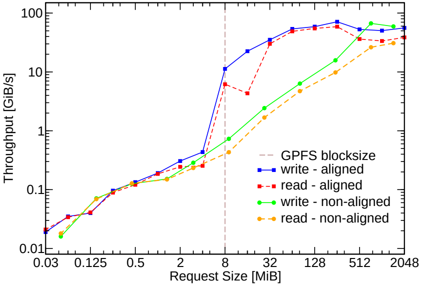
Ticket system provides support for technical problems with I/O implementations in scientific applications.
The *Application Labs* offer project based high level support for tuning, optimization and refactoring I/O implementations for user applications.

Technical Description

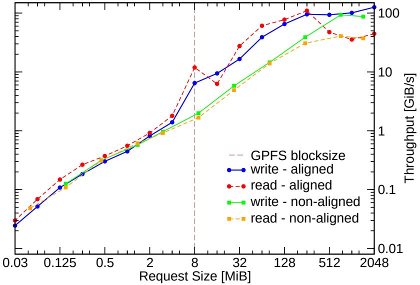
Compute System	Description	
Number of nodes	9216	
Nodes per Island	512	
Sockets per Node	2	
Cores per Node	16	
Memory per node (GByte)	32 (Usable 26)	
Communication Network	FDR10 IB	
Intra-Island topology	non-blocking tree	
Inter-Island topology	pruned tree 4:1	
I/O System	WORK	SCRATCH
Parallel Filesystem	IBM Spectrum Scale	
Network Shared Disk (NSD)	80 (DDN based)	16 (GSS based)
Stripe/Block Size	8MiB	8MiB
Filesystem Capacity	12 PiB	5.2 PiB
Max. I/O Performance		
Write(GiB/sec)	$\approx$ 180	$\approx$ 130
Read(GiB/sec)	$\approx$ 200	$\approx$ 150
Compute Node	$\approx$ 4.5 GiB/sec	

Throughput as a function of request sizes

SCRATCH



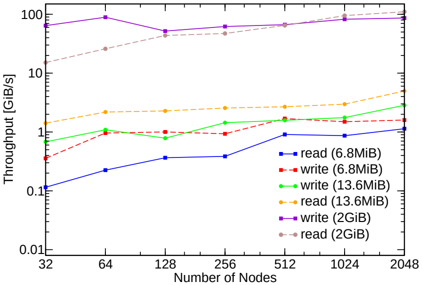
WORK



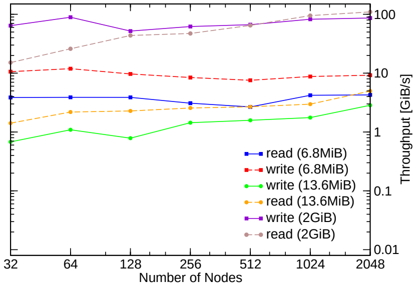
The benchmark was executed on 512 compute nodes of the SuperMUC Sandy Bridge system with 1 MPI task per node. There are two cases shown, one (blue/red) for `aligned` requests and a second one (yellow/green) for `non-aligned`.

Throughput as a function of the num. nodes

SCRATCH



WORK



The benchmark was executed on SuperMUC Sandybridge system partion with 2 MPI task per node. The plot shows the write and read performance for a request size of 6.8 MiB (blue/red), 13.6 MiB (green/yellow) and 2 GiB (purple/brown) per task.

A particle-in-cell code

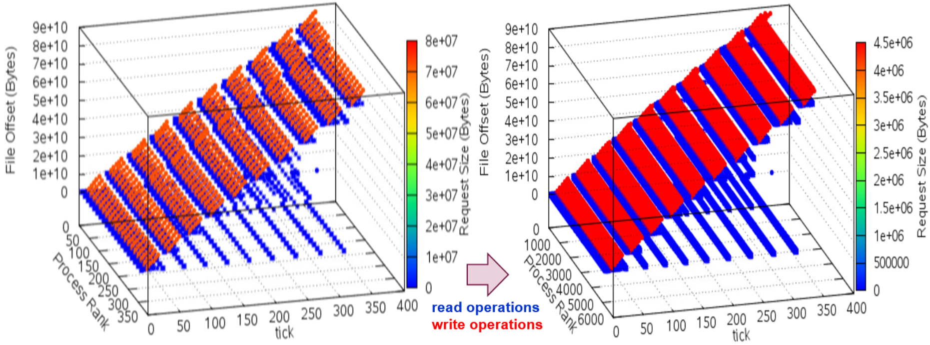
General characteristics of PiC codes:

- Domain decomposition
- Ghost cells
- Nearest neighbor communication
- Good scaling is expected

ACRONYM is well-tested and used on several different supercomputers with the HDF5 library providing output in the form of self-describing files.
Specific objective for ACRONYM:

- Optimize the output for maximum throughput on SuperMUC.
- Through comprehensive testing, we expect to determine the optimum number of output nodes.
- With IO being a major bottleneck for large scientific simulations this work will benefit other HPC projects, as well.

I/O Pattern Analysis



Global I/O pattern of the `Acronym's I/O Kernel` at MPI-IO level using 320 (left) and 5120 (right) MPI processes. *x*-axis corresponds to the MPI rank, *y*-axis represents calls to MPI-IO operations and *z*-axis represents the offset in the file for each MPI process. A heat map depicts the request size of each I/O operation.

Application parameters and I/O pattern

I/O Parameter	Values
Global Simulation Size	(x, y, z)
Local Simulation Size	$(x\_loc = x, y\_loc = y, z\_loc = \frac{z}{np})$
Compute Nodes	cn
Simulation step	st
fields	fi
writer processes	$wp = cn$
Data Size (Bytes)	ds
RequestSize(Bytes)	$rs = x\_loc \times y\_loc \times z\_loc \times ds$
FileSize(Bytes)	$fz = cn \times rs \times st \times fi$
Data per st (Bytes)	$D_{st} = cn \times rs \times fi$
Data per 1 cn per st (Bytes)	$D_{cnxst} = rs \times fi$

I/O Operation	Count
`MPI_File_open`	$st \times cn$
`MPI_File_write_at_all`	$st \times fi \times cn$
`MPI_File_write_at`	$(fi + 1) \times st$
`MPI_File_set_view`	$st \times fi \times cn \times 2$
`MPI_File_read_at`	$2 \times fi \times st \times cn + 23 \times cn$
`MPI_File_get_size`	st
`MPI_File_set_size`	$st \times cn$
`MPI_File_close`	$st \times cn$

lrz

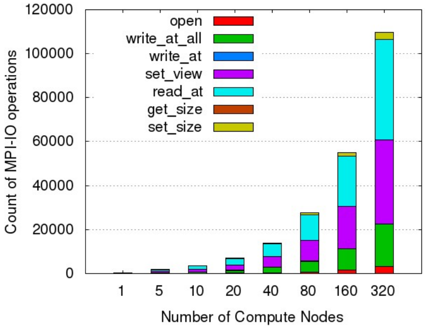
Parameters for experimentation

Global simulation size in cells, with 52 cells along the *x*- and *y*-direction and 66560 cells along the *z*-direction $(52, 52, 66560)$; 10 simulation steps (*st*) and 6 fields (*fi*). The size of data (*ds*) is 128 Bytes. By using these values we determine the *rs* and D_{cnxst} (Data per compute node per simulation step).

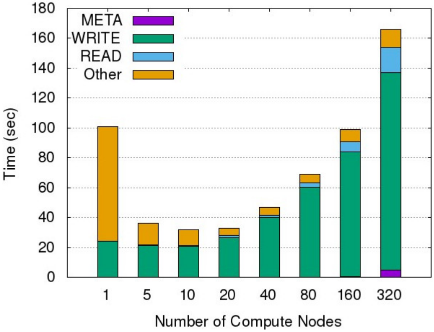
cn or writer	Number of Processes (*np*)	Local Simulation Size	Request Size *rs* (MiB)	Data per 1 *cn* per *st* D_{cnxst} (MiB)
1	16	(52,52,4160)	1373.13	8238.75
5	80	(52,52,832)	274.63	1647.75
10	160	(52,52,416)	137.31	823.88
20	320	(52,52,208)	68.66	411.94
40	640	(52,52,104)	34.33	205.97
80	1280	(52,52,52)	17.16	102.98
160	2560	(52,52,26)	8.58	51.49
320	5120	(52,52,13)	4.29	25.75

Evaluating the weight of I/O operations

Count of I/O Operations

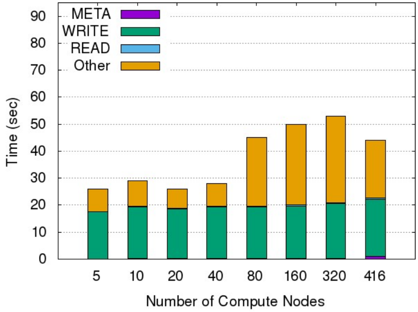


I/O Time per Operation Type

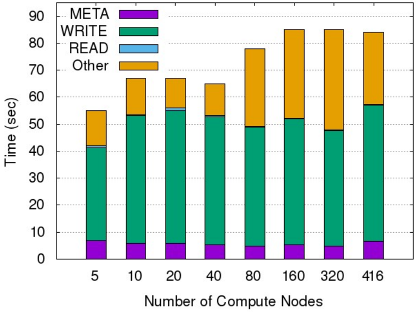


I/O Aggregation Evaluation

Normal I/O Aggregation

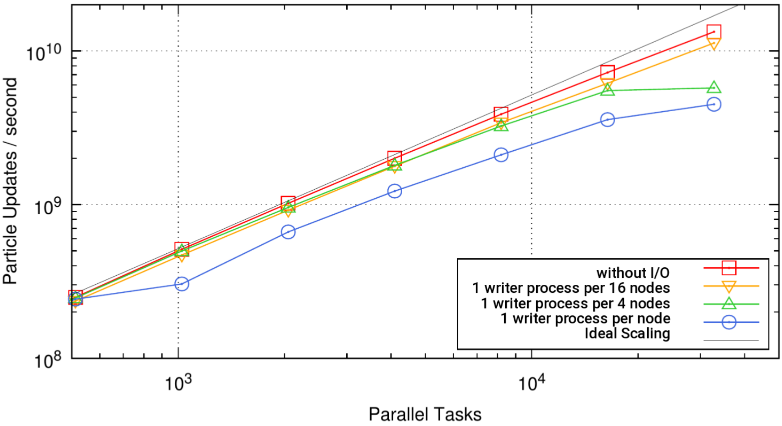


I/O Aggregation and Chunking



Experimentation

Weak scaling of the ACRONYM PiC-Code with and without I/O by using the optimized I/O implementation (plot provided by ACRONYM developer team)



Conclusions

- Selection of request size taking into account the simulation parameters and the I/O pattern.
- Characterization of the I/O system to explain the behavior of the original I/O implementation of ACRONYM. It can provide guidelines for other users of SuperMUC encountering problems with I/O scalability.
- A small number of computational ranks act as designated I/O agents that provides much better scaling even for simulations up to 32k cores. Results showed total time 4.5x faster than the original version for the best case.

Thank you for your attention!