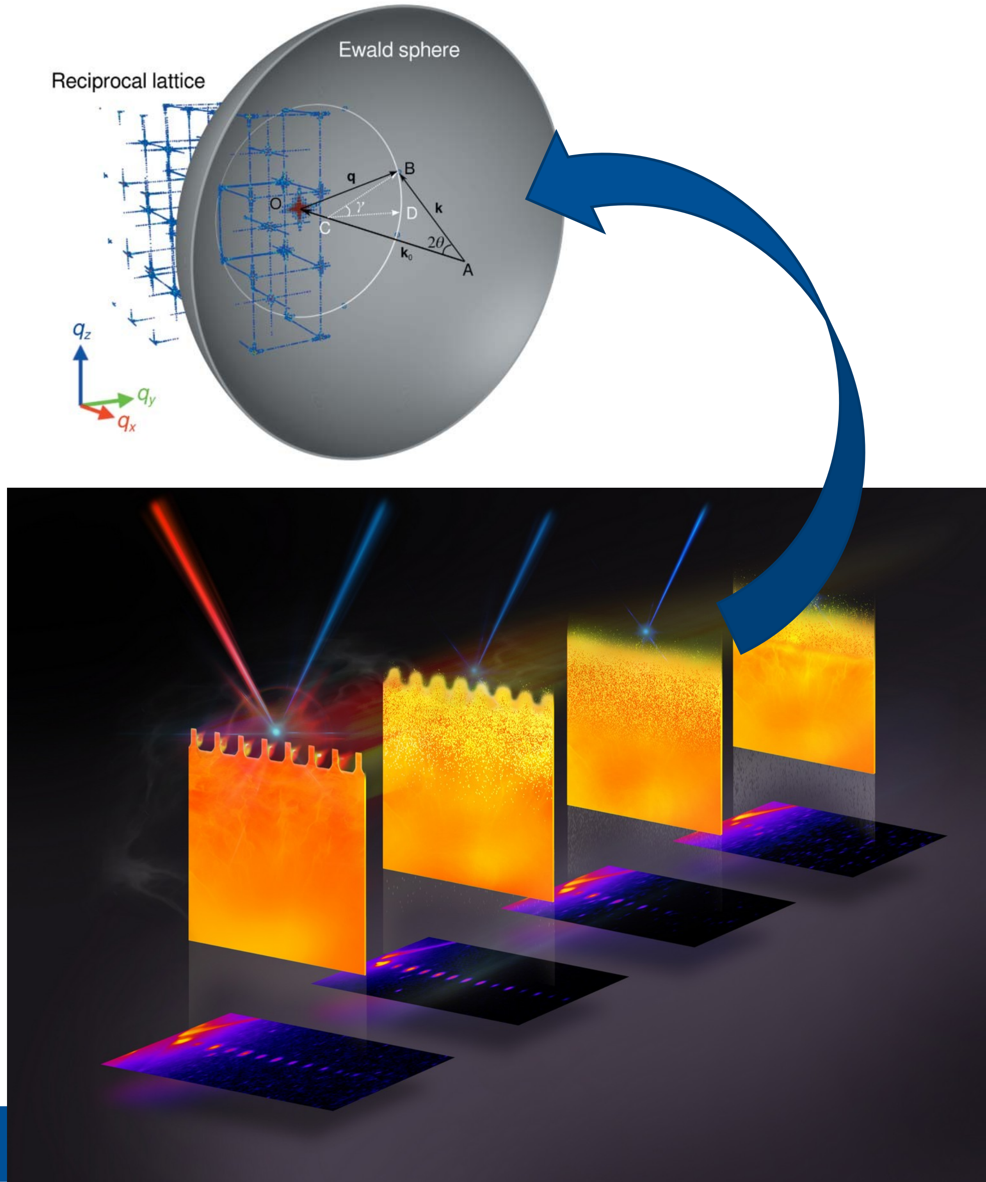
openPMD – Open and F.A.I.R. I/O for Particle-Mesh Data at the Exascale

Franz Poeschel[1,4], Juncheng E[5], William F. Godoy[3], Norbert Podhorszki[3], Scott Klasky[3], Greg Eisenhauer[6], Philip E. Davis[7], Lipeng Wan[3], Ana Gainaru[3], Junmin Gu[2], Fabian Koller[4], René Widera[4], Michael Bussmann[1,4] and Axel Huebl[2,4]

2022 SIAM Conference
on Parallel Processing for Scientific Computing

1) **Center for Advanced Systems Understanding (CASUS), D-02826 Görlitz, Germany**
2) **Lawrence Berkeley National Laboratory (LBNL), Berkeley 94720, California, USA**
3) **Oak Ridge National Laboratory (ORNL), Oak Ridge 37830, Tennessee, USA**
4) **Helmholtz-Zentrum Dresden-Rossendorf (HZDR), D-01328 Dresden, Germany**
5) **European XFEL GmbH (EU XFEL), D-22869 Schenefeld, Germany**
6) **Georgia Institute of Technology (Georgia Tech), Atlanta 30332, Georgia, USA**
7) **Rutgers University (Rutgers), New Brunswick 08901, New Jersey, USA**

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING



openPMD –
Open and F.A.I.R. I/O for Particle-Mesh Data at the Exascale

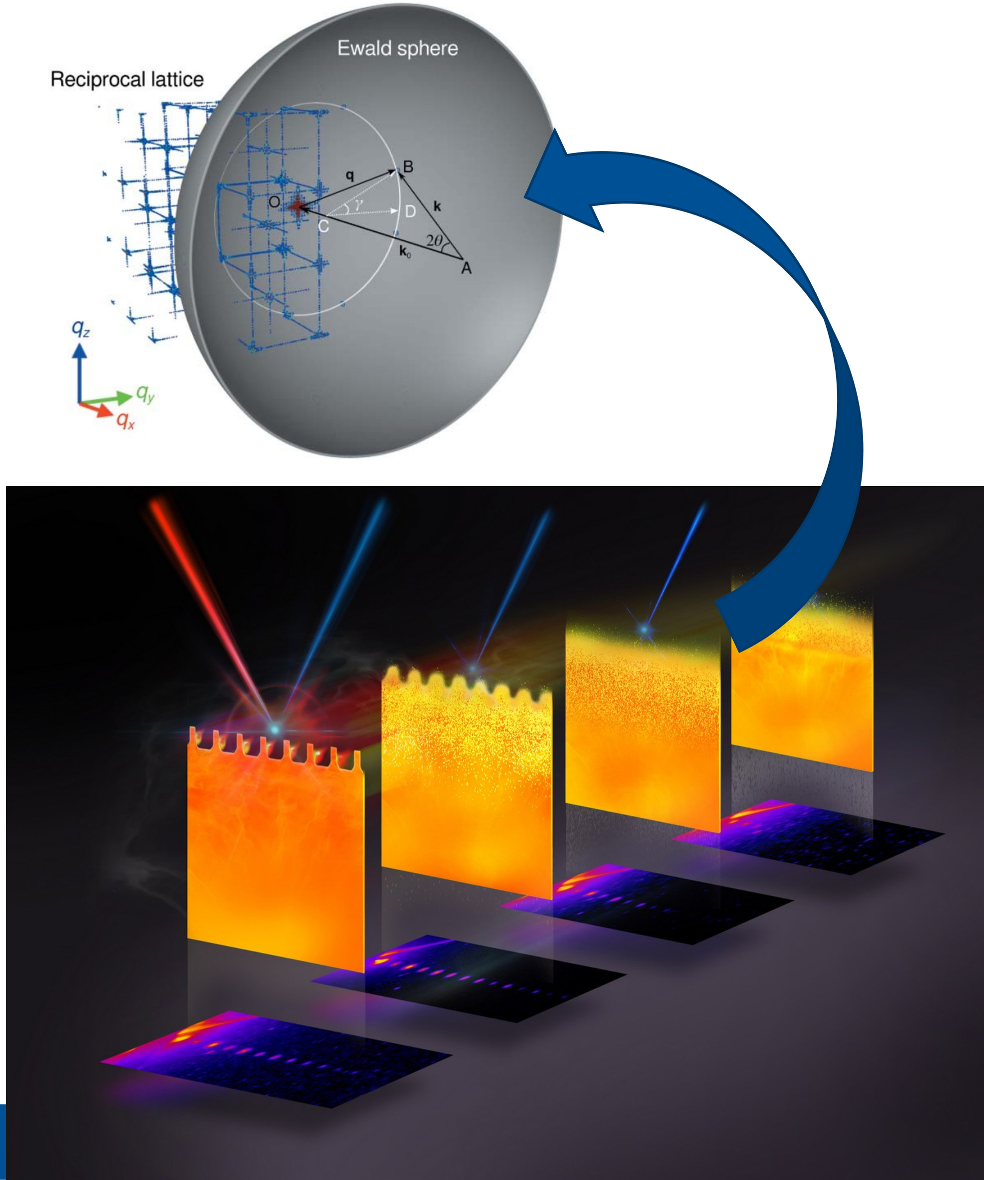
CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Acknowledgements
This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. Supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration). Supported by EC through Laserlab- Europe, H2020 EC-GA 871124. Supported by the Consortium for Advanced Modeling of Particles Accelerators (CAMPA), funded by the U.S. DOE Office of Science under Contract No. DE-AC02-05CH11231. This work was partially funded by the Center of Advanced Systems Understanding (CASUS), which is financed by Germany's Federal Ministry of Education and Research (BMBF) and by the Saxon Ministry for Science, Culture and Tourism (SMWK) with tax funds on the basis of the budget approved by the Saxon State Parliament.

Structure:

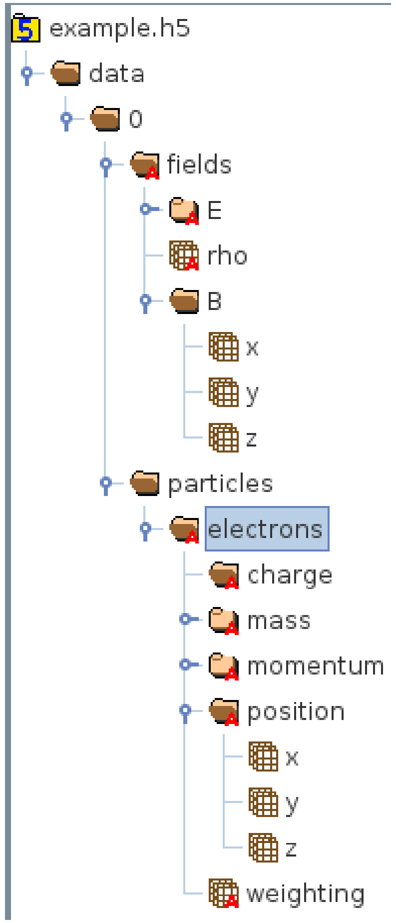
1) openPMD: Open and F.A.I.R. I/O
2) Benchmark: Asynchronous I/O
3) Benchmark:
   Loosely-coupled simulation pipeline

1) openPMD
open and F.A.I.R. I/O



openPMD: Open and F.A.I.R. I/O
for Particle-Mesh Data at the Exascale

openPMD – self-describing scientific data



Self-describing, **data format agnostic** standard for **frictionless exchange** of **particle-mesh data**

Flagship implementation: **openPMD-api**:

- API in C++ and Python (upcoming: Julia)
- Describe **particle-mesh data** in a unified way
- Flexibly store to / read from interchangeable backends:
  - ADIOS1/2
  - HDF5
  - JSON (serial only)

openPMD – a FAIR standard

Findable: Standardized metadata to identify the data producer

```
string    /author             attr   = "franz"
string    /software           attr   = "PIConGPU"
string    /softwareVersion    attr   = "0.5.0-dev"
```

Accessible: Open standard, implementable in various formats

*currently implemented,
but not limited to

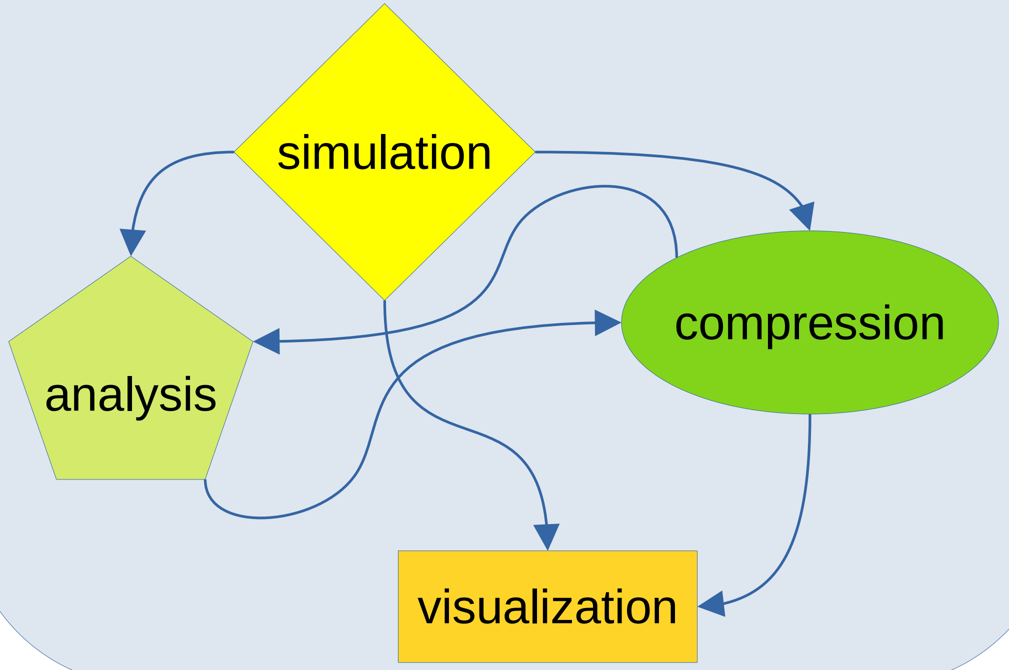
"The FAIR Guiding Principles for scientific data management and stewardship" (Mark D. Wilkinson et al.)

openPMD – a FAIR standard

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Interoperable:

Data exchange spans applications, platforms and teams



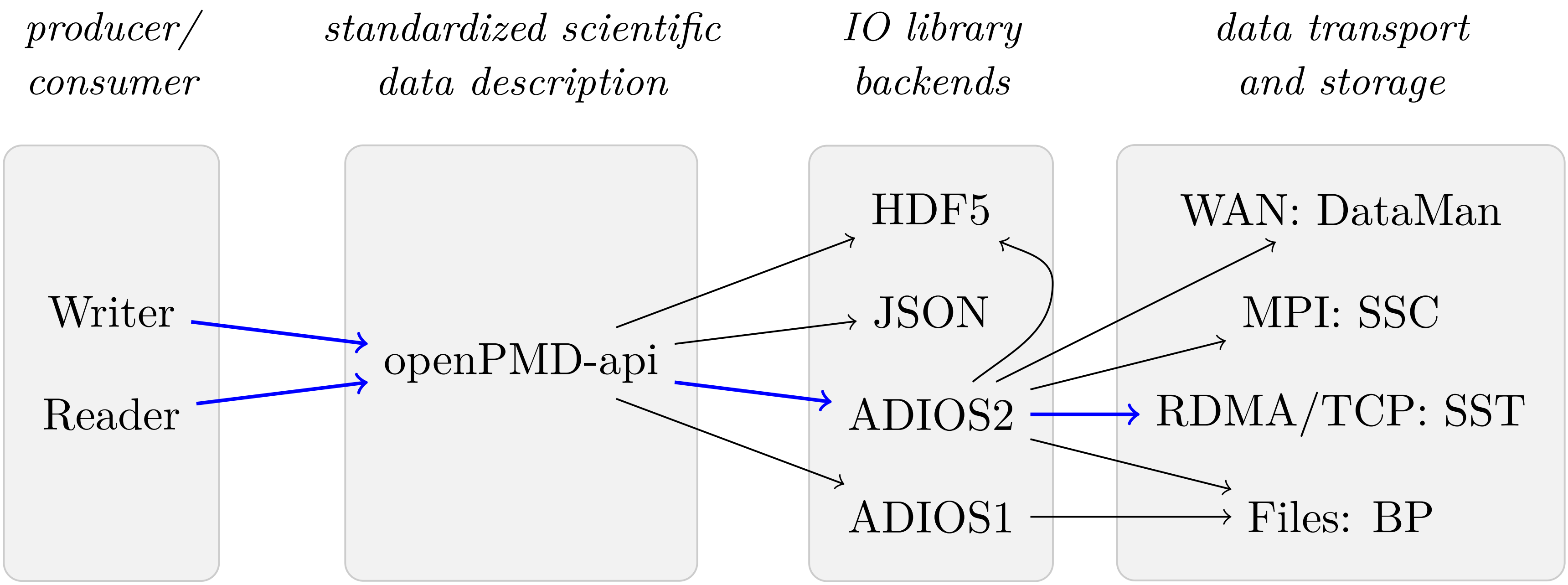
Reusable:

Rich and standardized description for physical quantities

Name	Value
axisLabels	[b'z' b'y' b'x']
dataOrder	b'C'
fieldSmoothing	b'none'
geometry	b'cartesian'
gridGlobalOffset	[0. 0. 0.]
gridSpacing	[4.252342 1.0630856 4.252342]
gridUnitSI	4.1671151662e-08
position	[0. 0. 0.]
timeOffset	0.0
unitDimension	[-3. 0. 1. 1. 0. 0. 0.]
unitSI	15399437.98944343

"The FAIR Guiding Principles for scientific data management and stewardship" (Mark D. Wilkinson et al.)

openPMD and ADIOS2 – open stack for scientific I/O



In blue: setup used for benchmarks in this talk

File-based I/O does not scale

The I/O bottleneck:

system	compute performance $[\mathrm{PFlop} \cdot \mathrm{s}^{-1}]$	parallel FS bandwidth $[\mathrm{TiByte} \cdot \mathrm{s}^{-1}]$	FS capacity $[\mathrm{PiByte}]$	example storage requirements $[\mathrm{PiByte}]$
Titan	27	1	27	5.3
Summit	200	2.5	250	21.1
Frontier	> 1500	$5 - 10$	$500 - 1000$	$80 - 100$

"example storage requirements": full-scale simulations, dump entire GPU memory to disk 50 times

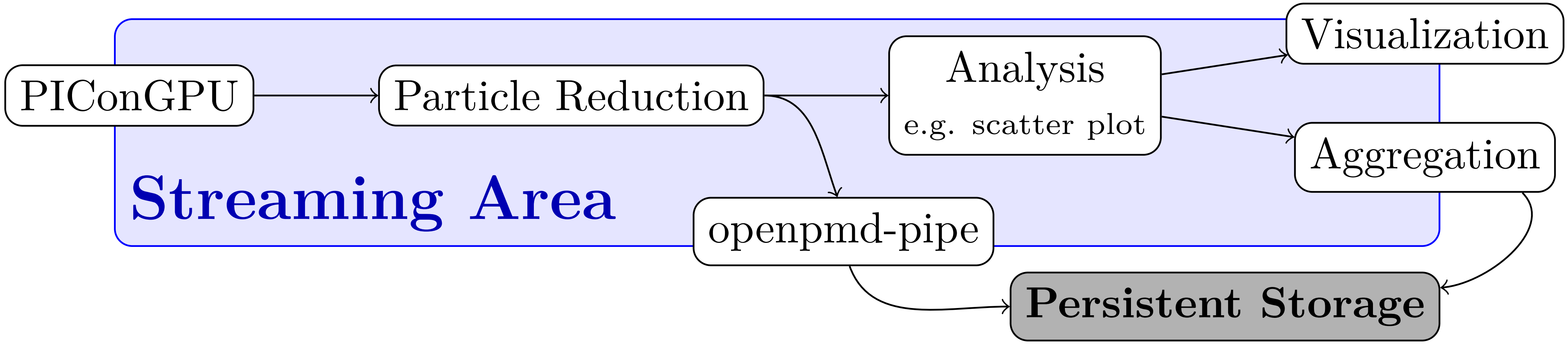
→ parallel bandwidth insufficient for HPC at full scale
→ filesystem capacity insufficient for HPC at full scale

Vision: Loosely coupled data processing pipeline

CASUS CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Loose coupling: Cooperate between independent applications, exchanging data

Streaming I/O between application bypasses PFS bottleneck:

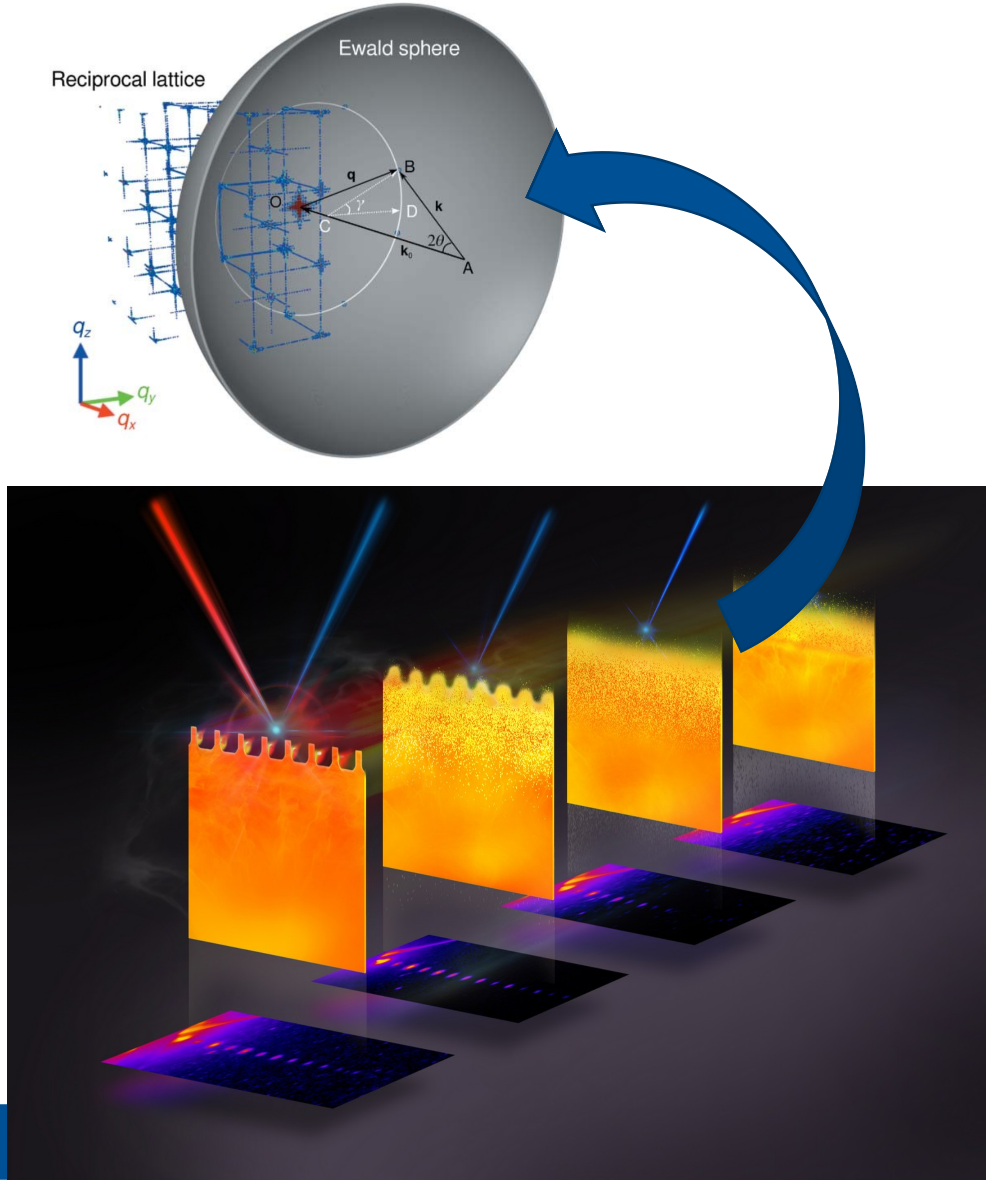


Focus of this talk:

Uniform scientific I/O communication layer between coupled applications

2) Benchmark: Asynchronous I/O

Benchmarks based on:
“Transitioning from file-based HPC workflows to streaming data pipelines with openPMD and ADIOS2” (F. Poeschel et al.)



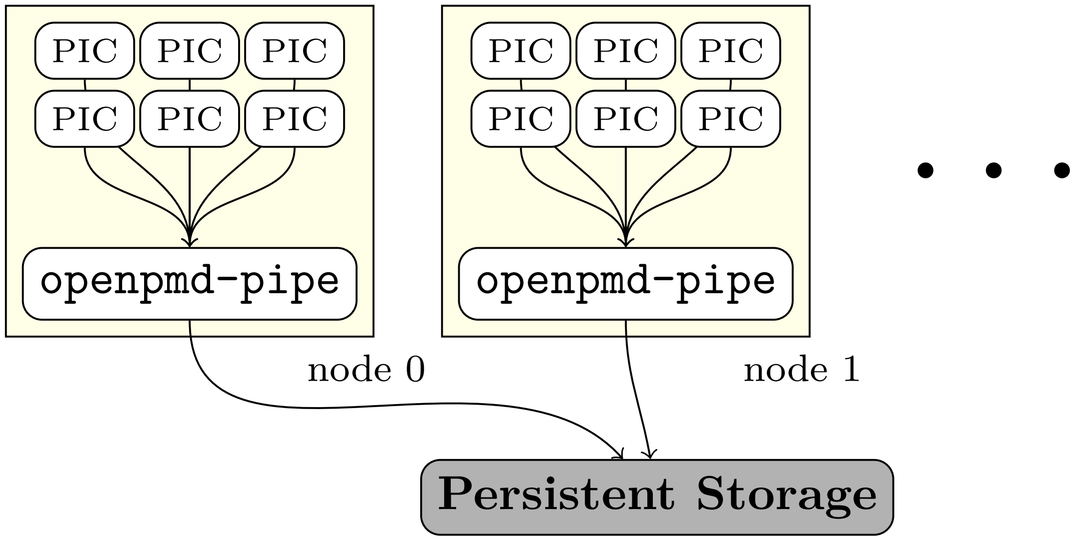
openPMD: Open and F.A.I.R. I/O
for Particle-Mesh Data at the Exascale

A simple use for streaming: Asynchronous I/O

A simple low-effort application for streaming:

- **Goal:** accelerate simulate-dump workflow
- **Assumption:** IO routines block other parts of the simulation
- **Solution:** Asynchronously launch a second application (`openpmd-pipe.py` – compare UNIX pipes)
  → Reads from stream, writes to disk
- **Effect:** Hides (not reduces!) disk IO times

No changes in the code required

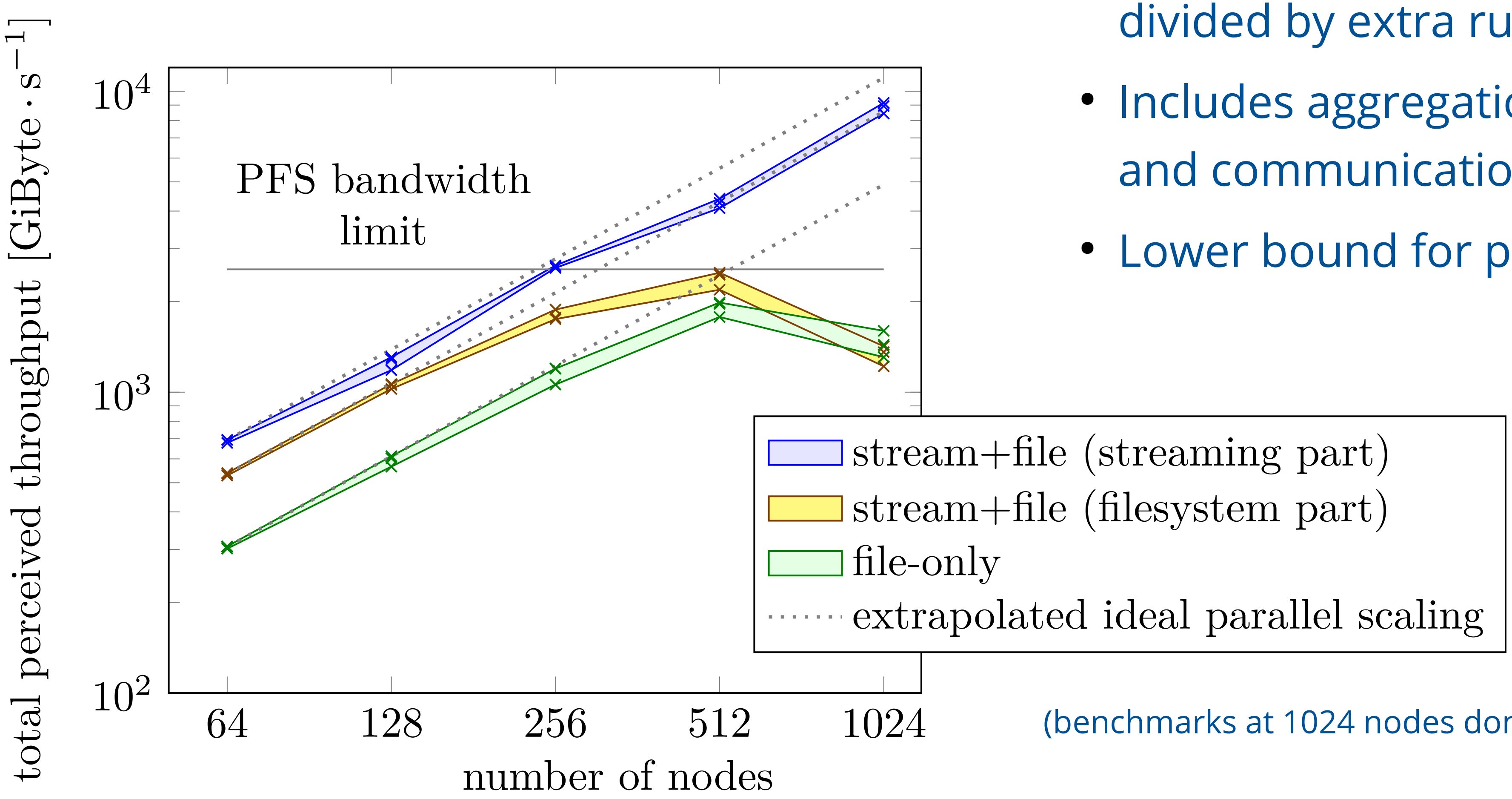


→ Compare this setup *(stream+file)* against regular file output *(file-only)*

```
> openpmd-pipe.py --infile stream.sst --outfile dump.bp
```

Axel Huebl. “PIConGPU: Predictive Simulations of Laser-Particle Accelera- tors with Manycore Hardware”. PhD thesis. Technische Universität Dresden, July 2019.

Streaming throughput stands out at high scale

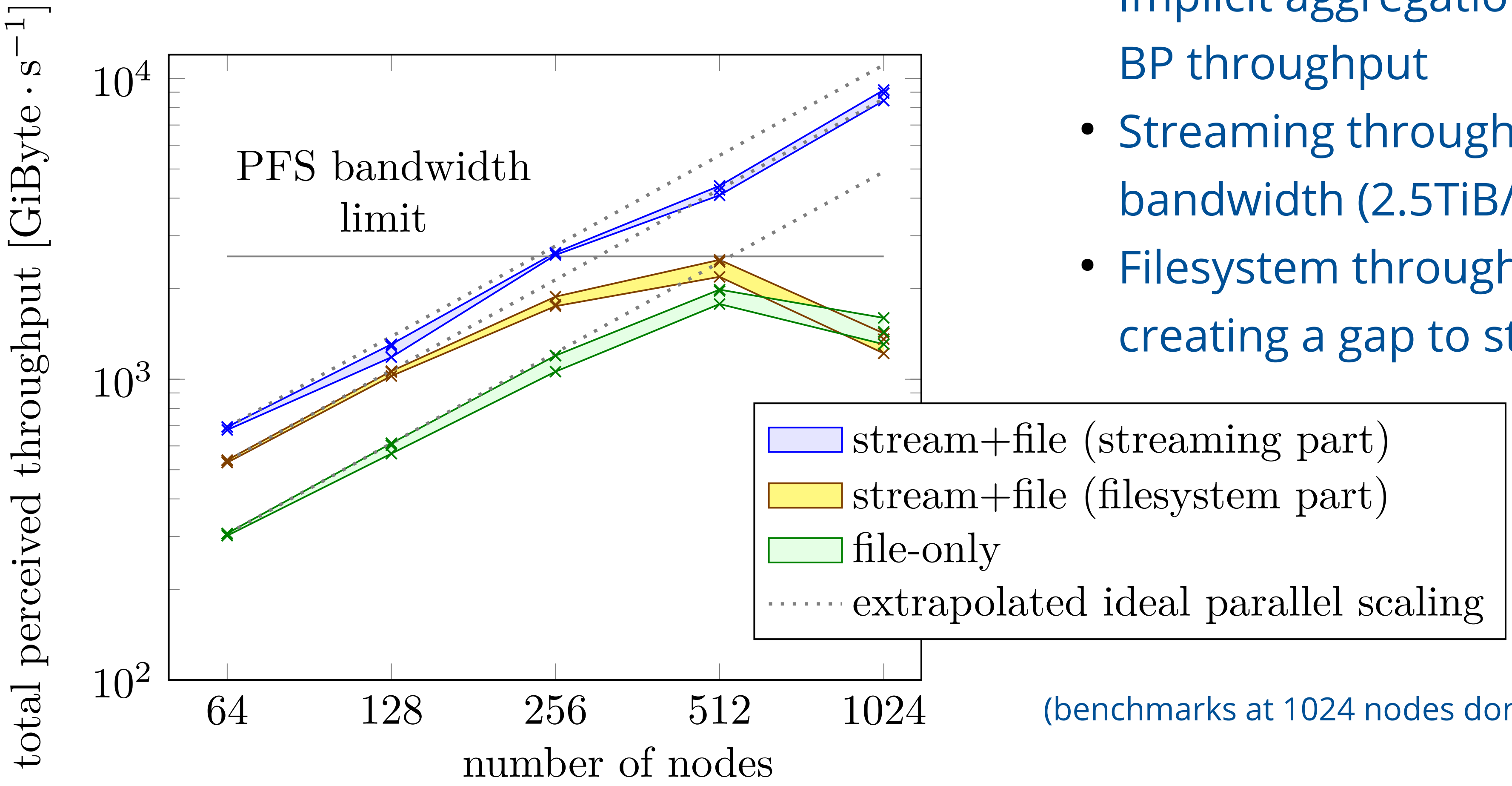


Perceived throughput:

- Defined as data written divided by extra runtime over no I/O
- Includes aggregation and communication overhead
- Lower bound for precise throughput

(benchmarks at 1024 nodes done after Summit system upgrade)

Streaming throughput stands out at high scale



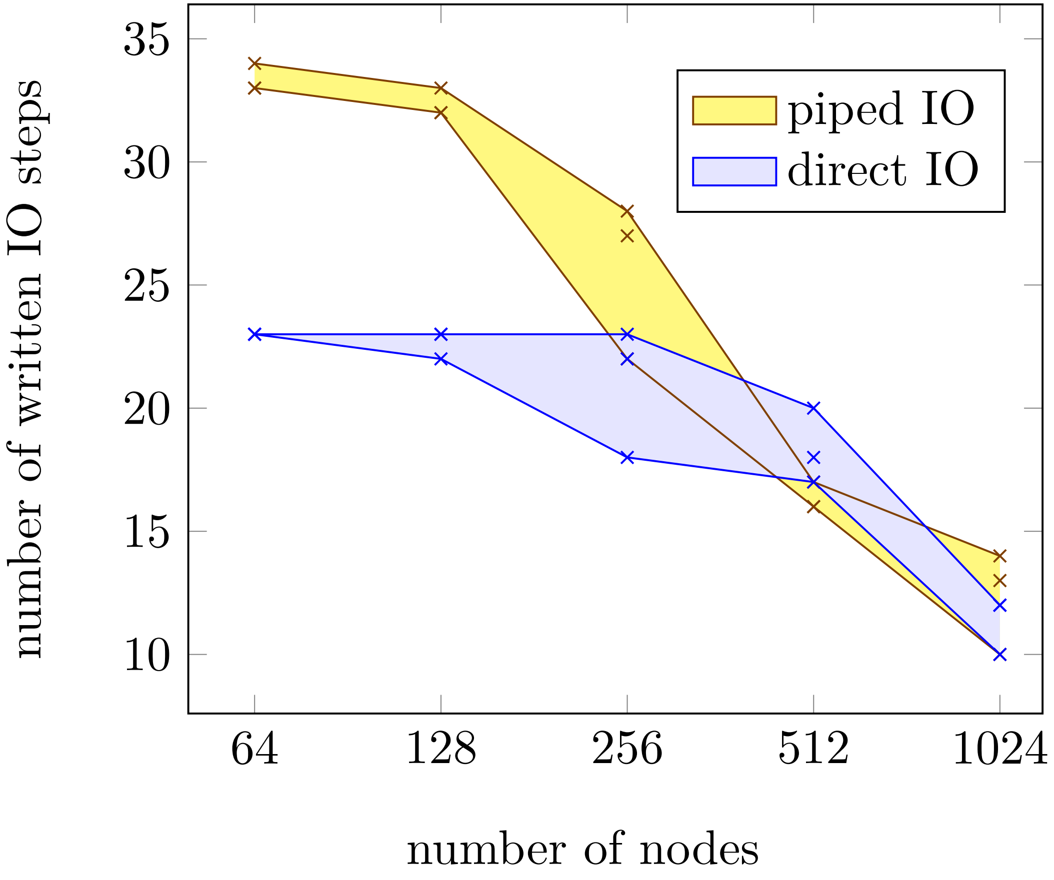
Evaluation:

- Overall reasonable scaling
- Implicit aggregation increases perceived BP throughput
- Streaming throughput exceeds PFS bandwidth (2.5TiB/s)
- Filesystem throughput limited by PFS, creating a gap to streaming throughput

(benchmarks at 1024 nodes done after Summit system upgrade)

Asynchronous I/O most helpful at lower scale

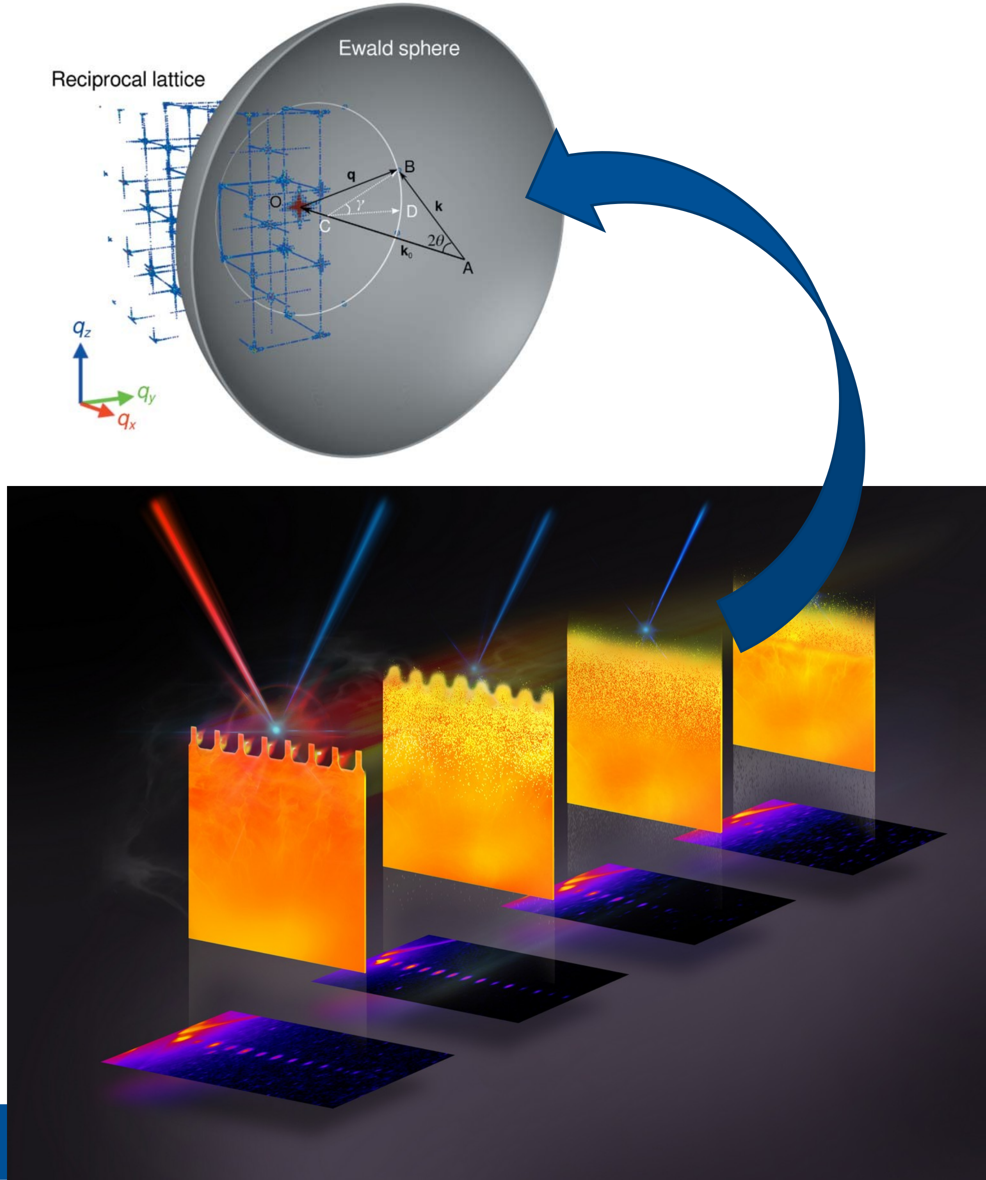
Number of written IO steps in 15 minutes:



Takeaway:

- At higher scale the PFS performance dominates
- For higher scale:
  Need something else
  → next setup

3) Benchmark: Loosely-coupled simulation pipeline

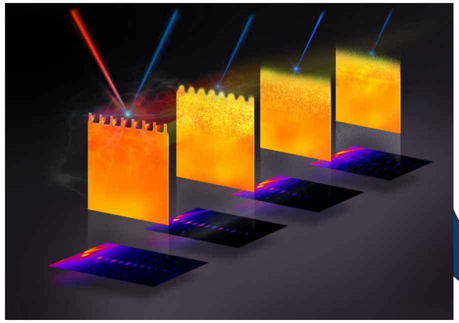


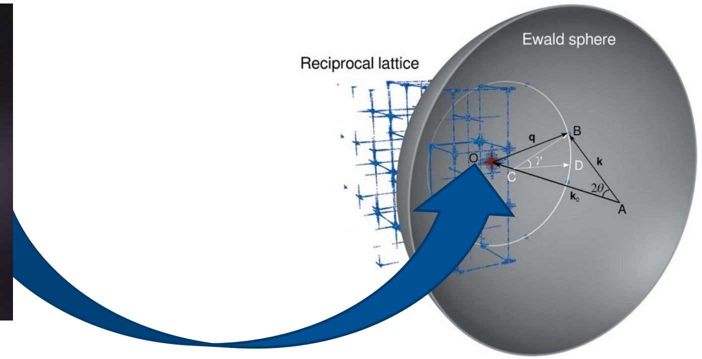
Benchmarks based on:
“Transitioning from file-based HPC workflows to streaming data pipelines with openPMD and ADIOS2” (F. Poeschel et al.)

openPMD: Open and F.A.I.R. I/O
for Particle-Mesh Data at the Exascale

Circumvent I/O bottleneck by loose coupling

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

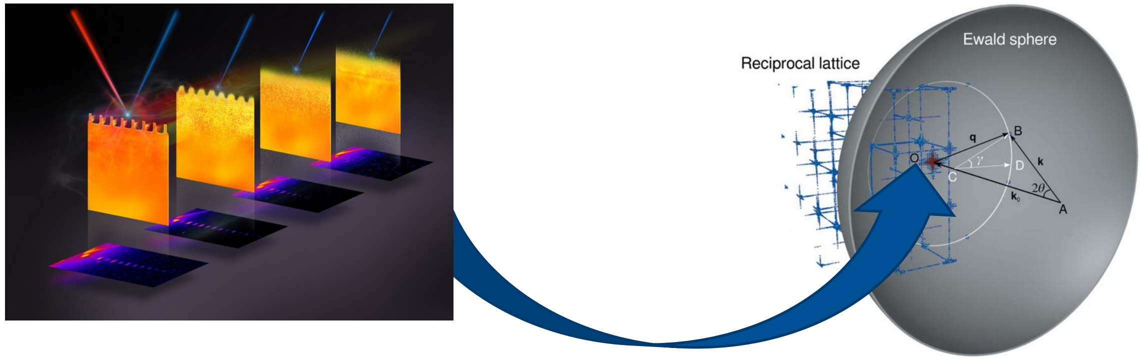




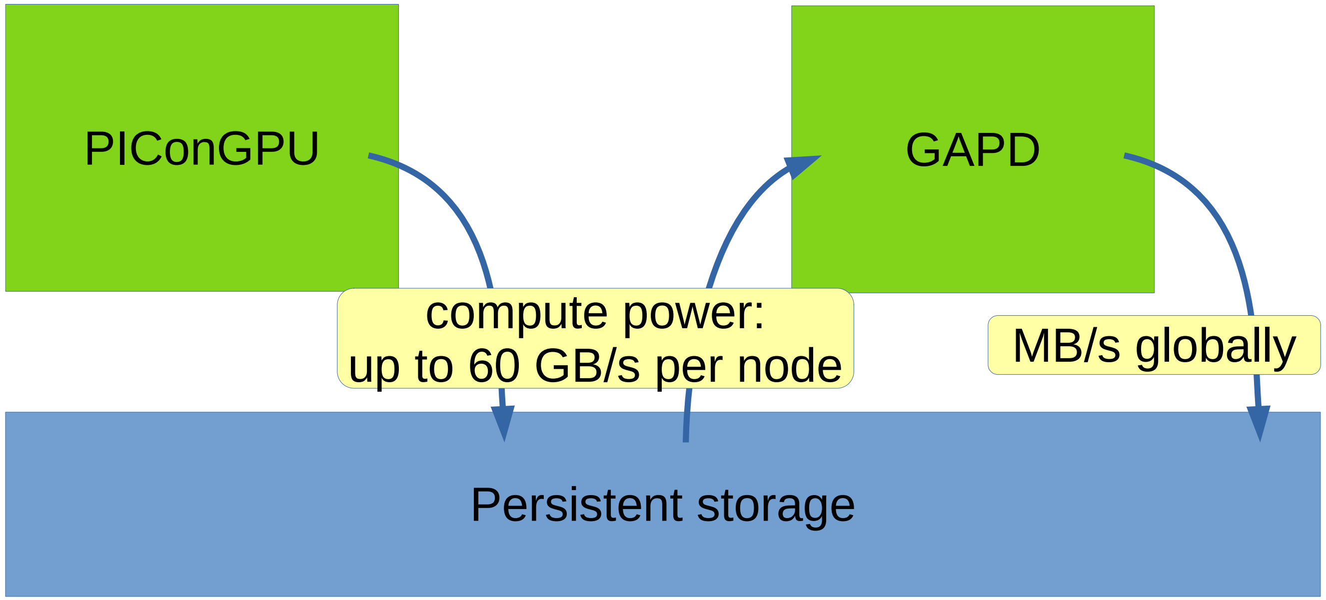
- Simulation pipeline: PIConGPU → GAPD
  GAPD: Scattering analysis
- Data description in openPMD and ADIOS is **independent of implementation**
- Use legacy, file-IO based implementations, but toggle a **streaming-aware backend**
- Only **store the final result persistently**

J. C. E, L. Wang, S. Chen, Y. Y. Zhang and S. N. Luo. “GAPD: a GPU- accelerated atom-based polychromatic diffraction simulation code”. In: Journal of Synchrotron Radiation 25.2 (Mar. 2018), pp. 604–611.

Circumvent I/O bottleneck by loose coupling

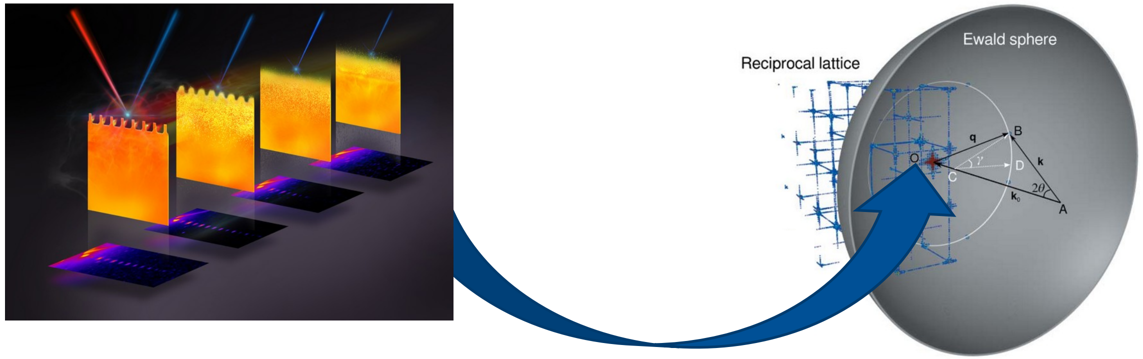


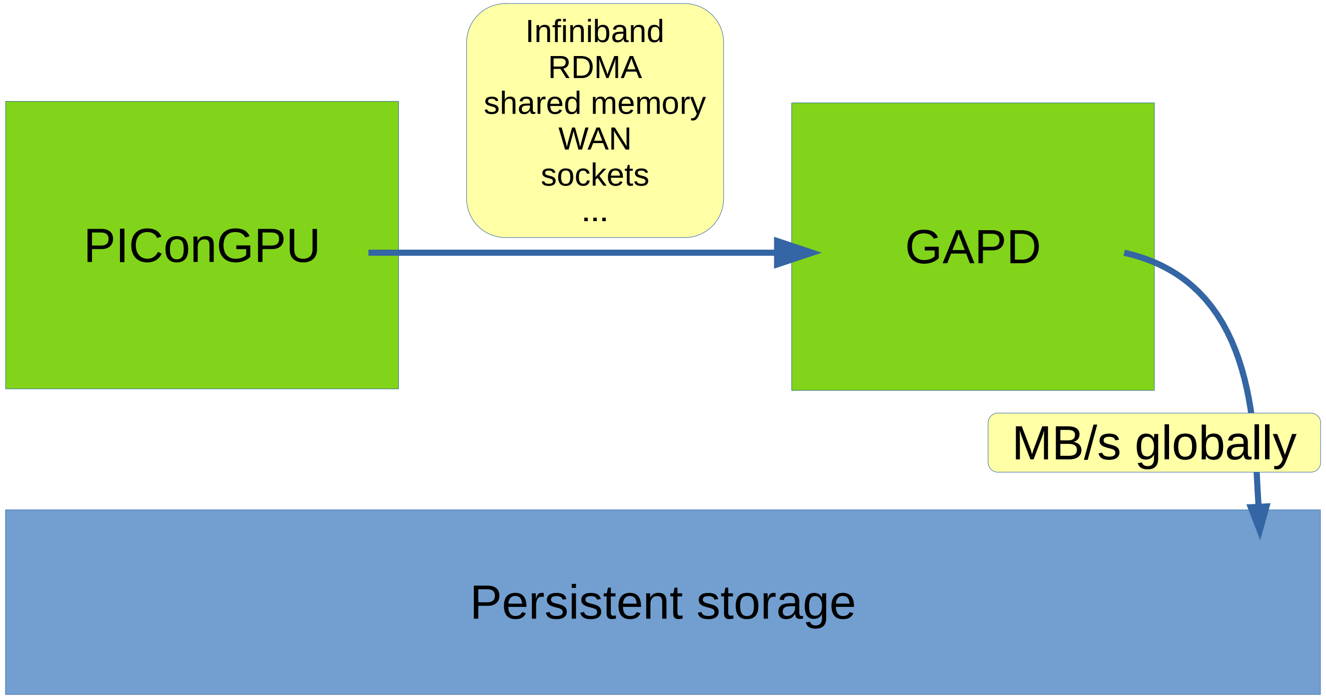
- Simulation pipeline: PIConGPU → GAPD
  GAPD: Scattering analysis
- Data description in openPMD and ADIOS is **independent of implementation**
- Use legacy, file-IO based implementations, but toggle a **streaming-aware backend**
- Only **store the final result persistently**



J. C. E, L. Wang, S. Chen, Y. Y. Zhang and S. N. Luo. “GAPD: a GPU- accelerated atom-based polychromatic diffraction simulation code”. In: Journal of Synchrotron Radiation 25.2 (Mar. 2018), pp. 604–611.

Circumvent I/O bottleneck by loose coupling



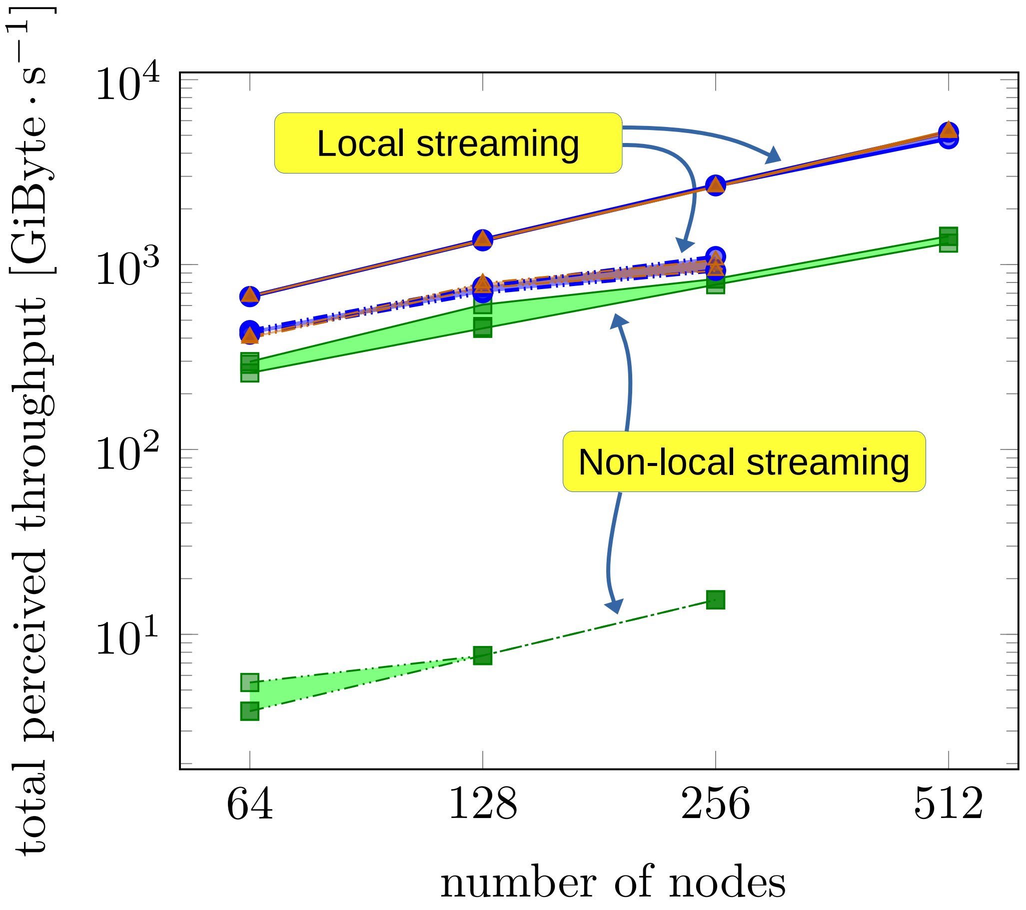


- Simulation pipeline: PIConGPU → GAPD
  GAPD: Scattering analysis
- Data description in openPMD and ADIOS is **independent of implementation**
- Use legacy, file-IO based implementations, but toggle a **streaming-aware backend**
- Only **store the final result persistently**

J. C. E, L. Wang, S. Chen, Y. Y. Zhang and S. N. Luo. “GAPD: a GPU- accelerated atom-based polychromatic diffraction simulation code”. In: Journal of Synchrotron Radiation 25.2 (Mar. 2018), pp. 604–611.

For good throughput: Local streaming patterns, Infiniband/RDMA

CASUS CENTER FOR ADVANCED SYSTEMS UNDERSTANDING



Local streaming:

Distribute data chunks only within a node (alternatively: to neighboring nodes)

Non-local streaming:

Distribute data chunks globally, optimize for balance and alignment

Straight lines:

Infiniband/RDMA

Dashed lines:

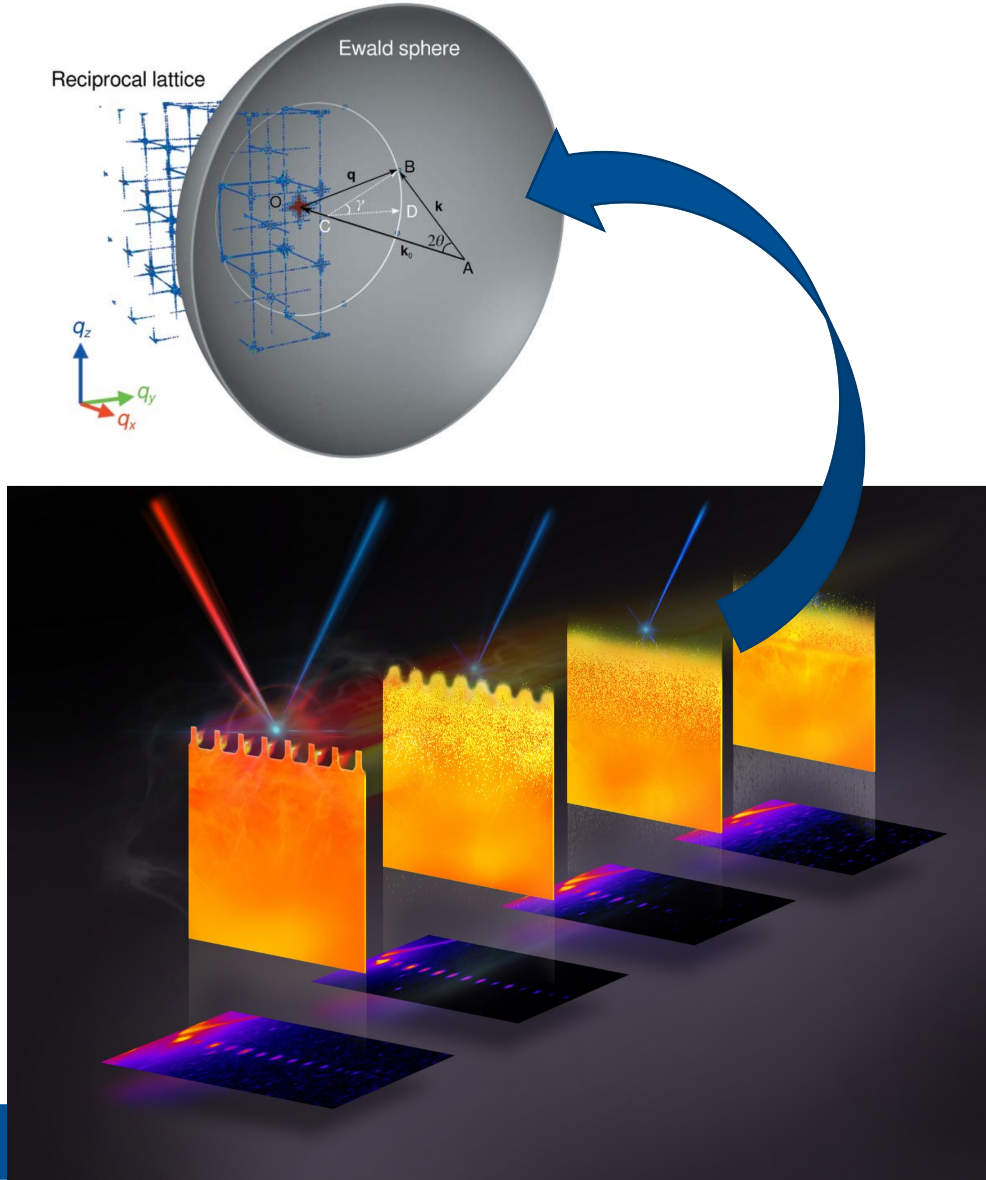
TCP/sockets

Takeaway:

- RDMA necessary for HPC
- Reasonable scaling with RDMA
- SST: number of communication partners for each single instance decisive
- Network topology has an impact

Conclusion

- **openPMD combines scientific F.A.I.R. compliance with performance at the Exascale**
- **Transition path:** file-based to streaming-based scientific data processing pipelines
- **Asynchronous I/O** through loose coupling (stream+file)
- **RDMA throughput** at 1024 nodes: more than 3 times PFS bandwidth
- Simulation → Analysis: **Bypass the PFS**



Outlook

- Larger loosely coupled pipelines
- Use streaming for surrogate modeling of simulations: Much more dynamic I/O patterns
- Data distribution patterns

Conclusion

- **openPMD combines scientific F.A.I.R. compliance with performance at the Exascale**
- **Transition path:** file-based to streaming-based scientific data processing pipelines
- **Asynchronous I/O** through loose coupling (stream+file)
- **RDMA throughput** at 1024 nodes: more than 3 times PFS bandwidth
- Simulation → Analysis: **Bypass the PFS**

Outlook

- Larger loosely coupled pipelines
- Use streaming for surrogate modeling of simulations: Much more dynamic I/O patterns
- Data distribution patterns

https://github.com/openPMD

Contact:

- f.poeschel@hzdr.de
- axelhuebl@lbl.gov

Acknowledgements

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE- AC05-00OR22725. Supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration). Supported by EC through Laserlab- Europe, H2020 EC-GA 871124. Supported by the Consortium for Advanced Modeling of Particles Accelerators (CAMPA), funded by the U.S. DOE Office of Science under Contract No. DE-AC02-05CH11231. This work was partially funded by the Center of Advanced Systems Understanding (CASUS), which is financed by Germany's Federal Ministry of Education and Research (BMBF) and by the Saxon Ministry for Science, Culture and Tourism (SMWK) with tax funds on the basis of the budget approved by the Saxon State Parliament.

21