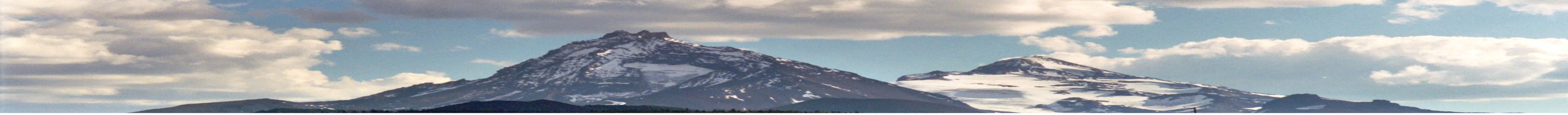


Integration of Modern HPC Performance Analysis in Vlasiator for Sustained Exascale

Camille Coti[1], Yann Pfau-Kempf[3], Markus Battarbee[3], Urs Ganse[3]
Sameer Shende[2], Kevin Huck[2], Jordi Rodriquez[2], Leo Kotipalo[3]
<u>Allen D. Malony[2]</u>, Minna Palmroth[3,4]

École de Technologie Supérieure
Montréal, Québec, Canada[1]

University of Oregon
Eugene, Oregon, USA[2]

University of Helsinki
Helsinki, Finland[3]

Finnish Meteorological Institute
Helsinki, Finland[4]

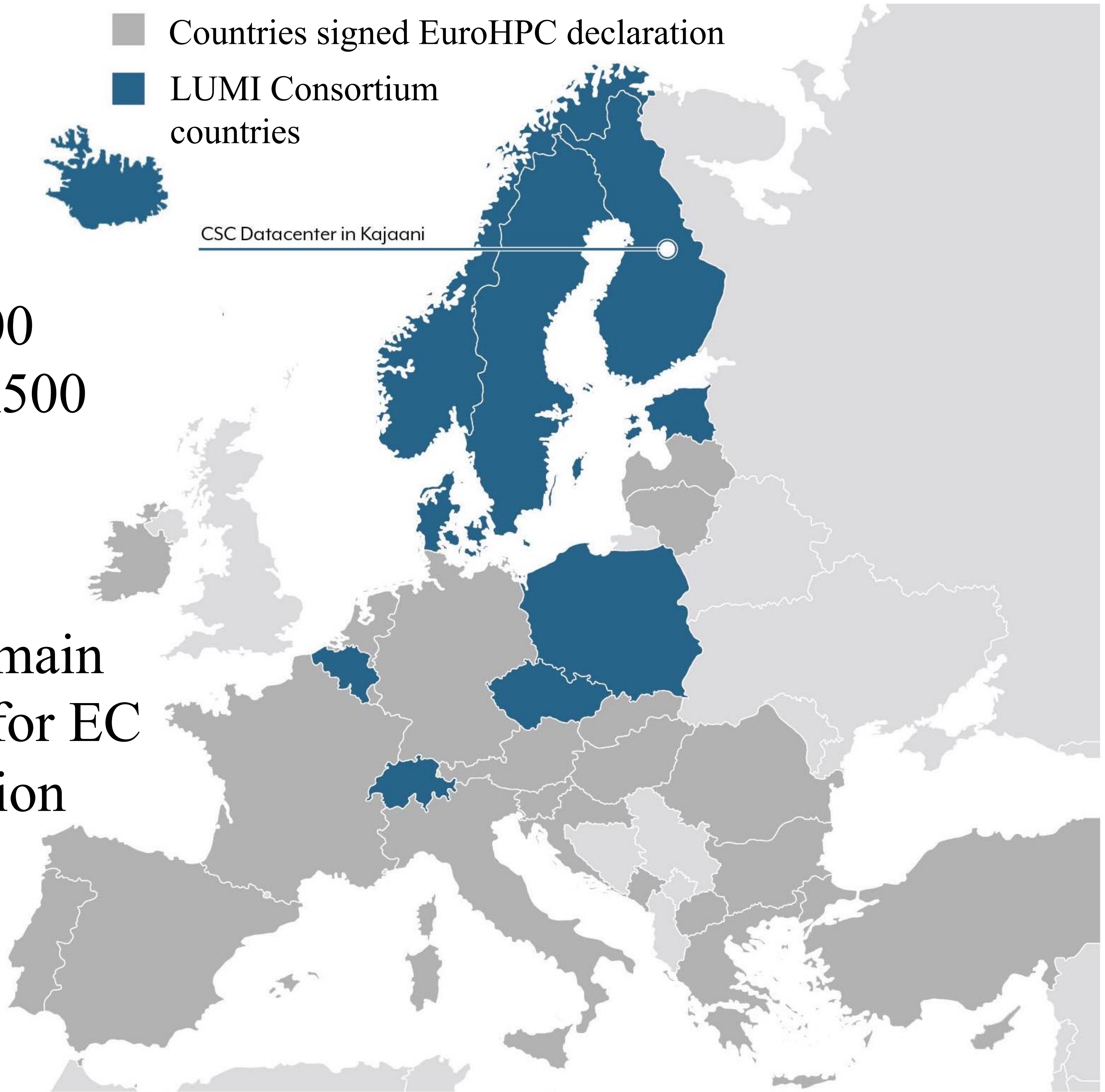
Motivation

- Heterogeneous computing with accelerated-node hardware is delivering extreme computational power
- Productive utilization of exascale systems for real-world workloads will require sustained application performance
- Heterogeneity raises application development concerns
  - Codes need to be (re-)developed with hybrid programming methods
  - High-level programming abstractions for performance portability
- Factors contributing to performance behavior and scaling efficiency will make performance variability more acute
- High importance to integrate HPC performance measurement and analysis technologies with applications

Opportunity

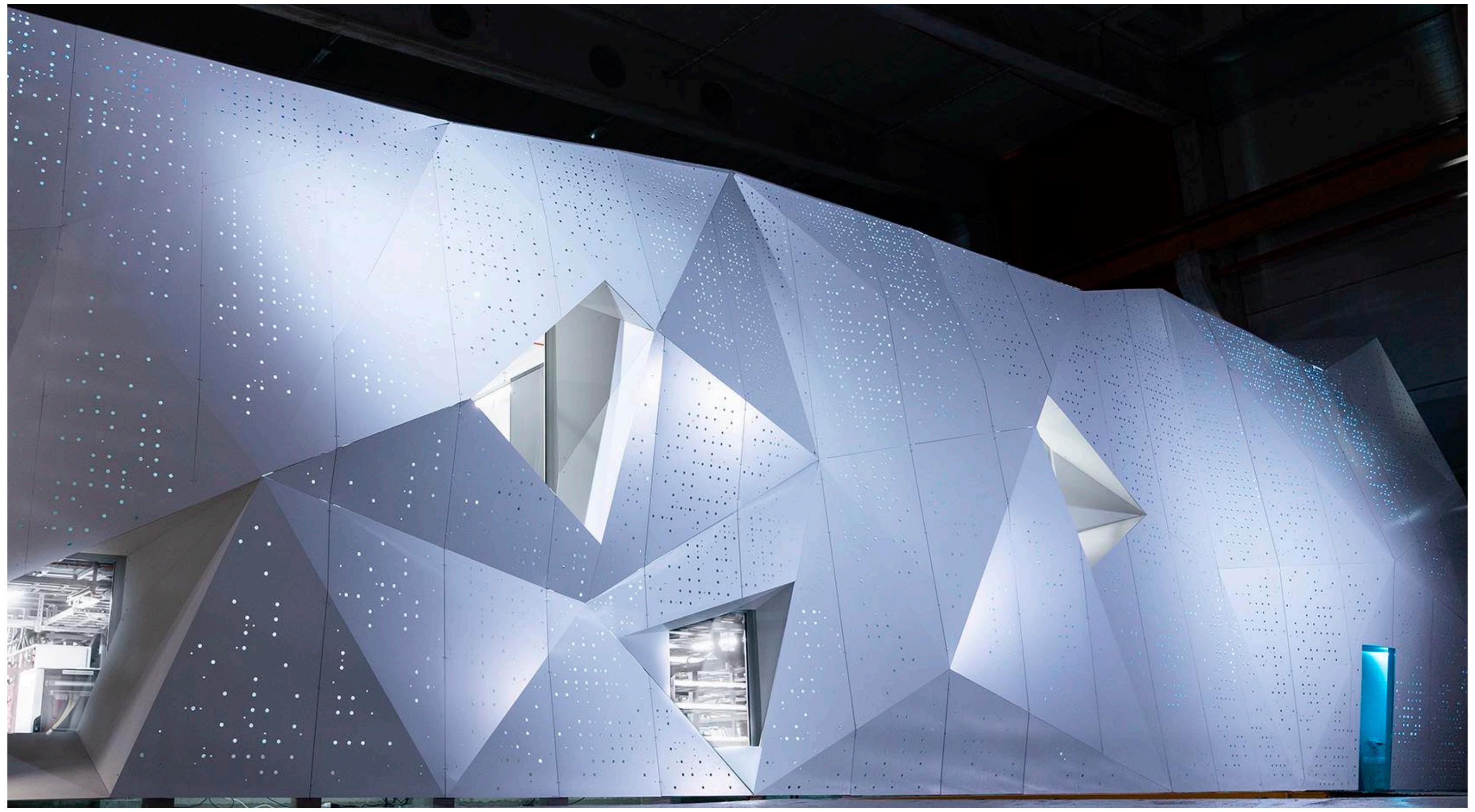
- Fulbright-Nokia Distinguished Chair
  - Study performance of HPC scientific and big data applications on LUMI supercomputer
  - University of Helsinki research host
  - Bring TAU tools to CSC–IT Center for Science
- Proposed project to work with Vlasiator team
  - Our groups have been working since summer 2022
- Paper reports research progress thus far

LUMI (Large Unifed Modern Infrastructure)



#3 Top500
#7 Green500

LUMI is main platform for EC "Destination Earth" research program



https://www.lumi-supercomputer.eu

Vlasiator

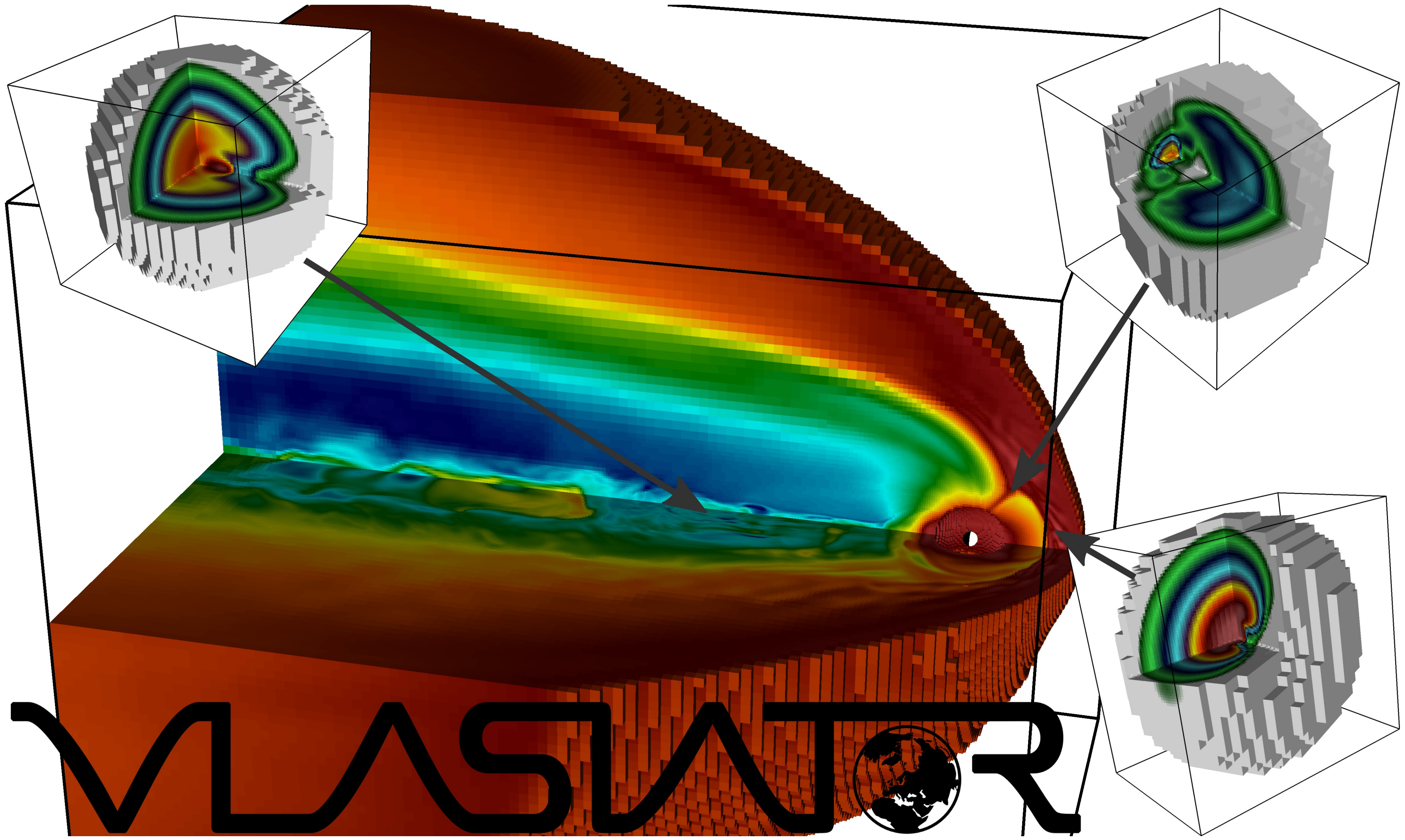
- Global hybrid-Vlasov simulation software
  - Modeling collisionless space plasma physics
  - Earth's magnetosphere and surrounding solar wind
- Simulates evolution of ion phase-space density
  - Adaptive Cartesian 3D-3V grid (three spatial, three velocity)
  - Closure provided by field solver acting on separate, uniform grid
    - fluid description of electrons and assumptions of quasineutrality
    - magnetohydrodynamic Ohm's law with Hall and electron pressure gradient
- Spatial grid can be cell-based refined to increase resolution in ROI
  - Either using parametrized regions
  - Or refining adaptively during runtime
- Velocity grid is sparse, storing and propagating ions only in regions of velocity space with non-negligible phase-space density

Simulation of Supersonic Solar Wind

Interaction of the supersonic solar wind with the Earth's magnetosphere

A shock and sheath form, encompassing the magnetosphere, in three spatial dimensions

Insets highlight the phase-space density distribution in the three velocity dimensions at selected locations

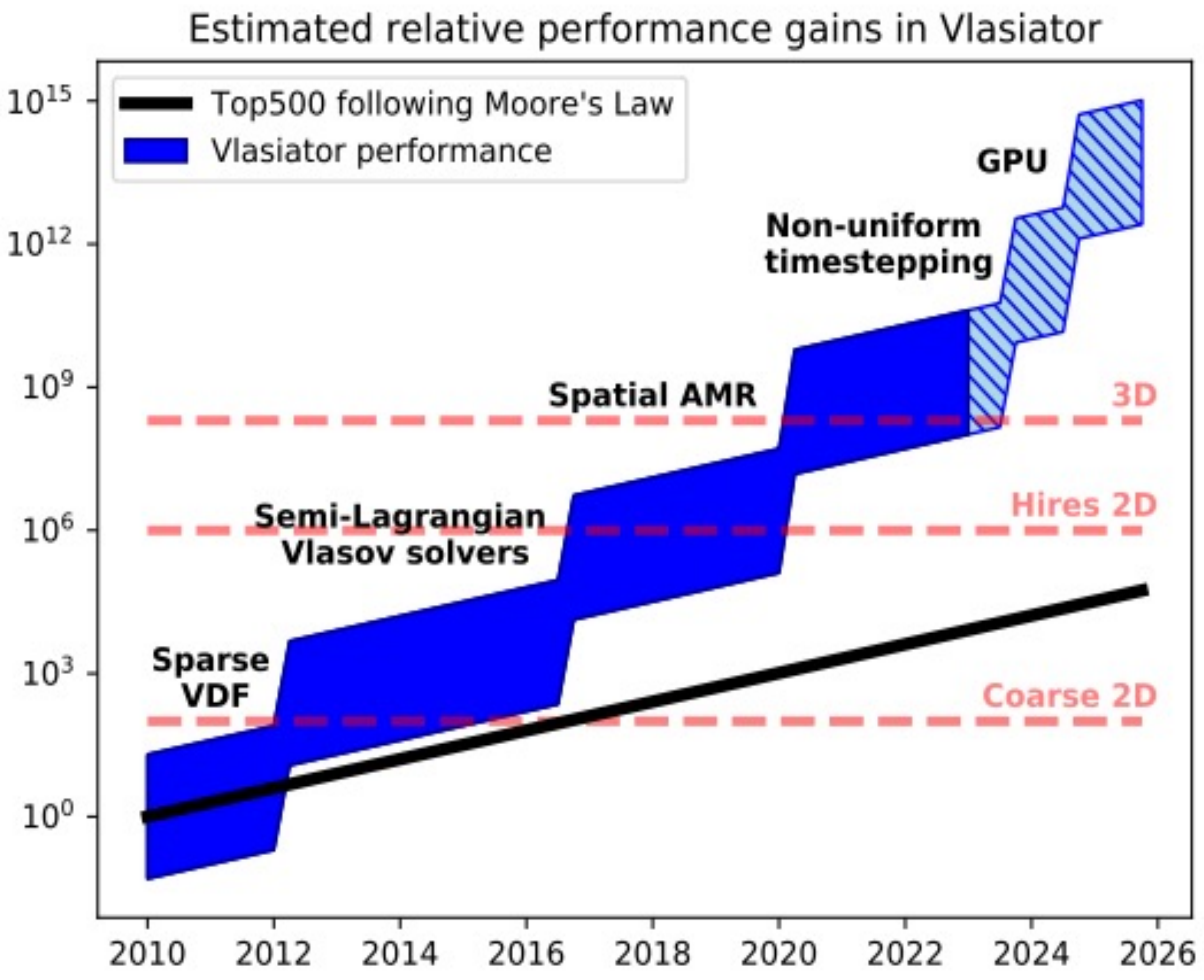


Vlasiator Code Development

- Written in C++17 with hybrid parallelization
  - CPU SIMD vectorization
  - OpenMP threading
  - MPI
- Simulation space is decomposed over MPI domains
  - Zoltan library performing recursive coordinate bisection
  - More recently using recursive inertial bisection
- Vlasiator does load-balancing in the spatial domain
  - Based on # phase-space cells to propagate in each spatial cell
- Support for SIMT GPU instructions is under development
  - Vlasov solvers
  - Field solver

Relative Performance of Vlasiator Evolution

- Relative performance of Vlasiator normalized to the performance in the beginning of the project
- Global target simulation
  - Parts of the solar wind
  - Large part of the magnetosphere
- Blue areas give the relative performance of Vlasiator compared to TOP500
- Dark blue is past, dashed future



TAU Project at the University of Oregon

- Research and development effort spanning 30+ years
- Focus on parallel performance problems and technologies
- Performance problem solving framework for HPC research
  - Integrated, scalable, flexible, portable
  - Target all parallel programming / execution paradigms
- Integrated performance toolkit (TAU Performance System®)
  - Multi-level performance instrumentation
  - Flexible and configurable performance measurement
  - Widely-ported performance profiling / tracing system
  - Performance data management and data mining
  - Open source (BSD-style license)
- Performance analysis of complex software, systems, applications

TAU Performance System

- Incorporates two performance toolkits
  - Each provides measurement and analysis support
  - ***TAU*** (Tuning and Analysis Utilities)
  - ***APEX*** (Autonomic Performance Environment for Exascale)
- Differ in respects to observation perspective
  - TAU: who is doing the "work" (per thread measurement)
  - APEX: what "work" (task) is done (per task measurement)
- HPC software can require either or both perspectives
- TAU and APEX can be used individually or together

Phiprof

- Profiling library developed by the Vlasiator team
- Used to profile MPI, OpenMP or both
  - Supports C, C++, Fortran 2008
  - Phiprof calls inserted in code
  - Low overhead (<1 μsec)
- Hierarchical timing report
  - Statistics: average, max, min
  - Different timer regions
  - Human-readable hierarchical
- Vlasiator used Phiprof early
  - Guided by estimations of regions that would be computationally expensive
  - Included with Vlasiator releases

```
int label;
phiprof::initialize();
label = phiprof::initializeTimer( "Propagate" );
/* ... */
phiprof::start( "Simulation" );
/* ... */
phiprof::start( label );
/* ... */
phiprof::stop( label );
/* ... */
phiprof::stop( "Simulation" );
/* ... */
```

Phiprof Events in Vlasiator (selection)

- Vlasiator programmers defined meaningful high-level timers to give semantic context for regions of computation
- They defined timers for relevant operations of interest

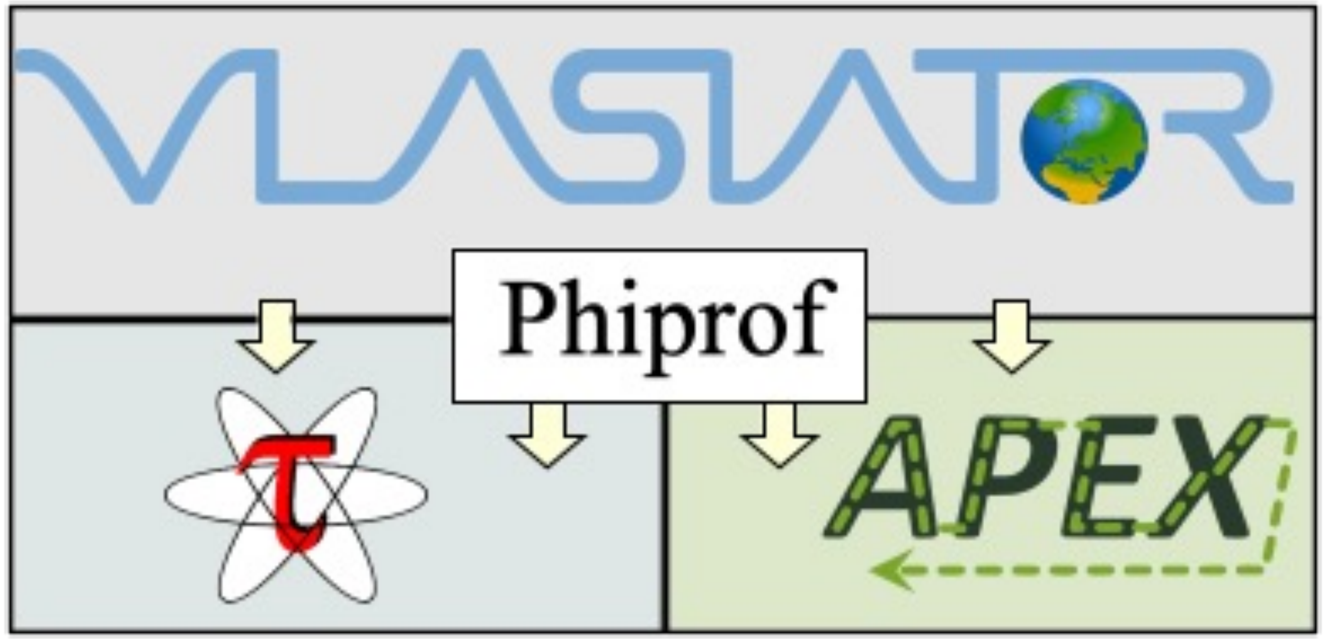
Level	Label	Brief description
1	main	Program `main()` function
2	Initialization	Grid and solver setup
2	report-memory-consumption	Function reporting node memory usage
2	Simulation	Main loop
3	IO	Data and bookkeeping IO operations
3	Propagate	Actual plasma and electromagnetic field propagation
4	Spatial-space	Position space advection
4	Update system boundaries (Vlasov post-translation)	Post-advection update of boundary cells
4	Compute interp moments	Interpolation of density/velocity/pressure between advection and acceleration
4	Propagate Fields	Electric and magnetic field update
4	Velocity-space	Velocity space advection a.k.a. acceleration
4	Update system boundaries (Vlasov post-acceleration)	Post-acceleration update of boundary cells
4	ionosphere-solve	Ionospheric potential update
4	Other	Remaining, non-instrumented sections in level 3 region "Propagate"
3	compute-timestep	Determination of time step limits, update of time step if necessary
3	Balancing load	Rebalancing of the computational load across MPI domains
3	Other	Remaining, non-instrumented sections in level 2 region "Simulation"
1	Other	Remaining, non-instrumented sections in level 1 region "main"

So, what's wrong with Phiprof?

- Nothing!
  - Phiprof makes visible Vlasiator events of interest
  - Vlasiator events provide important *semantic context*
  - But Phiprof sees what it sees and nothing more
- Is it possible to enhance Phiprof with more sophisticated measurement and analysis technology? Yes!
  - Reimplement Phiprof API with another profiling interface
    - TAU instrumentation API (both TAU and APEX work)
  - Run Vlasiator with TAU or/and APEX (without recompiling!)
  - Voilà !

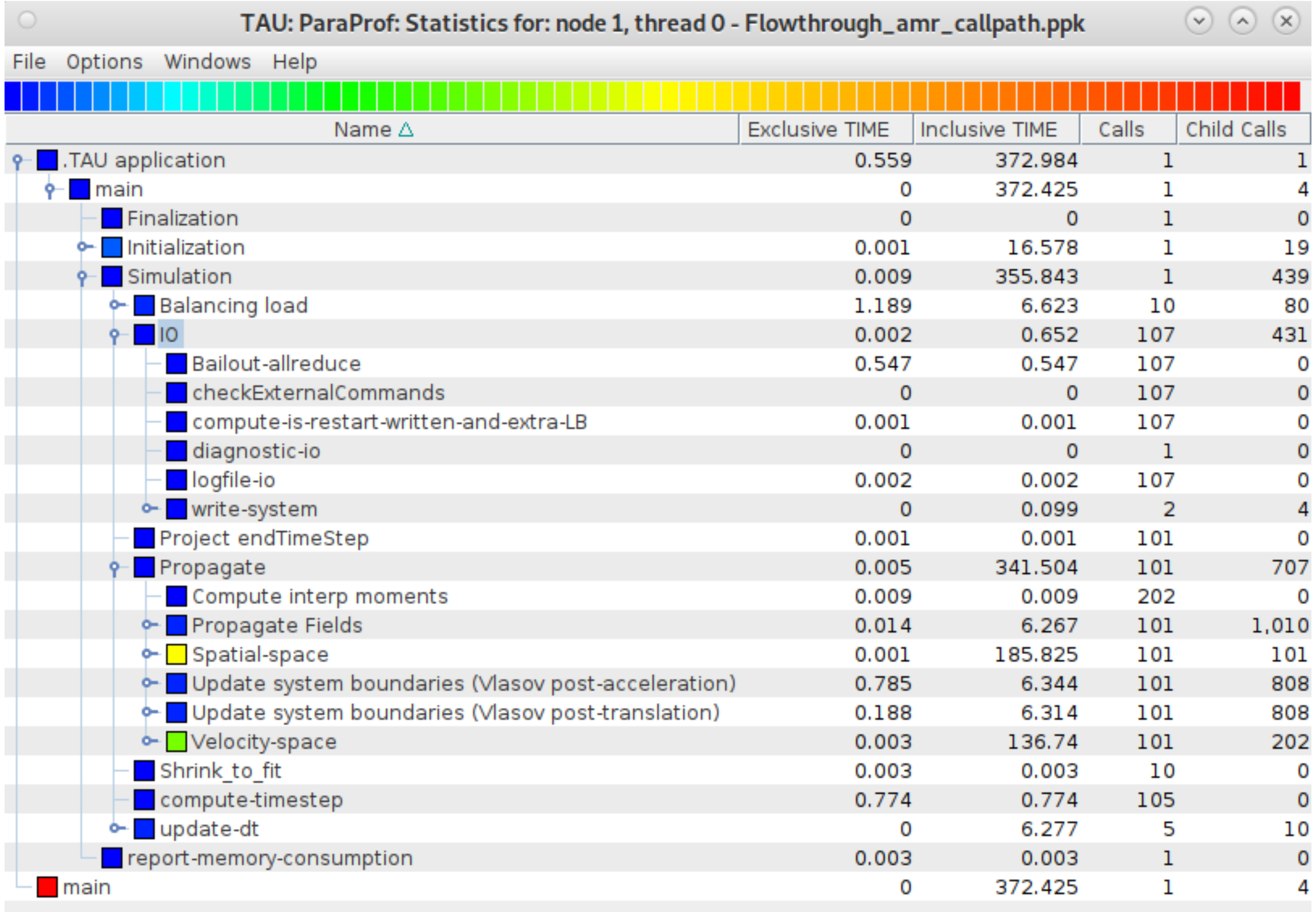
Integrating TAU and APEX with Vlasiator

- Bring full TAU and APEX capabilities to Vlasiator
- New features
  - MPI and OpenMP performance data
  - GPU performance (CUDA, HIP)
  - Profiling and tracing
  - Access to hardware counters
  - Rich analysis tools
  - Broad portability
  - …



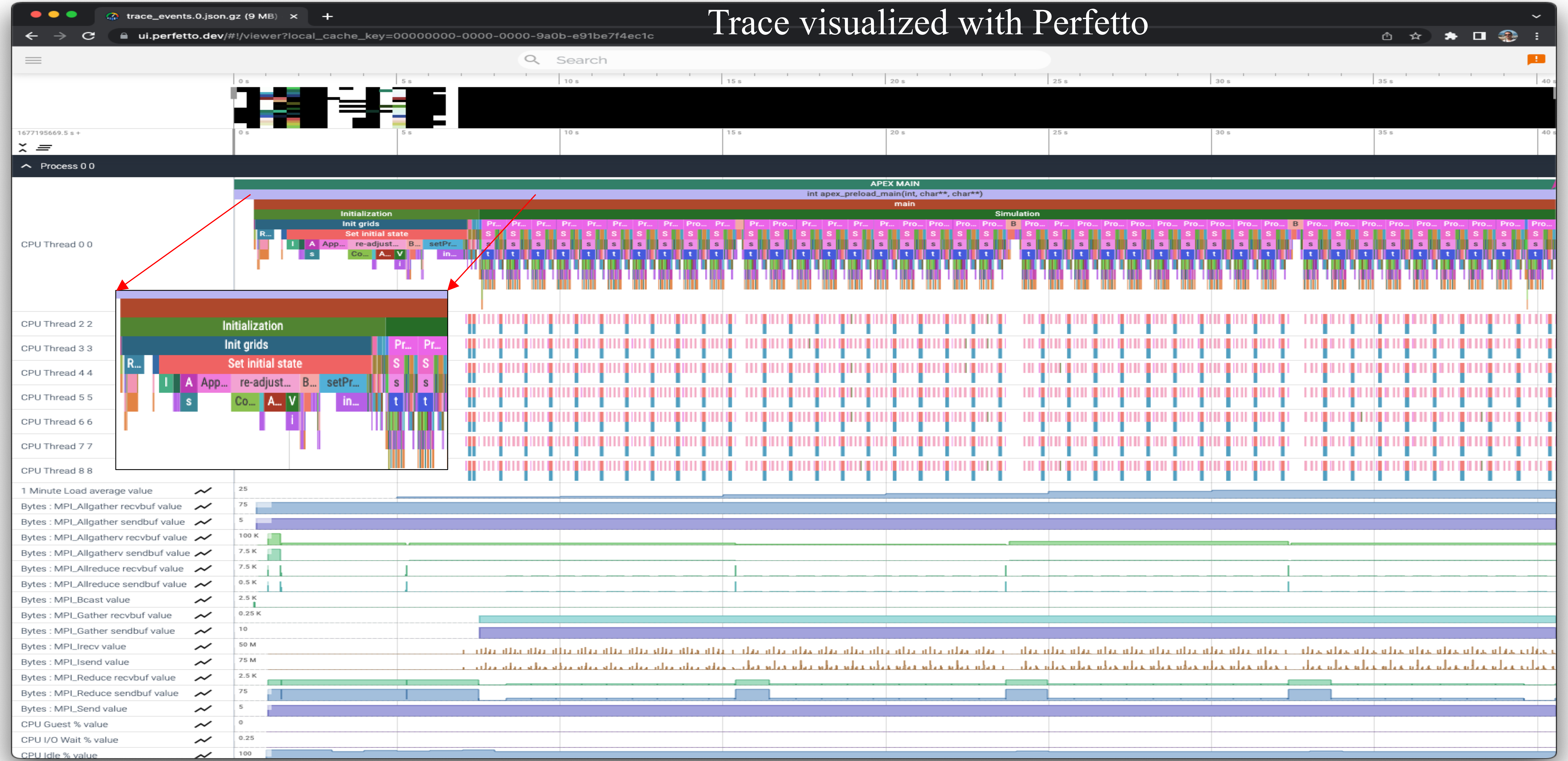
Support for Phiprof Interface

- Reimplement Phiprof API to interface with TAU instrumentation
- Need to correctly support Phiprof hierarchical model
- Vlasiator events appear in TAU measurement
- Immediate access to all of TAU and APEX measurement support

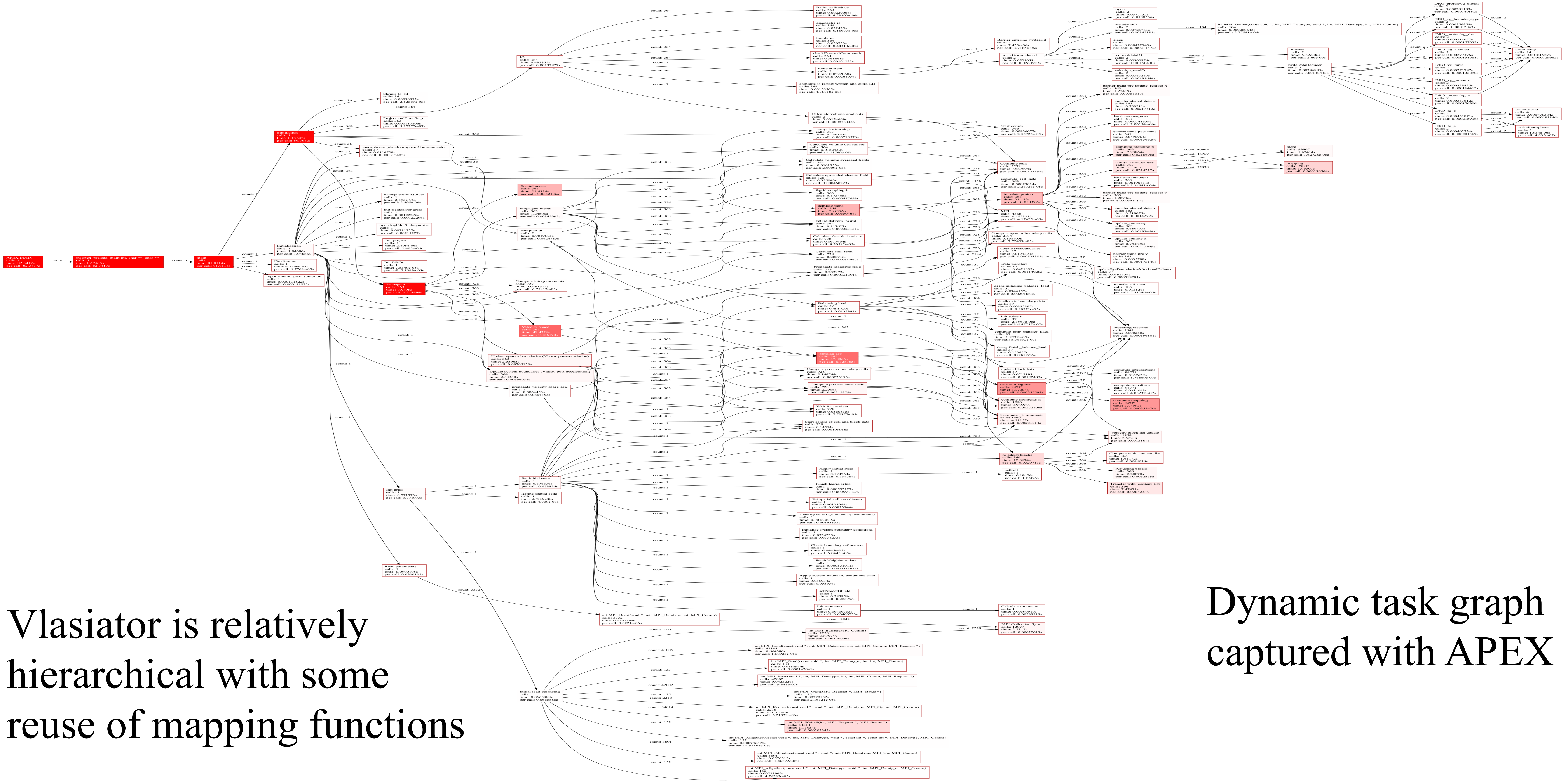


Name △	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	0.559	372.984	1	1
main	0	372.425	1	4
Finalization	0	0	1	0
Initialization	0.001	16.578	1	19
Simulation	0.009	355.843	1	439
Balancing load	1.189	6.623	10	80
IO	0.002	0.652	107	431
Bailout-allreduce	0.547	0.547	107	0
checkExternalCommands	0	0	107	0
compute-is-restart-written-and-extra-LB	0.001	0.001	107	0
diagnostic-io	0	0	1	0
logfile-io	0.002	0.002	107	0
write-system	0	0.099	2	4
Project endTimeStep	0.001	0.001	101	0
Propagate	0.005	341.504	101	707
Compute interp moments	0.009	0.009	202	0
Propagate Fields	0.014	6.267	101	1,010
Spatial-space	0.001	185.825	101	101
Update system boundaries (Vlasov post-acceleration)	0.785	6.344	101	808
Update system boundaries (Vlasov post-translation)	0.188	6.314	101	808
Velocity-space	0.003	136.74	101	202
Shrink_to_fit	0.003	0.003	10	0
compute-timestep	0.774	0.774	105	0
update-dt	0	6.277	5	10
report-memory-consumption	0.003	0.003	1	0
main	0	372.425	1	4

Tracing Support for Vlasiator Events



Task Graph of Vlasiator Execution



Vlasiator is relatively hierarchical with some reuse of mapping functions

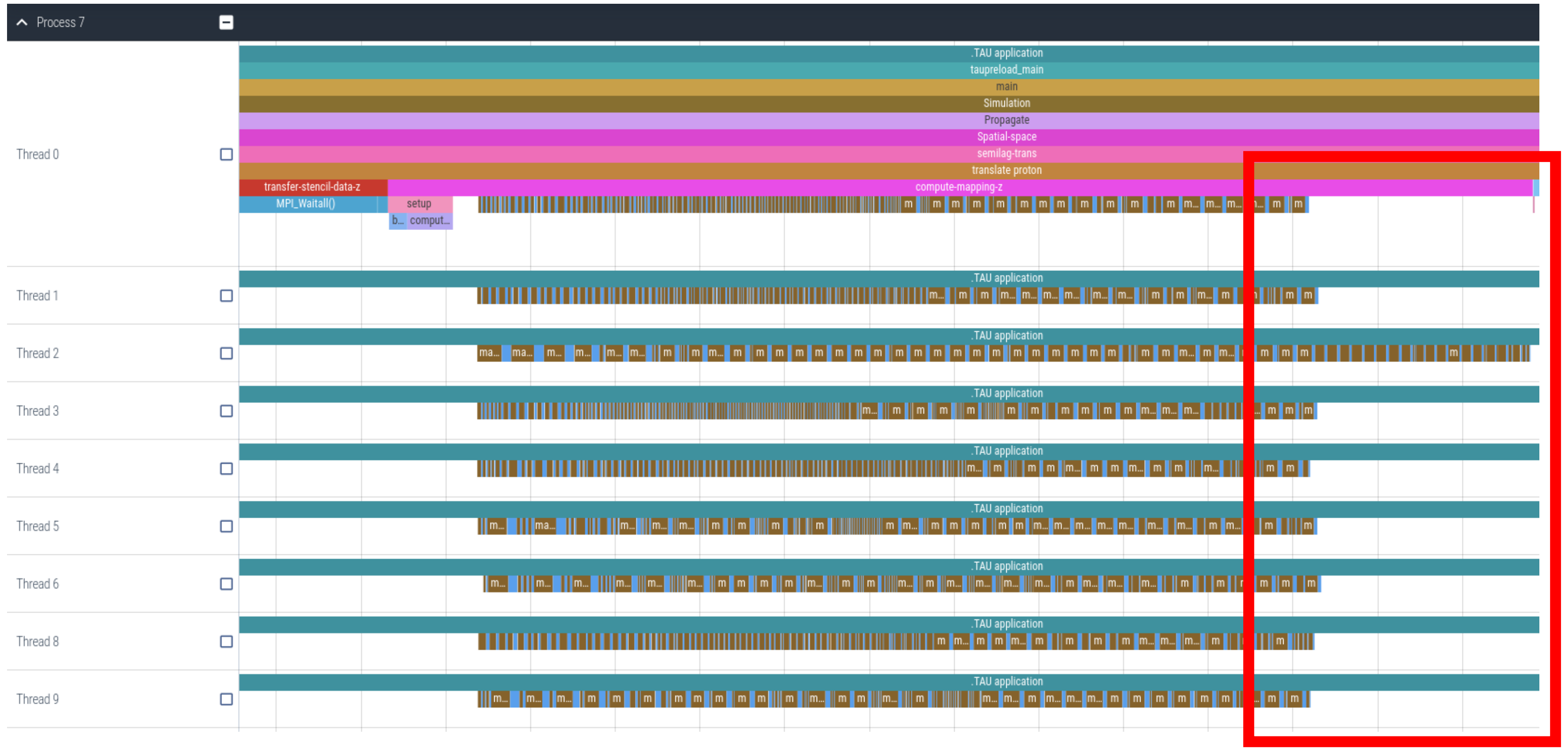
Dynamic task graph captured with APEX

Magnetospheric Simulation on LUMI

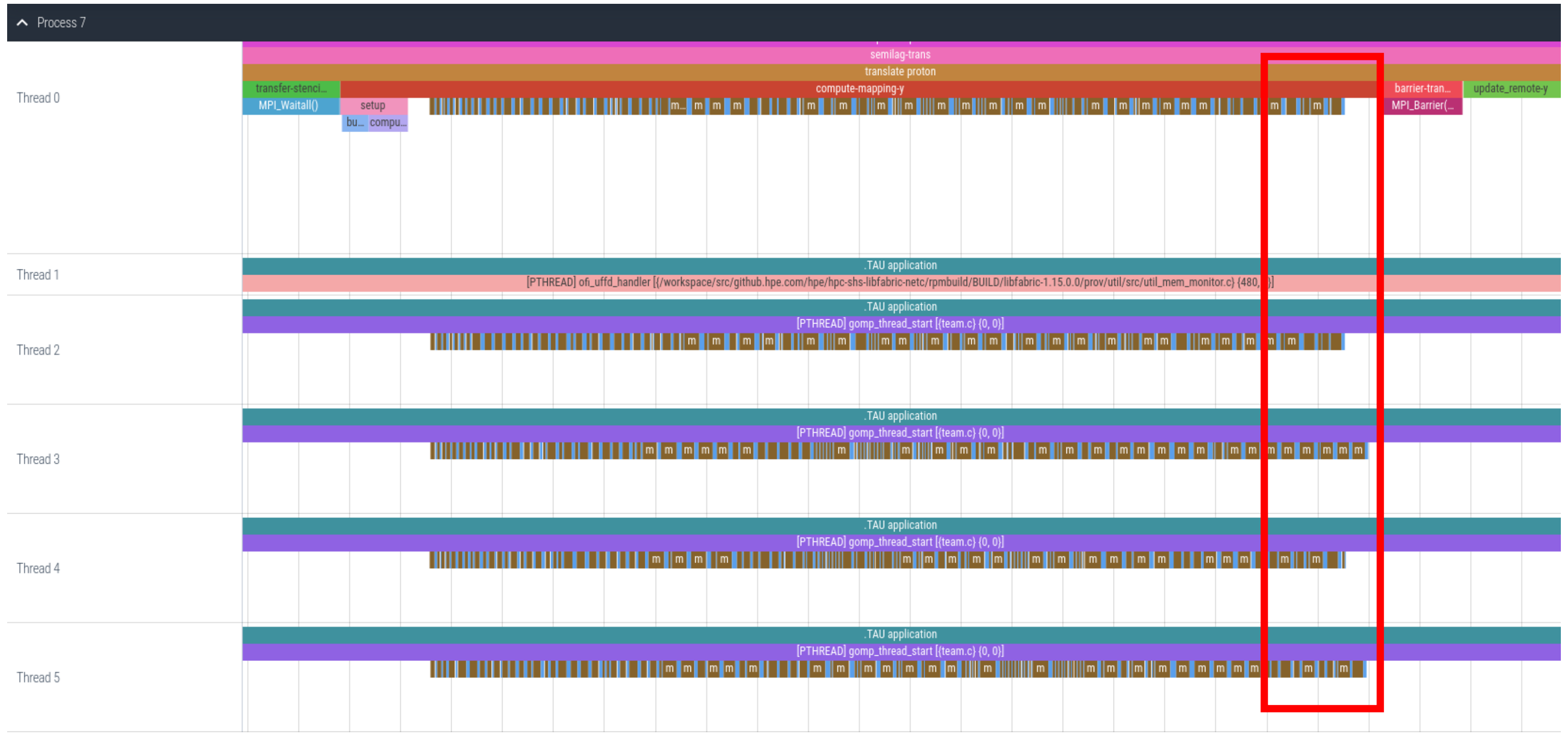
- 250 nodes
- Three main solvers
  - *Spatial-space*
  - *Velocity-space*
  - *Propagate fields*
- Majority of simulation loop time
- Relative contributions at this scale was a surprise

Name	Exclusive TIME	Inclusive TIME ▽	Calls	Child Calls
.TAU application	0.001	1,844.43	1	1
taupreload_main	0.001	1,844.429	1	17
main	0.002	1,842.288	1	4
Simulation	0.007	1,662.342	1	296
Propagate	0.003	1,558.669	71	513
Propagate Fields	0.054	668.607	71	16,330
Spatial-space	0.001	626.189	71	71
Velocity-space	0.003	164.332	71	213
Update system boundaries (Vlasov post-acceleration)	0.096	52.675	71	15,700
Update system boundaries (Vlasov post-translation)	0.099	34.415	71	15,753
ionosphere-solve	10.31	10.326	4	4
fieldtracing-ionosphere-fsgridCoupling	0.023	2.063	4	64
ionosphere-mapDownMagnetosphere	0.007	0.035	4	12
Compute interp moments	0.013	0.013	142	0
ionosphere-calculateConductivityTensor	0.011	0.011	4	0
IO	0.008	80.645	72	364
Balancing load	0.032	16.713	4	924
compute-timestep	3.071	5.146	70	70
ionosphere-updateIonosphereCommunicator	0	1.162	4	20
Shrink_to_fit	0.001	0.001	4	0
Project endTimeStep	0	0	71	0
Initialization	0.001	179.906	1	10
report-memory-consumption	0.023	0.038	1	14
Finalization	0	0	1	0
MPI_Init_thread()	2.112	2.112	1	1
MPI_Finalize()	0.028	0.028	1	0
MPI_Comm_free()	0	0	13	0
MPI_Comm_rank()	0	0	1	0

Task Graph of Vlasiator Execution



Original code scheduled work groups to all OpenMP threads according to the guided strategy with no batch limit, occasionally causing single threads to receive a disproportionate amount of work



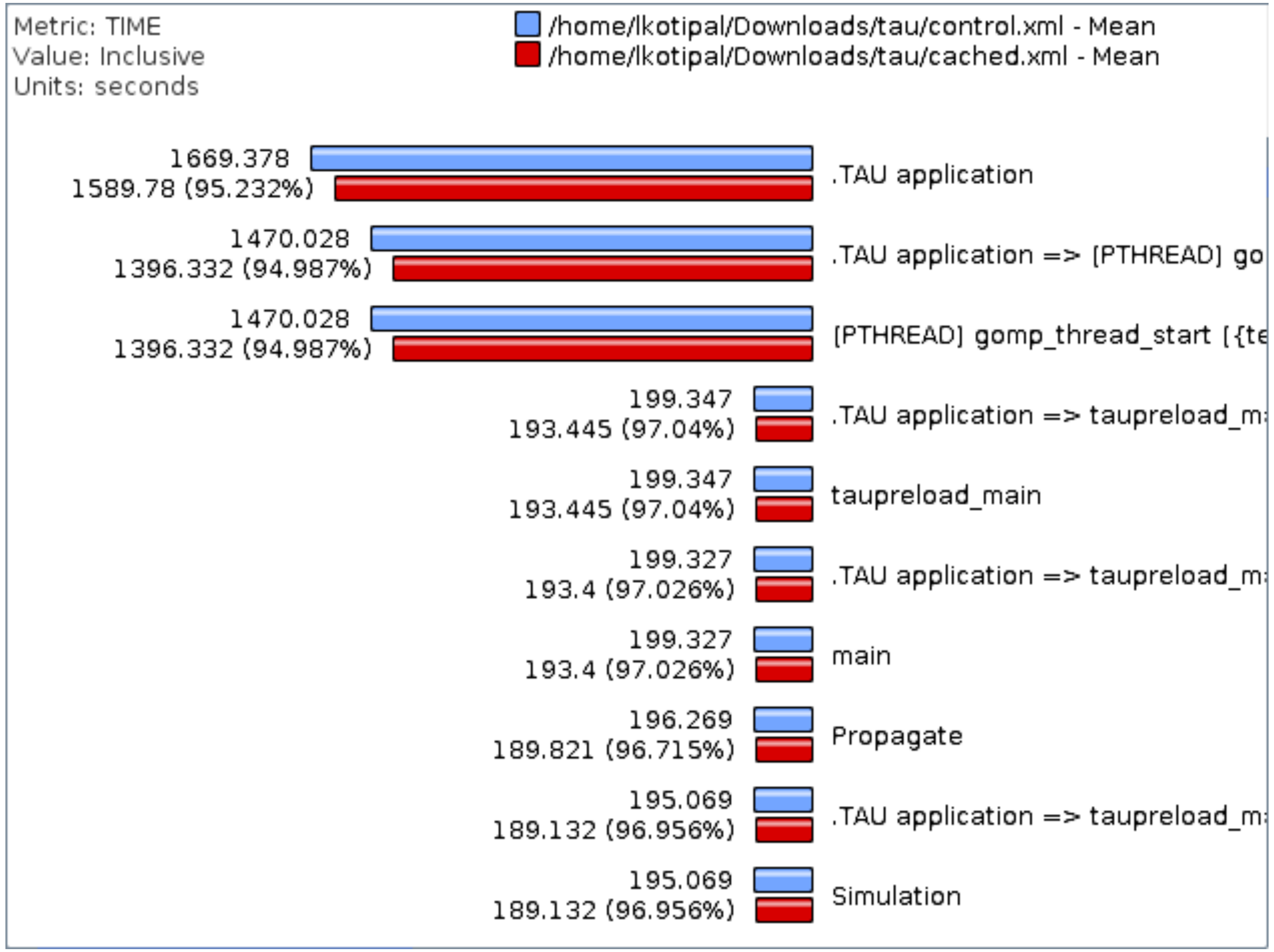
*

By limiting the batch size to 8 elements and adjusting number of threads, the imbalance was alleviated

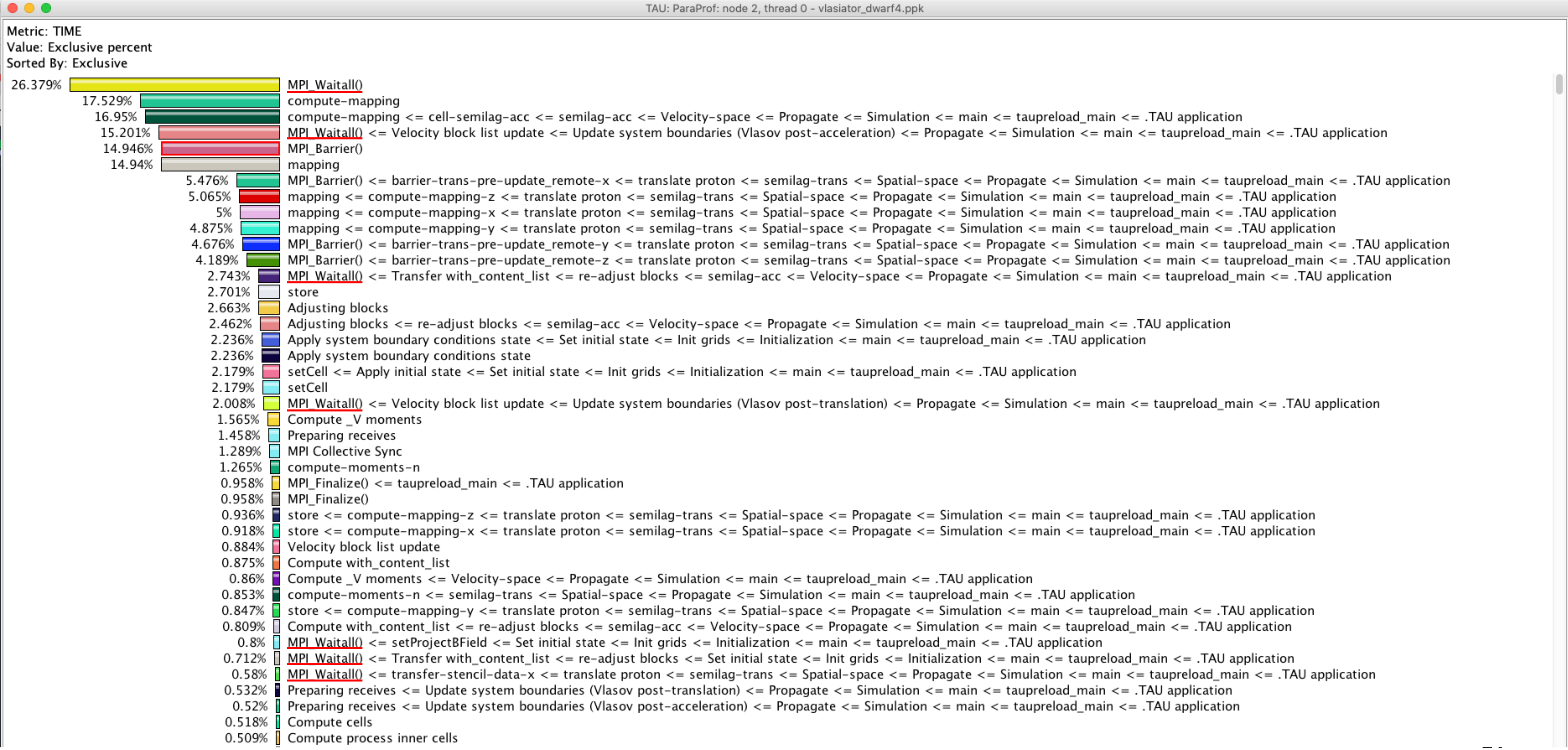
* Pthread for *ofi_uffd_handler*

Improvement from Neighbor Caching

- Problem
  - EBS revealed inefficient function in grid processing
  - Searching for spatial neighbors of simulation cells
    get_face_neighbors_of()
  - Iterated over nearby cells in a 3-cell-wide stencil multiple times
  - 1000x per cell worst case
- Solution
  - Cache neighbor results
  - Performance increase of 5%



Significant MPI Waiting Suggests Imbalance



Conclusion

- Implementation of a profiling interface
  - Integration in profiling tools TAU and APEX
  - Hierarchical performance information
  - Profiling and tracing, taskgraph, ...
- Performance engineering for Vlasiator
  - Time spent in computation steps that have a physical meaning
  - High-level timers carry this semantics
- Optimization and further developments
  - Provide info on where time is spent and how
  - Guide performance optimization
  - Leads for GPU port