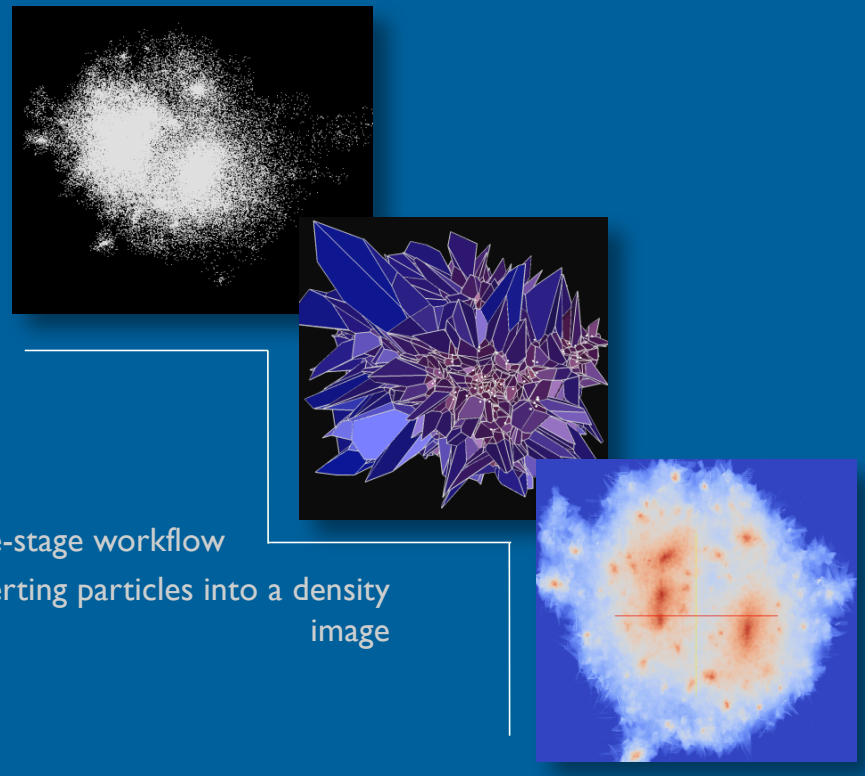
Argonne NATIONAL LABORATORY

High Performance Data Assimilation: A Computer Science Perspective of Parallel Data Coupling



"Data movement, rather than computational processing, will be the constrained resource at exascale."

– Dongarra et al. 2011

Three-stage workflow converting particles into a density image

RIKEN International Symposium
on Data Assimilation
February 28, 2017

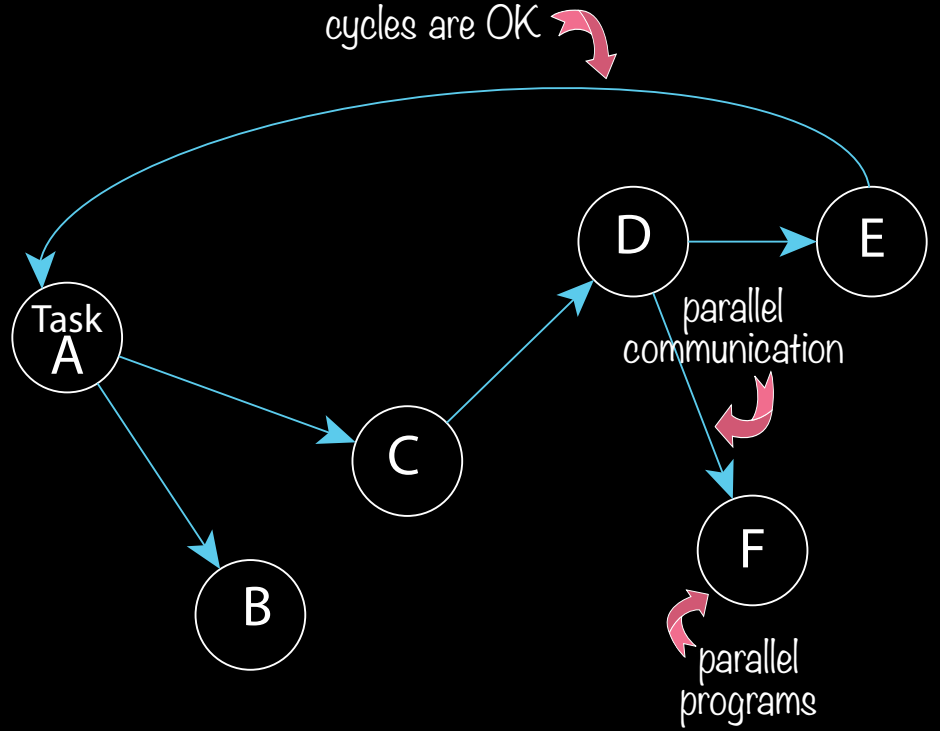
Tom Peterka
tpeterka@mcs.anl.gov
Mathematics and Computer Science Division

A Much Shorter Title

Dataflows for *workflows*

What's a Workflow?

- A directed graph of tasks and communication between them
- Graph nodes are the tasks
- Graph links are the communication

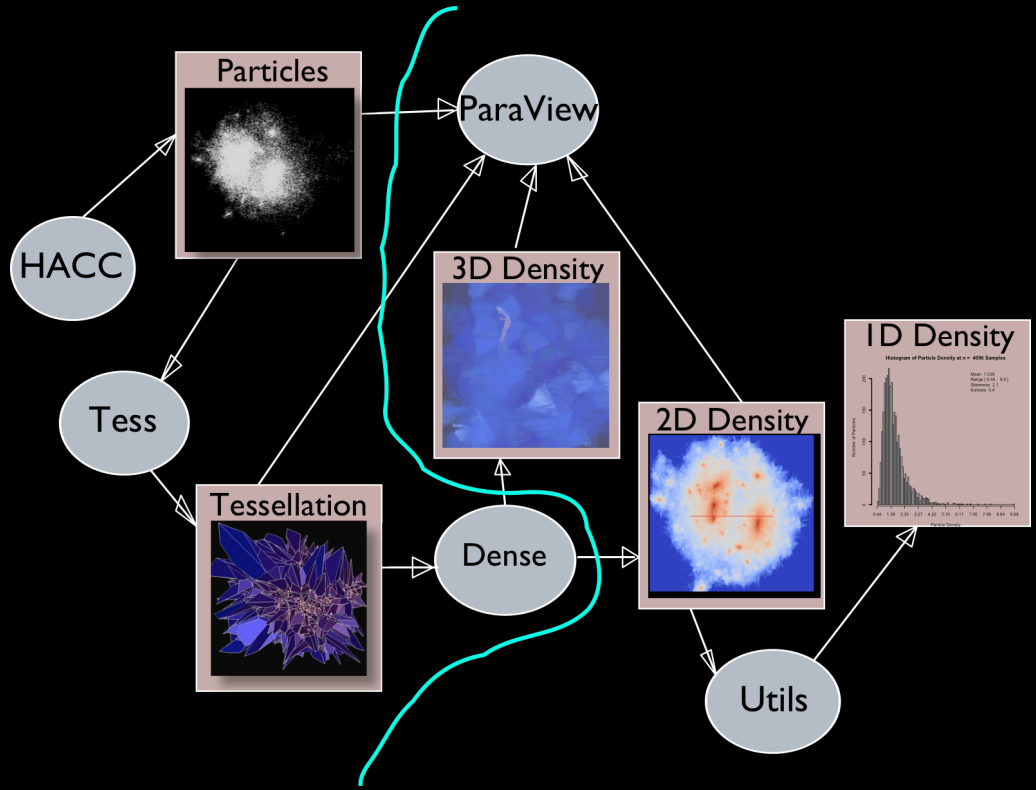


Footnotes

- Notice the graph does not have to be acyclic (digraph, not DAG)
- Think of "large tasks" (programs), not "small tasks" (threads)
- Nodes and links are parallel (parallel programs and parallel communication)

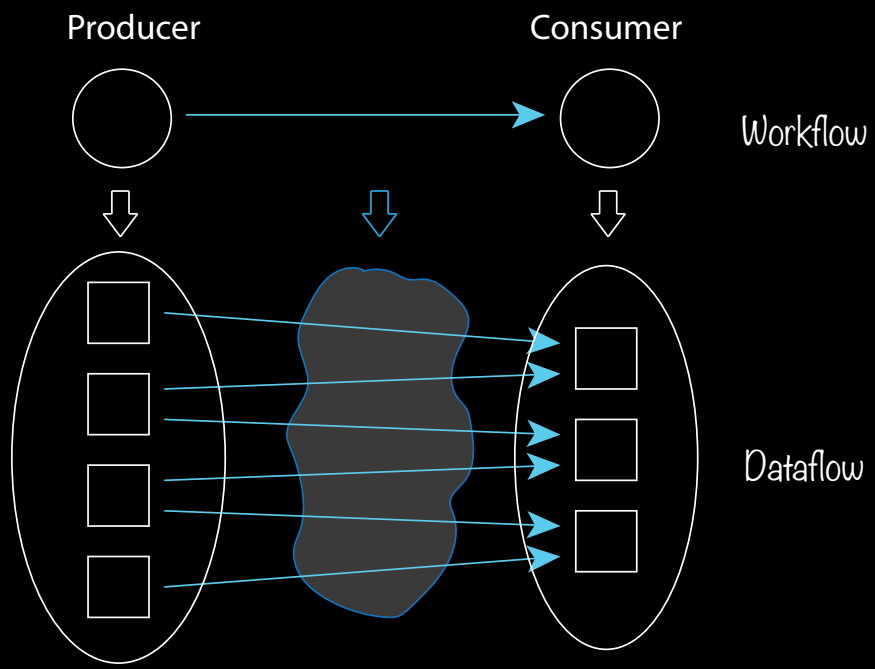
Simple In Situ Workflow Example
Analysis of Cosmology Simulations

- Just one small part of the complete cosmology workflow
- Converts dark matter particles to an unstructured mesh
- Converts an unstructured mesh to a regular grid
- Computes statistics over the grid and visualizes the results



What is Dataflow?

- For a pair of nodes connected by a directed edge (link) in the workflow graph,
- Dataflow is communication over the links in a workflow
- Workflow is in terms of tasks; dataflow is in terms of ranks



Footnotes

- For any 2 nodes + 1 (directed) link, call the starting node (wrt link direction) the *producer* and the finishing node the *consumer*
- Decompose any complex graph into a set of producer-link-consumer groups

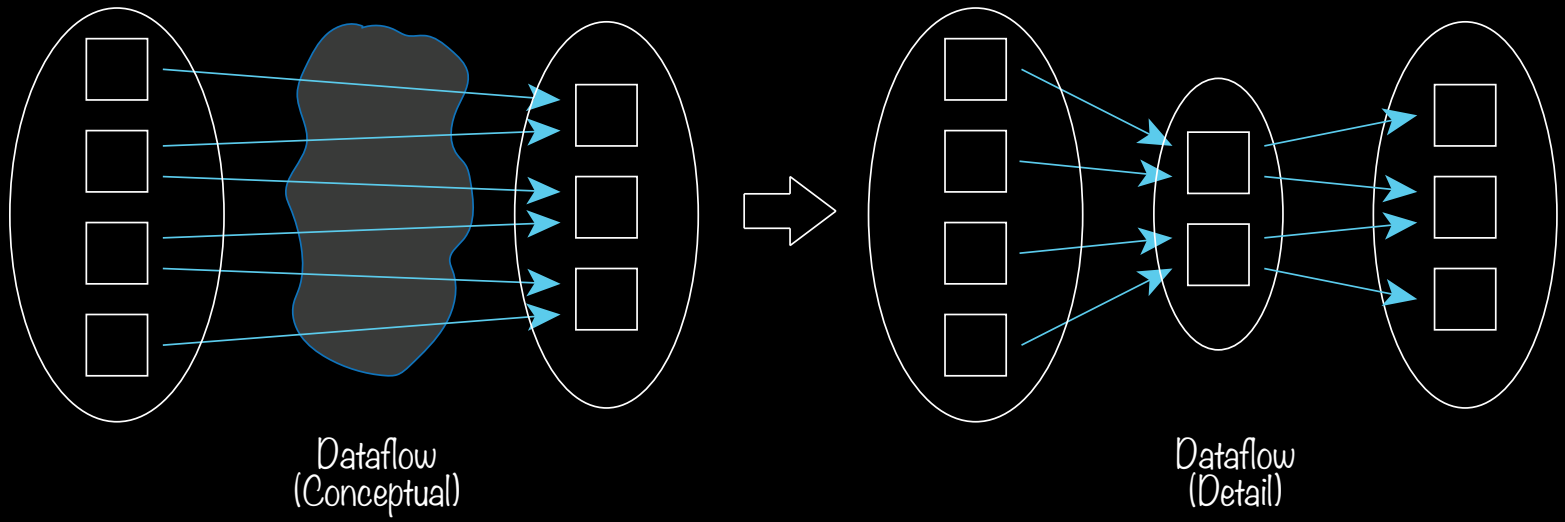
Dataflows in Detail

Challenges

- Parallel nodes and links
- Heterogeneous data models
- Disparate resource requirements
- Task placement: time and space division
- Varying flow rates
- Errors: hard and soft
- Programming model: API, usability

Links are Parallel

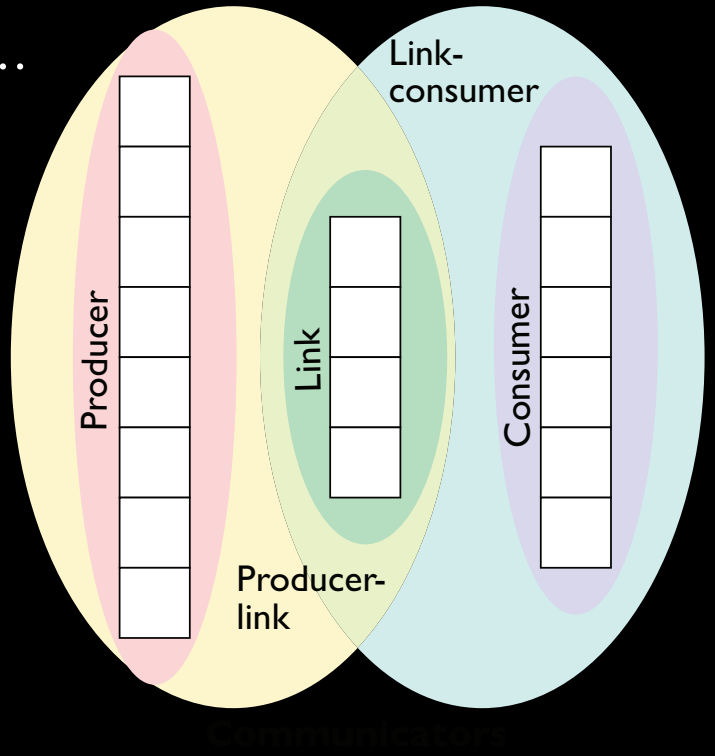
- Translate task-level put/get calls into rank-level MPI calls
- Links (can) have resources too
  - Run a parallel program, (almost) just like a node



Communicators

- All tasks can be one single MPI program, or
- Tasks can be separate MPI programs launched by MPMD
  - mpiexec prog1 –n n_nodes1 : prog2 –n n_nodes2 : …
- Either way, MPI_COMM_WORLD is

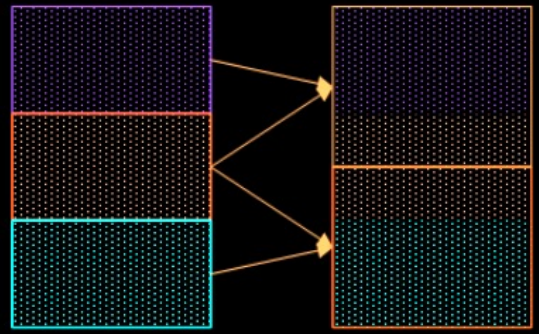
  n_nodes1 + n_nodes2 + …
- Create many smaller communicators from MPI_COMM_WORLD
  - 5 communicators for each producer-consumer pair
  - Use MPI-3's MPI_Comm_create_group



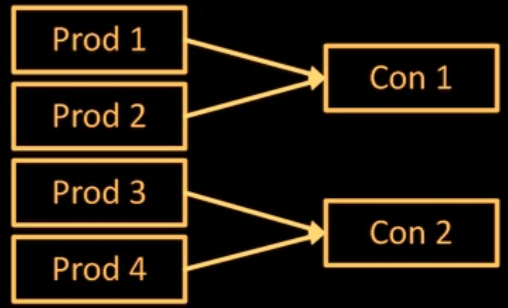
Common Redistribution Patterns

Different ways to split/merge a data model on M producer ranks to N consumer ranks

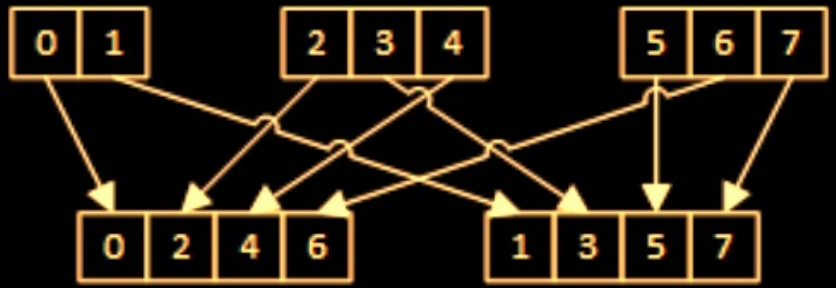
- Bounding Box
- Block
- Round robin
- Contiguous



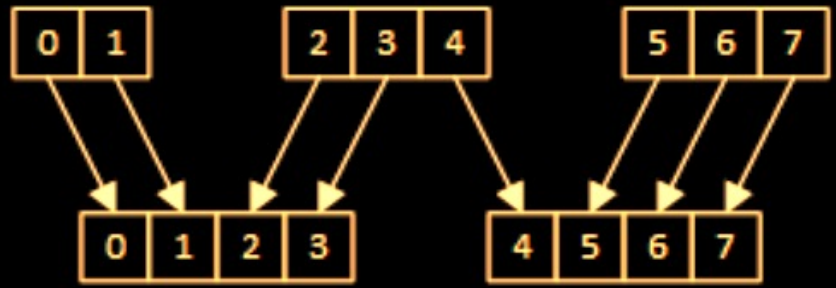
Bounding box redistribution of 3 producer ranks to 2 consumer ranks



Block redistribution of 4 producer ranks to 2 consumer ranks



Round robin redistribution 3 producer ranks to 2 consumer ranks

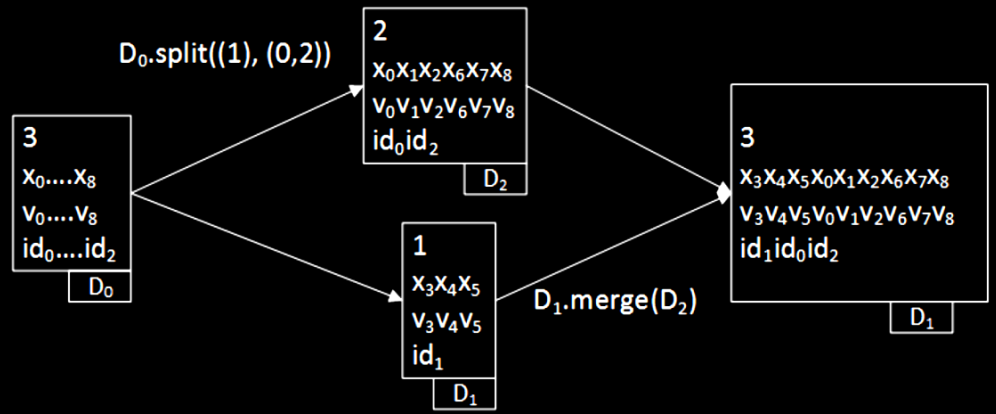


Contiguous redistribution of 3 producer ranks to 2 consumer ranks

Splitting and Merging Data

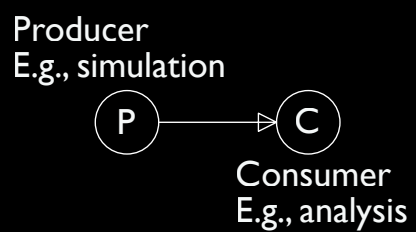
Containers abstraction

- Annotate fields in a data model with tags
- Tags tell runtime how to split/merge the fields
- Push tagged fields into a container
- Fields are any subset of any data model



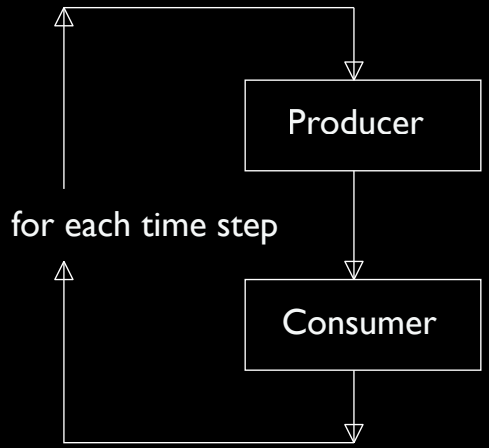

```
struct State
{
    int     n_pts;      // size: 1
    float* pos;         // size: n_pts * 3
    float* vel;         // size: n_pts * 3
    int *   pt_id;      // size: n_pts
};

void main()
{
    State state;
    pConstructData container;

    SimpleFieldi n_pts(state.n_pts);

    container->appendData("n_pts",  n_pts,
        DECAF_NO_FLAG,                      // type
        DECAF_SHARED,                       // scope
        DECAF_SPLIT_SUBTRACT_VALUE, // split
        DECAF_MERGE_ADD_VALUE);     // merge

    // similar for other fields

    decaf->put (container);
}
```

Time and Space Division



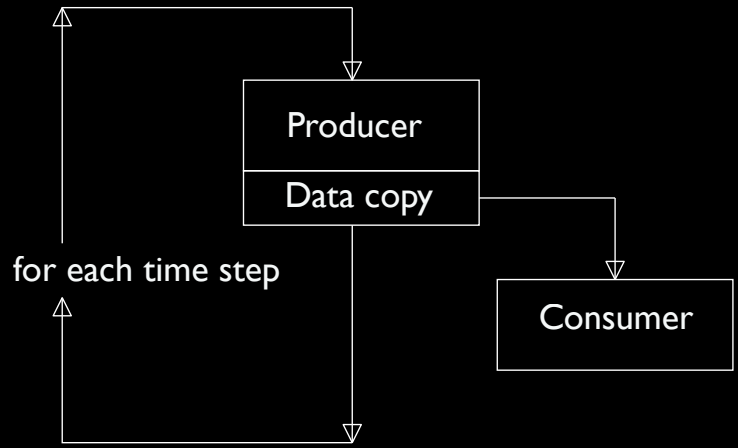
Workflow Graph

Conceptual workflow graph with one producer and one consumer



Time Division Coupling

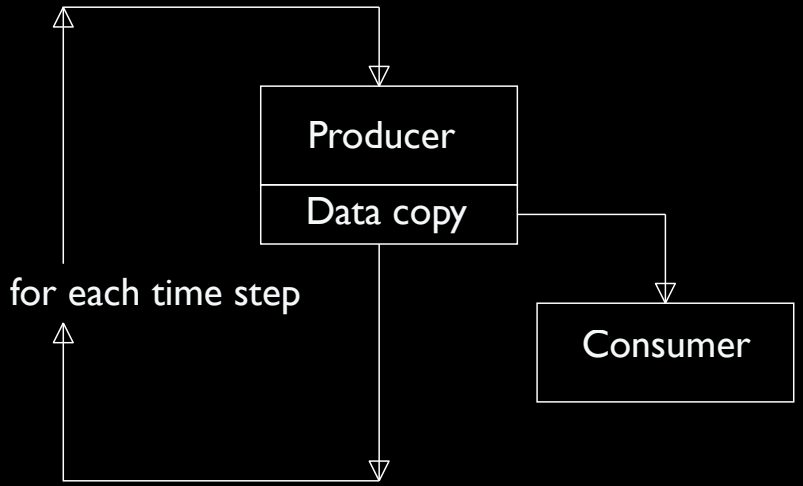
One way to couple the tasks is in the same space and sequential in time.



Space Division Coupling

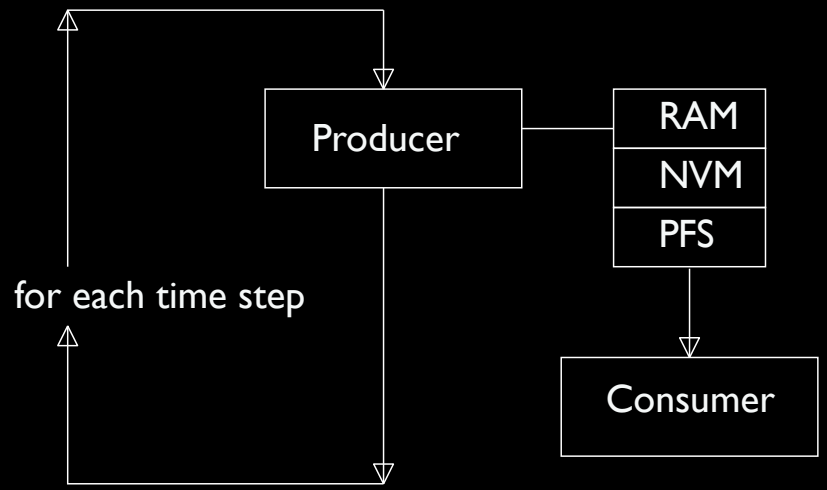
The other way to couple the tasks is to use more space but execute concurrently in time.

Data Rates Can Vary



Space Division Coupling

Space division coupling has to accommodate varying data rates of producer and consumer

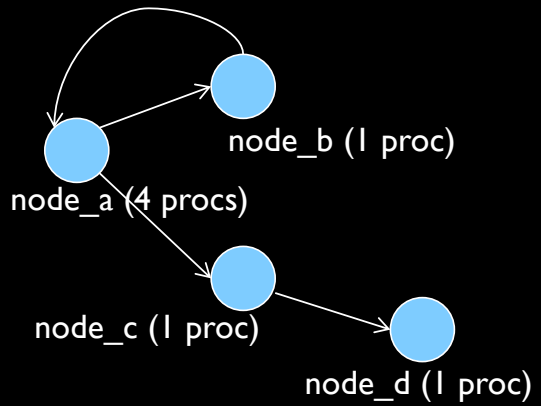


Space Division Coupling with Flow Control

Multiple levels of memory/storage can act as burst buffers to allow a fast producer to couple to slow consumer

Python Workflow Definition

- Define in Python
- Add nodes and edges
- Add attributes to nodes and edges
- Convert to JSON
- Read JSON in application

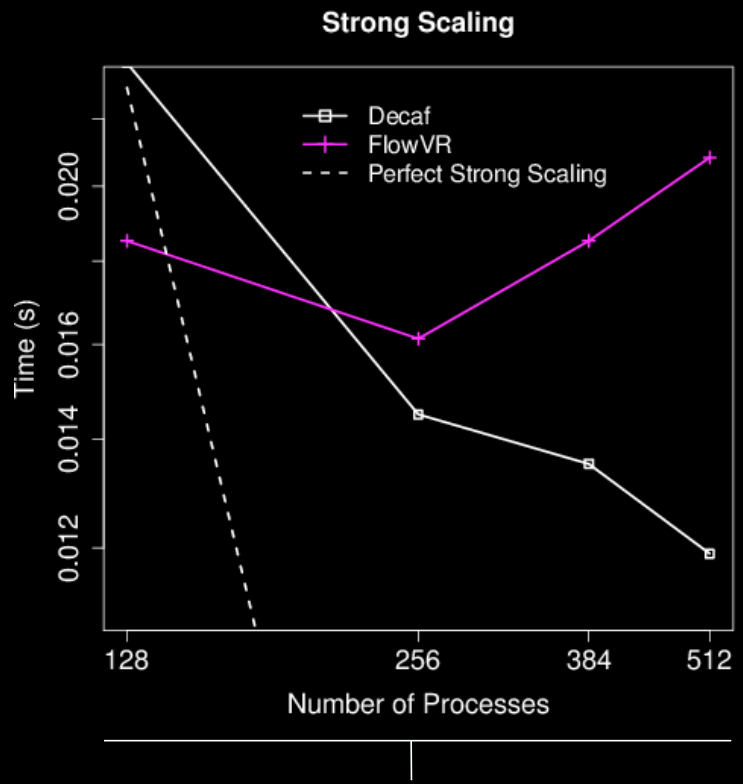


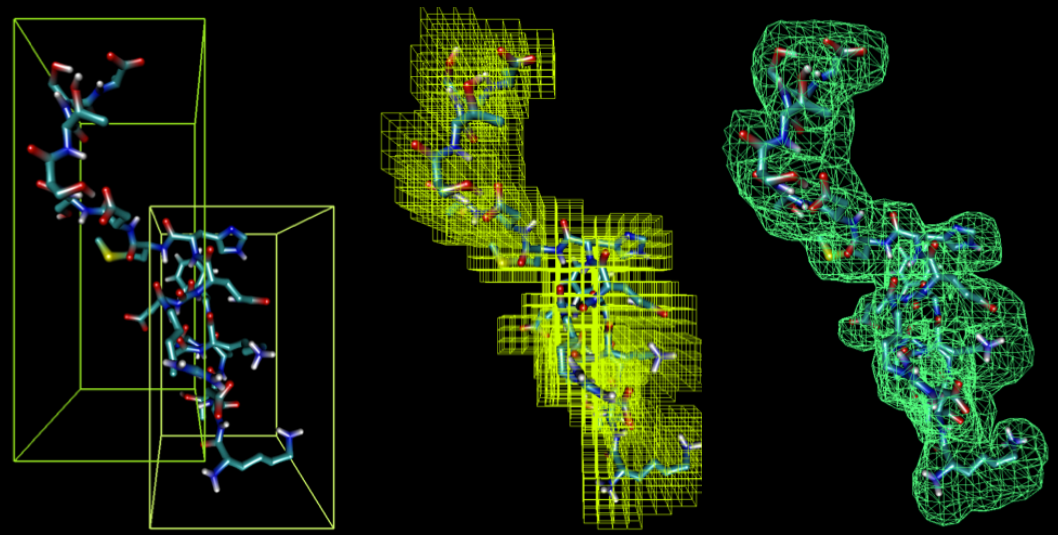

```
# initialize
import networkx as nx
import os
import imp
wf = imp.load_source('workflow', os.environ['DECAF_PREFIX'] + '/python/workflow.py')
mod_path = os.environ['DECAF_PREFIX'] + '/examples/direct/mod_cycle_4nodes.so'

# define workflow graph
w = nx.DiGraph()

w.add_node("node_b", start_proc=5, nprocs=1, func='node_b')
w.add_node("node_d", start_proc=9, nprocs=1, func='node_d')
w.add_node("node_c", start_proc=7, nprocs=1, func='node_c')
w.add_node("node_a", start_proc=0, nprocs=4, func='node_a')

w.add_edge("node_c", "node_d", start_proc=8,  nprocs=1, func='dflow', path=mod_path,
           prod_dflow_redist='count', dflow_con_redist='count')
w.add_edge("node_a", "node_b", start_proc=4,  nprocs=1, func='dflow', path=mod_path,
           prod_dflow_redist='count', dflow_con_redist='count')
w.add_edge("node_a", "node_c", start_proc=6,  nprocs=1, func='dflow', path=mod_path,
           prod_dflow_redist='count', dflow_con_redist='count')
w.add_edge("node_b", "node_a", start_proc=10, nprocs=1, func='dflow', path=mod_path,
           prod_dflow_redist='count', dflow_con_redist='count')

# convert the graph into a JSON config file
wf.workflowToJson(w, mod_path, "cycle.json")
```

Four Examples

Courtesy Matthieu Dreher

Data Redistribution in Molecular Dynamics





Three different redistributions are performed while computing an isosurface from an MD simulation of 54,000 lipids(2.1M particles). [Dreher et al. 2014]

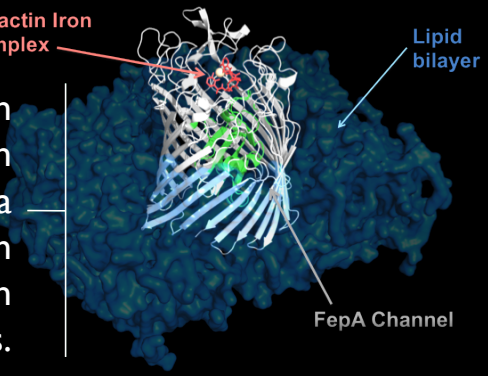
We applied the Decaf redistribution library to the Gromacs molecular dynamics code in order to visualize isosurfaces from molecular density. Code complexity was reduced dramatically, while maintaining performance improved.

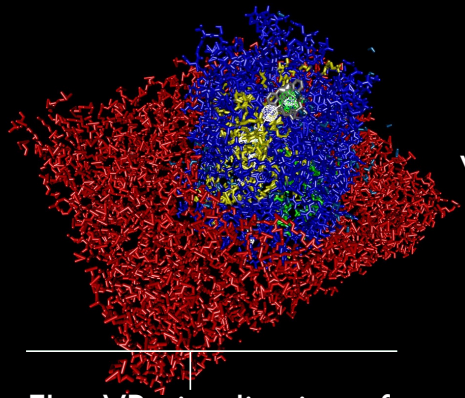
Dreher and Peterka, Bredala: Semantic Redistribution for In Situ Applications . Cluster 2016.

Courtesy Matthieu Dreher

Computational Steering in Molecular Dynamics

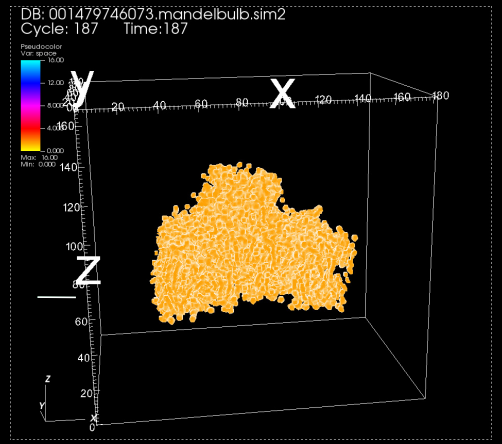
The FepA protein is a channel on the surface of a cell through which iron can pass.

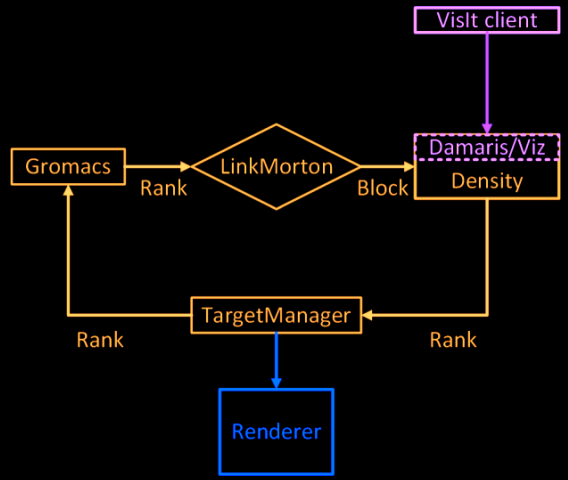




FlowVR visualization of the steering progress.

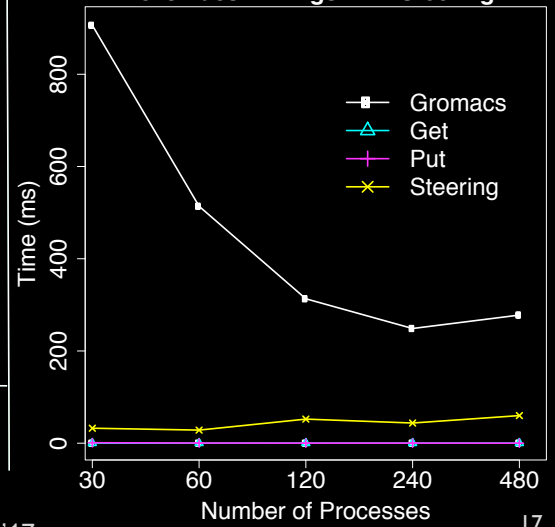
VisIt visualization of the molecular density.





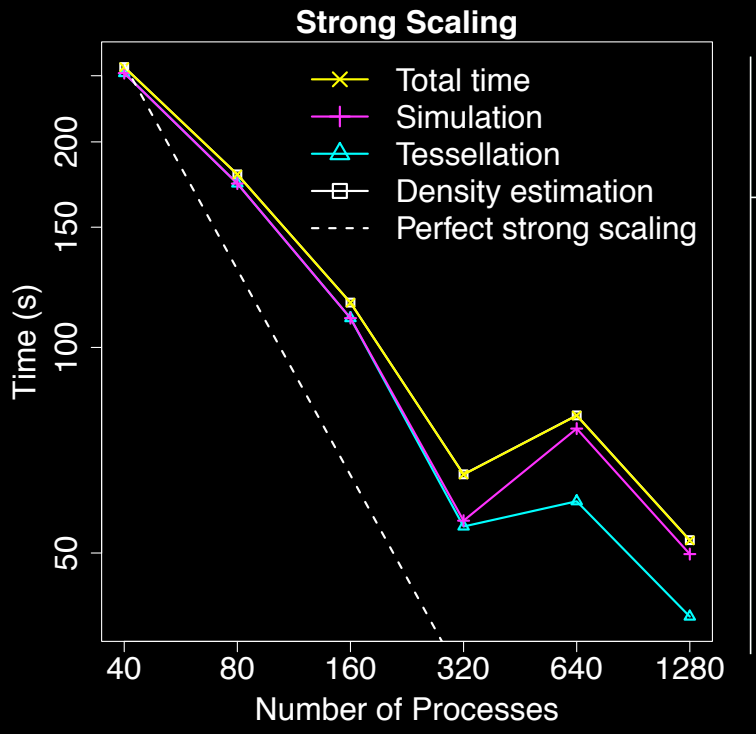
Steering workflow with Decaf, Damaris, and FlowVR. Decaf and FlowVR tasks are in space-division mode while Damaris is in time-division mode.

Strong scaling of the steering pipeline without visualization. The measured time is the average over100 iterations.



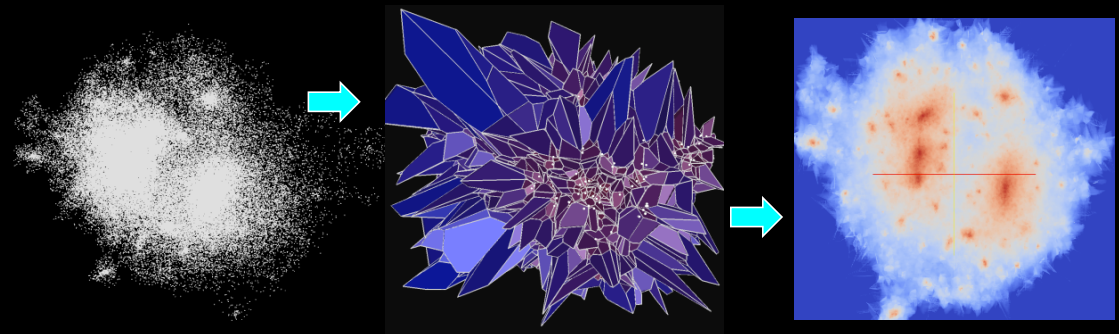
Dreher and Peterka, Decaf: Decoupled Dataflows for In Situ Workflows. Submitted to HPDC'17.

Density Estimation in Cosmology



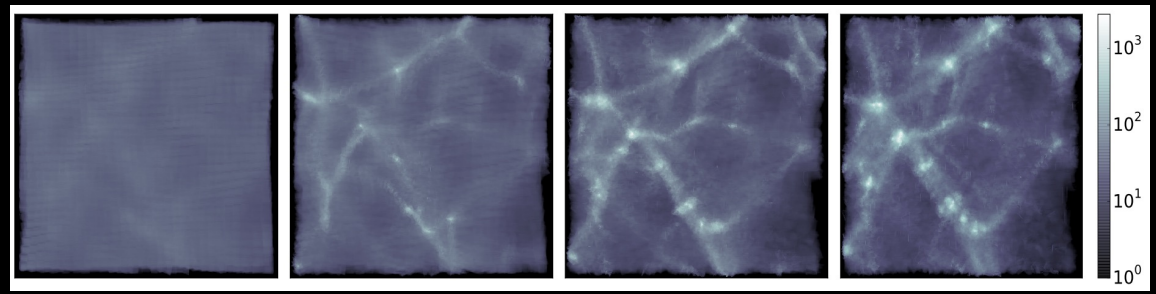
Strong scaling of the end-to-end workflow shows good efficiency. The analysis time effectively overlaps the simulation time.

Density estimation: Tessellations as intermediate representations enable accurate regular grid density estimators.



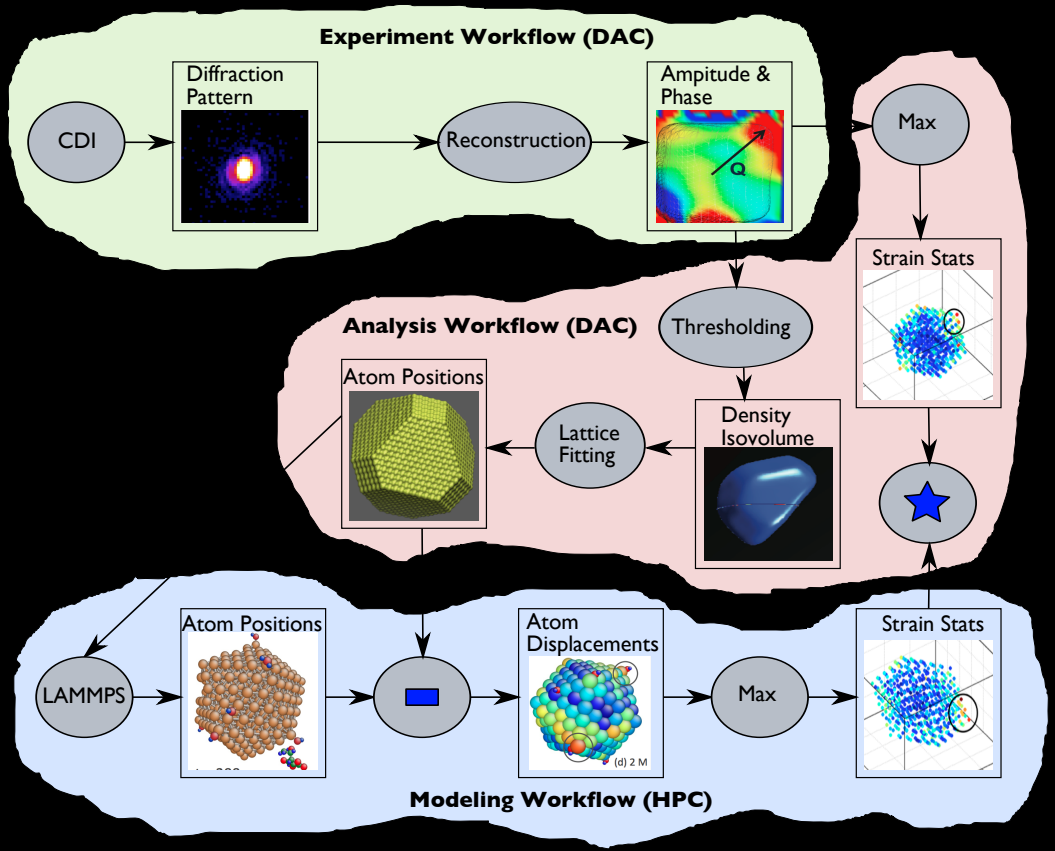
Peterka et al., Self-Adaptive Density Estimation, SIAM SISC 2016.

Output of in situ cosmology analysis workflow at four time steps



Dreher and Peterka, Decaf: Decoupled Dataflows for In Situ Workflows. Submitted to HPDC'17.

Workflows Combining Simulation and Experiment



Science workflow for the comparison of a molecular dynamics simulation with a high-energy X-ray microscopy of the same material system includes three interrelated computational (HPC) and distributed area (DAC) experimental workflows.

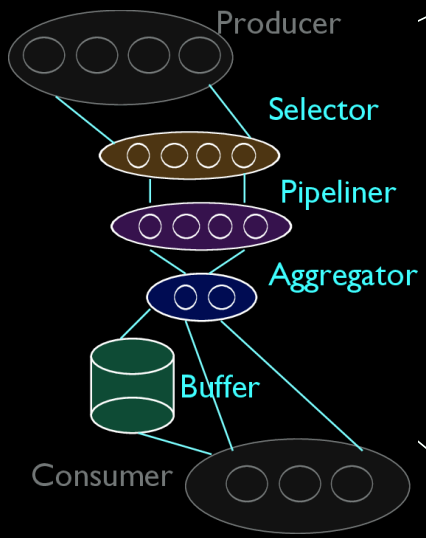
Open question: How to combine different (HPC and DAC) WMSs?

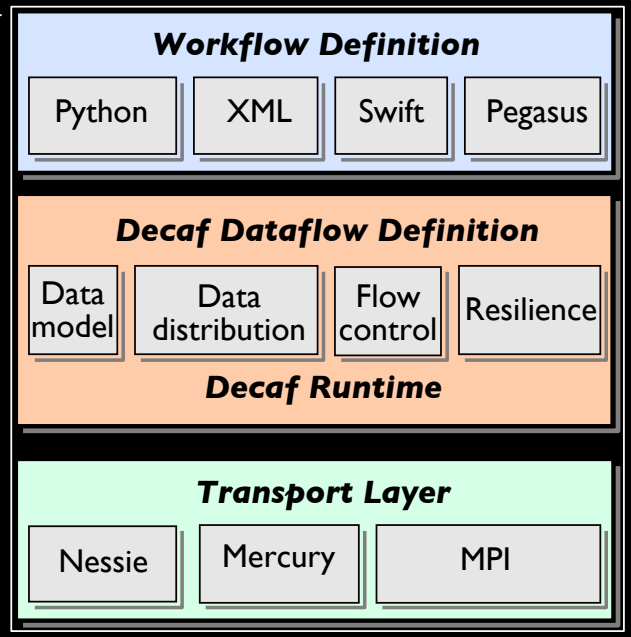
Ulvestad et al.: In-situ 3D Imaging of Catalysis Induced Strain in Gold Nanoparticles. Physical Chemistry Letters, 2016.

References

Tom Peterka,
Franck Cappello, ANL
Jay Lofstead, SNL
bitbucket.org/tpeterka1/decaf

Decaf: Decoupled Dataflows





- If Decaf is the only workflow software, then it needs to do manage the workflow
  - Have a way for the user to define the workflow graph (Python)
  - Instantiate the graph (Decaf constructor)
  - Launch the tasks (run the nodes)
  - Communicate (run the links)
  - Terminate cleanly (when one of the tasks does)
- But Decaf doesn't have to be the top level workflow manager: e.g., Swift or Damaris or FlowVR or ADIOS or PyCOMPSs
  - In such cases, Decaf just runs the links

Decaf generates dataflows for workflows.

- Decoupled workflow links with configurable dataflow
- Data redistribution patterns
- Flow control
- Resilience

Future Work

- Dynamic workflows
  - Resources and even graph topology
- Shared dataflows (shared resources in general)
  - Shared by more than 2 producer/consumer nodes
- Architectures change
  - Deep memory/storage, heterogeneous nodes
  - Shared + distributed hybrid parallelism
- EOD + computing
  - HPC + DAC workflows
- Deeper software stacks
  - Integration with other systems
  - Both above (workflow systems)
  - And below (transport layers, storage services, operating systems)

Further Reading

- Deelman, E., Peterka, T., et al.: The Future of Scientific Workflows. Report of the DOE NGNS/CS Scientific Workflows Workshop, 2016.
- Wozniak, J., Peterka, T., Armstrong, T., Dinan, J., Lusk, E., Wilde, M., Foster, I.: Dataflow Coordination of Data-Parallel Tasks via MPI 3.0. EuroMPI, 2013.
- Dorier, M., Dreher, M., Peterka, T., Wozniak, J., Antoniu, G., Raffin, B.: Lessons Learned from Building In Situ Coupling Frameworks. Proceedings of ISAV 2015.
- Peterka, T., Croubois, H., Li, N., Rangel, E., Cappello, F.: Self-Adaptive Density Estimation of Particle Data. SIAM SISC 2016.
- Dreher, M., Peterka, T.: Bredala: Semantic Data Redistribution for In Situ Applications. Proceedings of IEEE Cluster 2016, Taipei, Taiwan, 2016.
- Dorier, M., Antoniu, G., Cappello, F., Snir, M., Sisneros, R., Yildiz, O. Ibrahim, S., Peterka, T., Orf, L.: Damaris: Addressing Performance Variability in Data Management for Post-Petascale Simulations. To appear in ACM ToPC journal, 2016.
- Dreher.M., Peterka, T.: Decaf: Decoupled Dataflows for In Situ Workflows. Submitted to HPDC'17.

Acknowledgments

Facilities

Argonne Leadership Computing Facility (ALCF)

Oak Ridge National Center for Computational Sciences (NCCS)

National Energy Research Scientific Computing Center (NERSC)

Funding

DOE SDMAV Exascale Initiative

DOE SciDAC SDAV Institute

People

Franck Cappello (ANL), Matthieu Dreher (ANL), Jay Lofstead (SNL), Patrick Widener (SNL)

Tom Peterka
tpeterka@mcs.anl.gov
Mathematics and Computer Science Division