# **Integration of Modern HPC Performance Analysis in Vlasiator for Sustained Exascale**

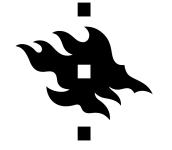
Camille Coti<sup>1</sup>, Yann Pfau-Kempf<sup>3</sup>, Markus Battarbee<sup>3</sup>, Urs Ganse<sup>3</sup> Sameer Shende<sup>2</sup>, Kevin Huck<sup>2</sup>, Jordi Rodriquez<sup>2</sup>, Leo Kotipalo<sup>3</sup> <u>Allen D. Malony<sup>2</sup></u>, Minna Palmroth<sup>3,4</sup>

École de Technologie Supérieure Montréal, Québec, Canada<sup>1</sup>

University of Oregon Eugene, Oregon, USA<sup>2</sup> University of Helsinki Helsinki, Finland<sup>3</sup>







- Finnish Meteorological Institute Helsinki, Finland<sup>4</sup>



# Motivation

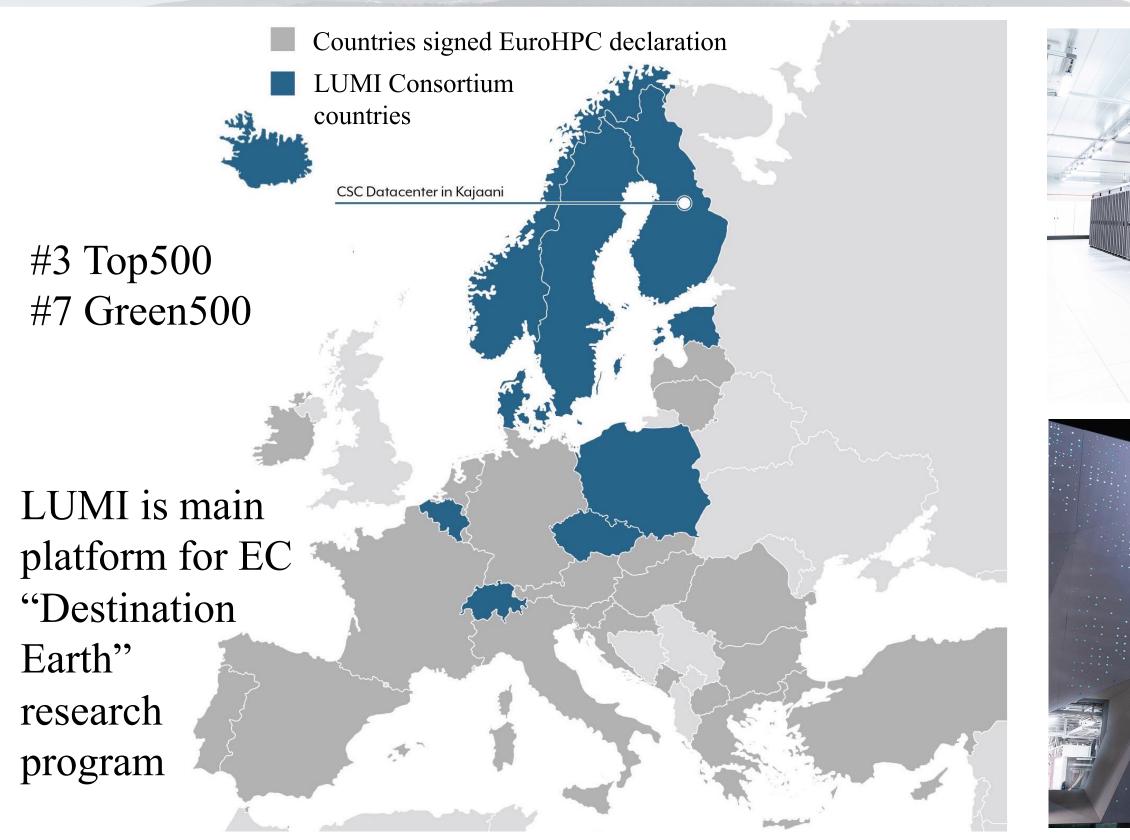
- Heterogeneous computing with accelerated-node hardware is delivering extreme computational power
- □ Productive utilization of exascale systems for real-world workloads will require sustained application performance □ Heterogeneity raises application development concerns • Codes need to be (re-)developed 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 FOUNDATION • 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 CSC 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)



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

## VL/SI/J@R

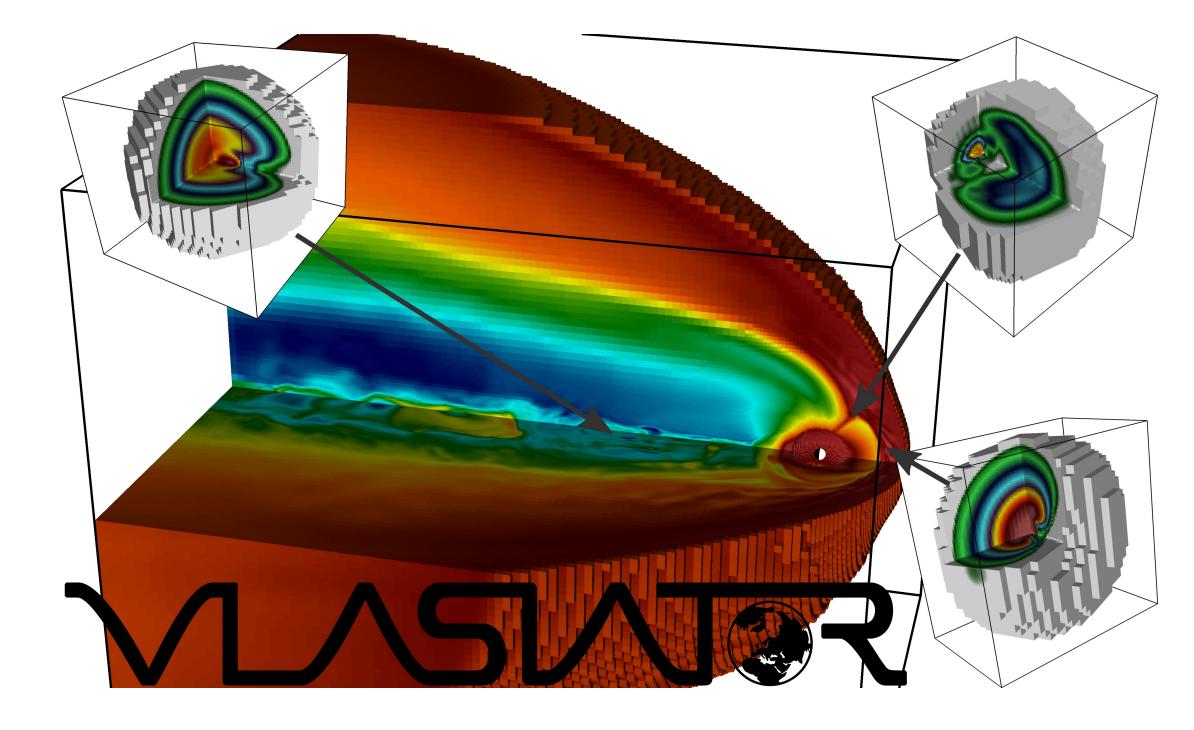
e velocity) , uniform grid neutrality on pressure gradient

# 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 phasespace density distribution in the three velocity dimensions at selected locations

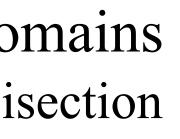




6

# Vlasiator Code Development

- $\Box$  Written in C++17 with hybrid parallelization
  - CPU SIMD vectorization
  - OpenMP threading
  - o 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



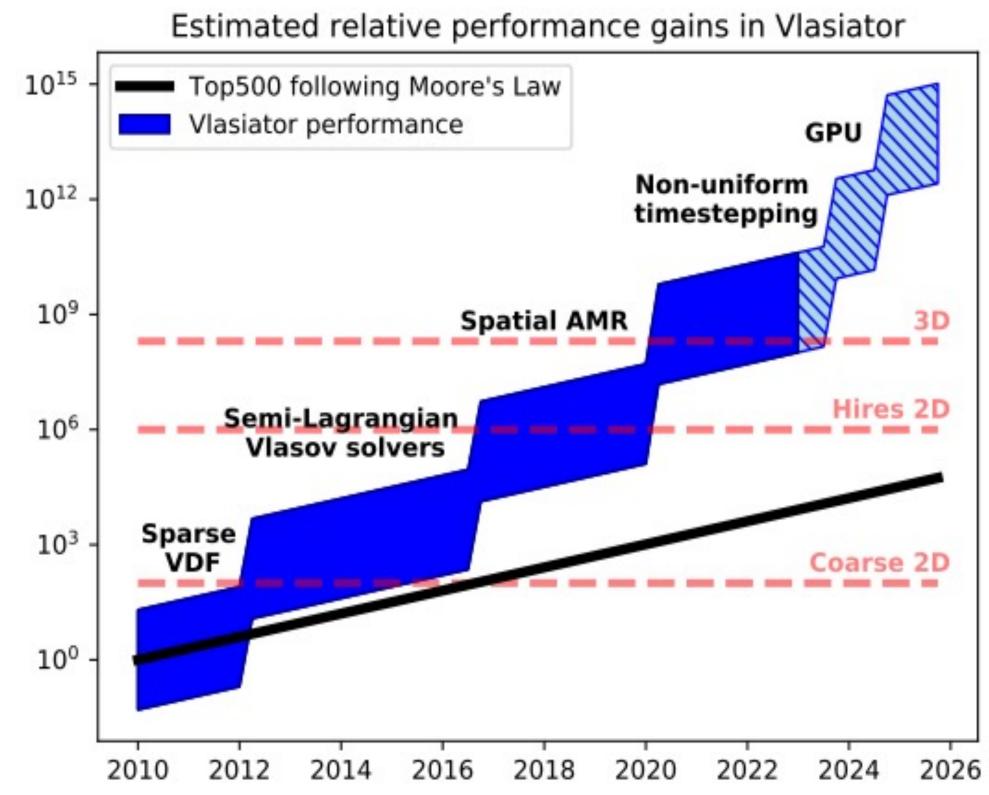
omain h spatial cell development

# **Relative Performance of Vlasiator Evolution**

Relative performance of Vlasiator normalized to the performance in the beginning of the project

## Global target simulation

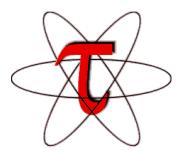
- Parts of the solar wind
- Large part of the magnetosphere
- □ Blue areas give the relative performance of Vlasiator compared to TOP500
- Dark blue is past, dashed future



CUG 2023

# TAU Project at the University of Oregon

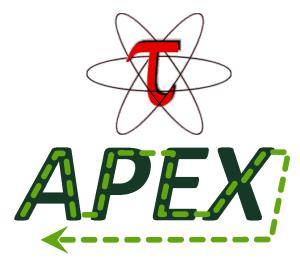
- □ Research and development effort spanning 30+ years
- Focus on parallel performance problems and technologies
- □ Performance problem solving framework for HPC research
  - Integrated, scalable, flexible, portable
  - Target all parallel programming / execution paradigms
- □ Integrated performance toolkit (TAU Performance System<sup>®</sup>)
  - Multi-level performance instrumentation
  - Flexible and configurable performance measurement
  - Widely-ported performance profiling / tracing system Ο
  - 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

## s sis support



# 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  $\mu$ sec)
- □ 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() functio
2	Initialization	Grid and solver setup
2	report-memory-consumption	Function reporting node r
2	Simulation	Main loop
3	ΙΟ	Data and bookkeeping IO
3	Propagate	Actual plasma and electro
4	Spatial-space	Position space advection
4	Update system boundaries (Vlasov post-translation)	Post-advection update of
4	Compute interp moments	Interpolation of density/ve
4	Propagate Fields	Electric and magnetic fiel
4	Velocity-space	Velocity space advection
4	Update system boundaries (Vlasov post-acceleration)	Post-acceleration update of
4	ionosphere-solve	Ionospheric potential update
4	Other	Remaining, non-instrumer
3	compute-timestep	Determination of time ste
3	Balancing load	Rebalancing of the compu
3	Other	Remaining, non-instrumer
1	Other	Remaining, non-instrumen

n

memory usage

) operations omagnetic field propagation

boundary cells

elocity/pressure between advection and acceleration ld update

a.k.a. acceleration

of boundary cells

ate

nted sections in level 3 region "Propagate" ep limits, update of time step if necessary utational load across MPI domains nted sections in level 2 region "Simulation" nted sections in level 1 region "main"

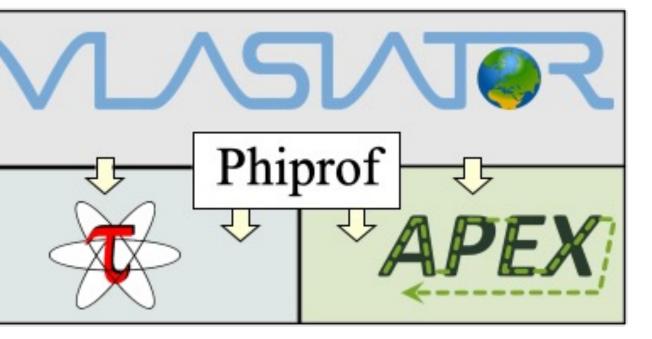
# So, what's wrong with Phiprof?

## □ 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)
  - o Run Vlasiator with TAU or/and APEX (without recompiling!)
    o Voilà !

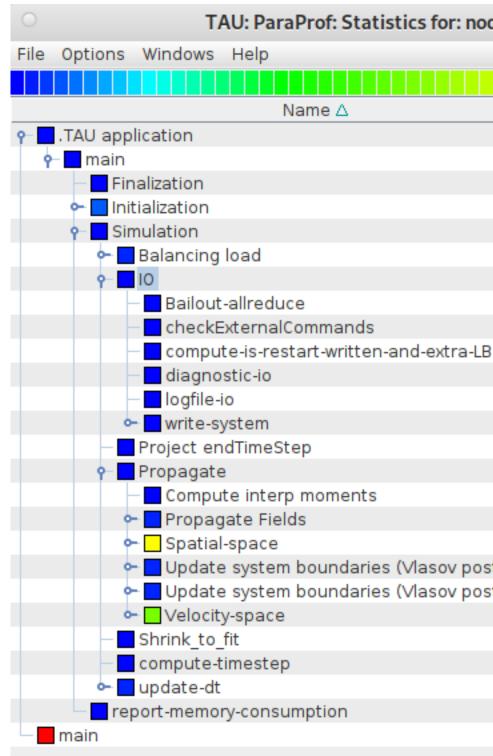
# Integrating TAU and APEX with Vlasiator

- □ 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
  - Ο...



# 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





## TAU: ParaProf: Statistics for: node 1, thread 0 - Flowthrough\_amr\_callpath.ppk

	Exclusive TIME	Inclusive TIME	Calls	Child Calls
	0.559	372.984	1	1
	0	372.425	1	4
	0	0	1	0
	0.001	16.578	1	19
	0.009	355.843	1	439
	1.189	6.623	10	80
	0.002	0.652	107	431
	0.547	0.547	107	0
	0	0	107	0
3	0.001	0.001	107	0
	0	0	1	0
	0.002	0.002	107	0
	0	0.099	2	4
	0.001	0.001	101	0
	0.005	341.504	101	707
	0.009	0.009	202	0
	0.014	6.267	101	1,010
	0.001	185.825	101	101
st-acceleration)	0.785	6.344	101	808
st-translation)	0.188	6.314	101	808
	0.003	136.74	101	202
	0.003	0.003	10	0
	0.774	0.774	105	0
	0	6.277	5	10
	0.003	0.003	1	0
	0	372.425	1	4

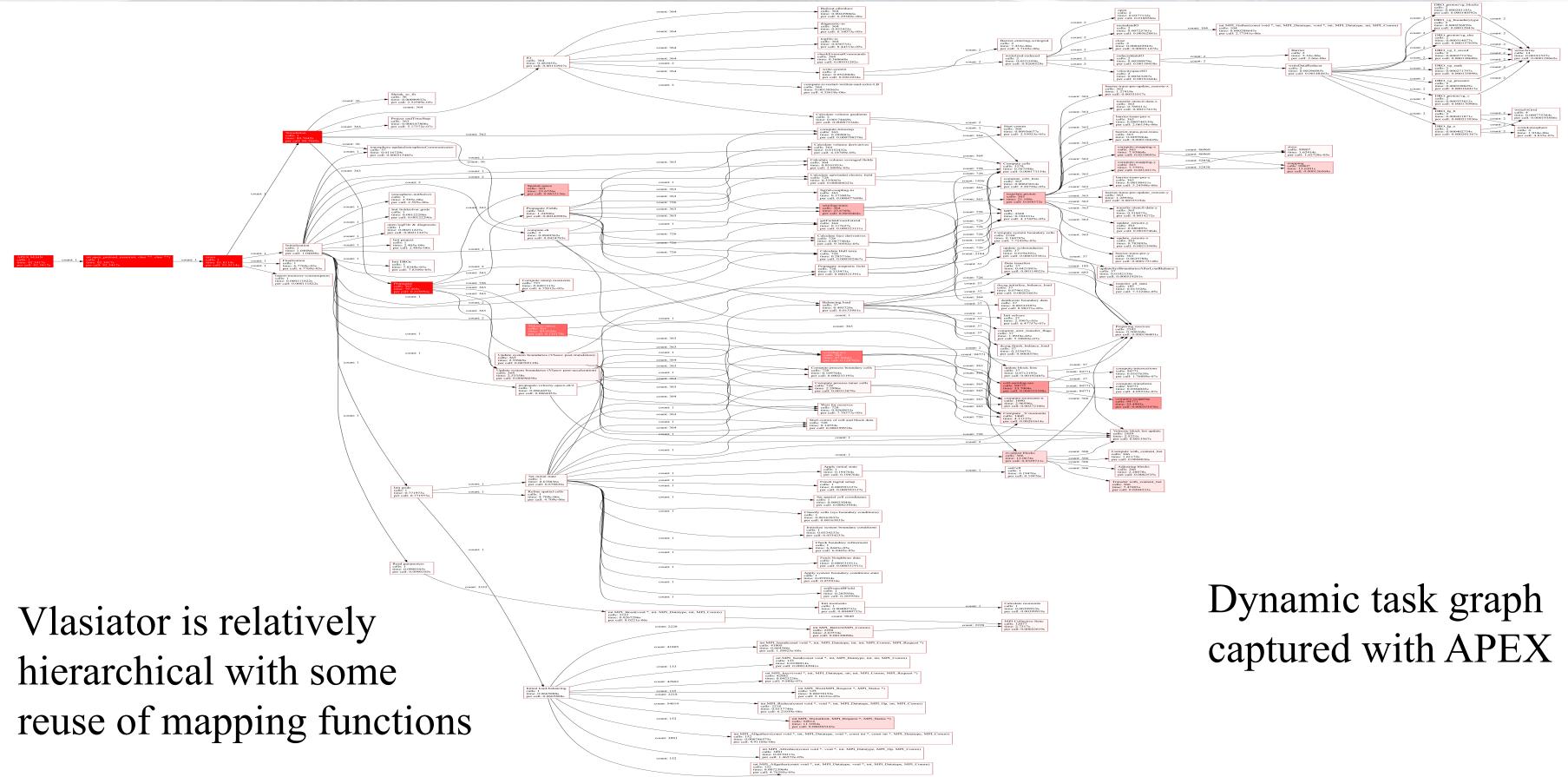
 $\bigtriangledown$   $\land$   $\times$ 

# **Tracing Support for Vlasiator Events**

●●●	trace_events.0.jsor ui.perfetto.dev	n.gz (9 MB) × + /#!/viewer?local_cach	e_key=00000000-00	000-0000-9a0b	)-e91be7f4ec1c	Trac	e visua	lized wit	h Perfet	to		~ ► □ � :
=				Q s	learch							<b>P</b>
		0 s	5 s	I I I	10 s	15 s	1 1 1	20 s	25 s	30 s	35 s	40 s
1677195669.5 s +		0 s	5 s		10 s	15 s		20 s	25 s	30 s	35 s	40 s
× =												
Process 0 0												
				· · · · · · · · · · · · · · · · · · ·				PEX MAIN _main(int, char**, char**)				
			tialization					main Sir	mulation			
			it grids et initial state	Pr. Pr. Pr. F	Pr., Pr., Pr., Pr., SSSSS	Pr Pro Pr Pr	Pro Pr Pr Pr	Pr Pro Pro Pro S S S S S S S	B Pro Pr Pro Pro.	Pro Pro Pro Pro Pro S S S S S S S S S S S S S S S S S S	B Pro Pro Pro Pro P S S S S S S S S S S S	ro Pro Pro S S S S S
CPU Thread 0 0		I A App	re-adjust B setPr	s s s t t t t	SSSS tttt	SSSSS tt	SSSSS tt	SSSSSS tt	SSSSS tt	SSSSSSS TTTTTTTT		SSSS ttt
			¥	קי מתי מחי מחך		, ma , ma , ma , ma ,	ה, הננה, הנה, הנה, ה	י החוי ההוי החוי החוי ה	ומיזוי, שניני, הניה		ה, הה, נוה, שה, הה, ונה	ן, שנו, שנו, שנו, ו
CPU Thread 2 2		Initialization										
CPU Thread 3 3		Init grids	Pr Pr									
CPU Thread 4 4		Set initial state	S S									
CPU Thread 5 5	A App.	re-adjust B setPr										
CPU Thread 6 6	S S	Co A V										
CPO Initead 00												
CPU Thread 7 7												
CPU Thread 8 8												
1 Minute Load aver	rage value 📈	25										
	ner recvbuf value 📈	75										
	ner sendbuf value 📈	5 100 K										
	nerv recvbuf value 📈	7.5 K									L	
	ice recybuf value 📈	7.5 K	1									
	ice sendbuf value 📈	0.5 K	1									
Bytes : MPI_Bcast v		2.5 K										
Bytes : MPI_Gather	recvbuf value 📈	0.25 K										
Bytes : MPI_Gather	sendbuf value 📈	10									1	
Bytes : MPI_Irecv va	alue 📈	50 M		الله بعله متله بعله	a dha dha dha a	والمتركب والمترك والمراك والمرا	الدخيلة عبله عبار خيا	وملتجعات بالتجار	بالمحتولية والمحتول	սիս միս միս միս միս միս	المعالمعالمعالمعاله ما	بحبائه فالحداقه
Bytes : MPI_Isend v		75 M								ساية ساية شاية ساية ساية ساية		
Bytes : MPI_Reduce		2.5 K										
Bytes : MPI_Reduce		5										
Bytes : MPI_Send va		0										
CPU I/O Wait % value		0.25										
CPU Idle % value	~ ~	100										



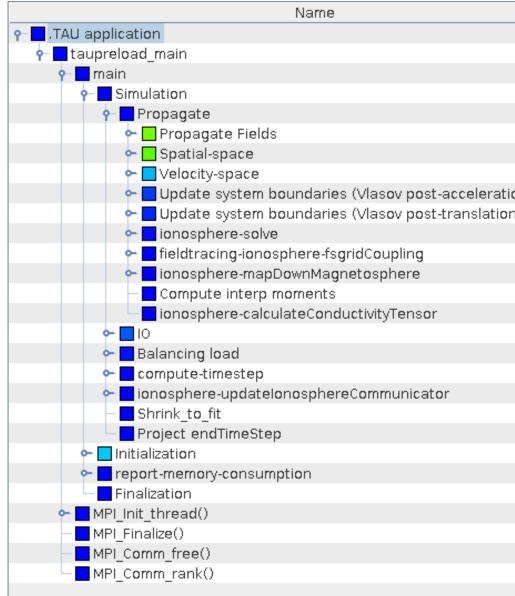
# Task Graph of Vlasiator Execution





# Magnetospheric Simulation on LUMI

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



	Exclusive TIME	Inclusive TIME 🗸	Calls	Child Calls
	0.001	1,844.43	1	1
	0.001	1,844.429	1	17
	0.002	1,842.288	1	4
	0.007	1,662.342	1	296
	0.003	1,558.669	71	513
	0.054	668.607	71	16,330
	0.001	626.189	71	71
	0.003	164.332	71	213
tion)	0.096	52.675	71	15,700
on)	0.099	34.415	71	15,753
	10.31	10.326	4	4
	0.023	2.063	4	64
	0.007	0.035	4	12
	0.013	0.013	142	0
	0.011	0.011	4	0
	0.008	80.645	72	364
	0.032	16.713	4	924
	3.071	5.146	70	70
	0	1.162	4	20
	0.001	0.001	4	0
	0	0	71	0
	0.001	179.906	1	10
	0.023	0.038	1	14
	0	0	1	0
	2.112	2.112	1	1
	0.028	0.028	1	0
	0	0	13	0
	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

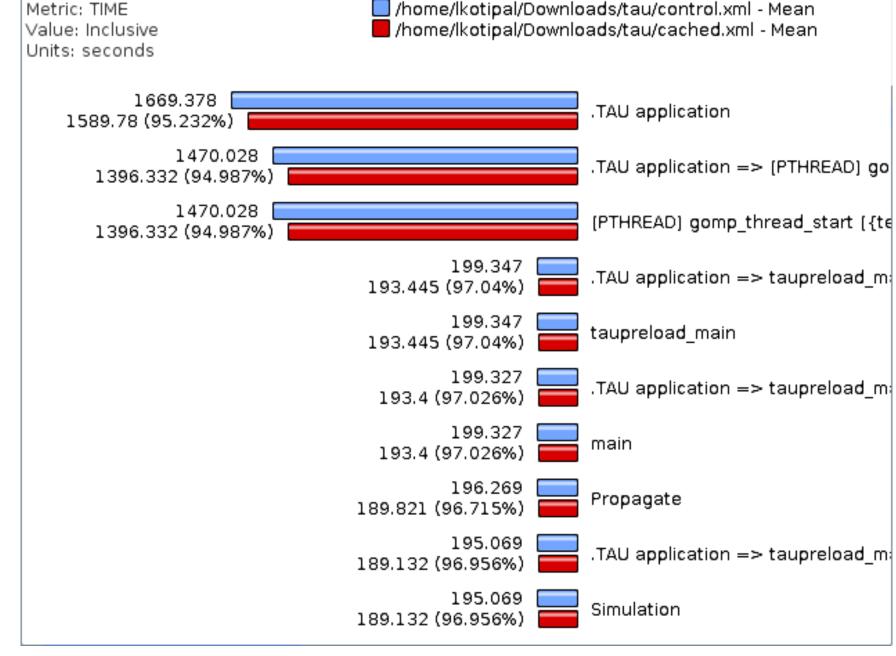
<ul> <li>Process 7</li> </ul>	=	
		.TAU application
		taupreload_main
		main Simulation
		annanana Propagate
		Spatial-space
Thread 0		semilag trans
		transfer-stencil-data-z compute-mapping-z
		MPC Varial () setup
		b comput_
TI 14		
Thread 1		
		.TAU application
Thread 2		
1110002		
		.TAU application
Thread 3		
		.TAU application
Thread 4		
Thread 5		
Thiedu 5		
		.TAU application
Thread 6		
	_	
		.TAU application
Thread 8		
		.TAU application
Thread 9		

▲ Process 7	semilagitans
Thread 0	transfer-tend
Thread 1	TAU application [PTHREAD] of_uffd_handler [{/workspace/src/github.hpe.com/hpe/hpc-shs-libfabric-netc/rpmbulld/BUILD/libfabric-1.15.0.0/prov/util/src/util_mem_monitor.c) {480, }]
Thread 2	. TAU application [PTHREAD] gomp_thread_start [(team.c) (0, 0)] [THREAD] gomp_thread_start [(team.c) (0, 0)] [T
Thread 3	.TAU application [PTHREAD] gomp_thread_start [(team c) (0,0)] [@THREAD] gomp_thread_start [(team c) (0,0)] [@TH
Thread 4	.TAU application [PTHREAD] gomp_thread_start [team c) (0, 0)] [PTHREAD] gomp_thread_start [team c) (0, 0)]
Thread 5	.TAU application [PTHREAD] gomp_thread_start [team c) (0,0)] [PTHREAD] gomp_thread_start [team c) (0,0]

## CUG 2023

# 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%





📃 /home/lkotipal/Downloads/tau/control.xml - Mean

# Significant MPI Waiting Suggests Imbalance

## • • • TAU: ParaProf: node 2, thread 0 - vlasiator\_dwarf4.ppk Metric: TIME Value: Exclusive percent Sorted By: Exclusive 26.379% MPI\_Waitall() 17.529% compute-mapping 16.95% compute-mapping $\leq$ cell-semilag-acc $\leq$ semilag-acc $\leq$ Velocity-space $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload main $\leq$ .TAU application <u>MPI Waitall()</u> $\leq$ Velocity block list update $\leq$ Update system boundaries (Vlasov post-acceleration) $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload\_main $\leq$ .TAU application 15.201% 14.946% MPI Barrier() 14.94% mapping 5.476% MPI Barrier() $\leq$ barrier-trans-pre-update remote-x $\leq$ translate proton $\leq$ semilag-trans $\leq$ Spatial-space $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload main $\leq$ .TAU application 5.065% mapping $\leq$ compute-mapping-z $\leq$ translate proton $\leq$ semilag-trans $\leq$ Spatial-space $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload\_main $\leq$ .TAU application 5% mapping $\leq$ compute-mapping-x $\leq$ translate proton $\leq$ semilag-trans $\leq$ Spatial-space $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload main $\leq$ .TAU application 4.875% mapping $\leq$ compute-mapping-y $\leq$ translate proton $\leq$ semilag-trans $\leq$ Spatial-space $\leq$ Propagate $\leq$ Simulation $\leq$ main $\leq$ taupreload\_main $\leq$ .TAU application 4.676% $MPL_Barrier() \le barrier-trans-pre-update_remote-y \le translate proton \le semilag-trans \le Spatial-space \le Propagate \le Simulation \le taupreload_main \le TAU application$ $MPL_Barrier() \le barrier-trans-pre-update_remote-z \le translate proton \le semilag-trans \le Spatial-space \le Propagate \le Simulation \le main \le taupreload_main \le TAU application$ 4.189% 2.743% MPI Waitall() <= Transfer with content list <= re-adjust blocks <= semilag-acc <= Velocity-space <= Propagate <= Simulation <= main <= taupreload main <= .TAU application 2.701% store Adjusting blocks 2.663% 2.462% Adjusting blocks <= re-adjust blocks <= semilag-acc <= Velocity-space <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application Apply system boundary conditions state <= Set initial state <= Init grids <= Initialization <= main <= taupreload\_main <= .TAU application 2.236% 2.236% Apply system boundary conditions state 2.179% setCell <= Apply initial state <= Set initial state <= Initialization <= main <= taupreload main <= .TAU application 2.179% setCell MPL Waitall() <= Velocity block list update <= Update system boundaries (Vlasov post-translation) <= Propagate <= Simulation <= main <= taupreload main <= .TAU application 2.008% 1.565% Compute V moments 1.458% Preparing receives 1.289% MPI Collective Sync 1.265% compute-moments-n 0.958% MPI\_Finalize() <= taupreload\_main <= .TAU application 0.958% MPI\_Finalize() 0.936% store <= compute-mapping-z <= translate proton <= semilag-trans <= Spatial-space <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.918% store <= compute-mapping-x <= translate proton <= semilag-trans <= Spatial-space <= Propagate <= Simulation <= main <= taupreload main <= .TAU application 0.884% Velocity block list update 0.875% Compute with content list 0.86% Compute V moments <= Velocity-space <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.853% compute-moments-n <= semilag-trans <= Spatial-space <= Propagate <= Simulation <= main <= taupreload main <= .TAU application 0.847% store <= compute-mapping-y <= translate proton <= semilag-trans <= Spatial-space <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.809% Compute with content list <= re-adjust blocks <= semilag-acc <= Velocity-space <= Propagate <= Simulation <= main <= taupreload main <= .TAU application 0.8% MPI Waitall() <= setProjectBField <= Set initial state <= Initialization <= main <= taupreload main <= .TAU application 0.712% MPI Waitall() <= Transfer with content list <= re-adjust blocks <= Set initial state <= Initialization <= main <= taupreload main <= .TAU application 0.58% MPI Waitall() <= transfer-stencil-data-x <= translate proton <= semilag-trans <= Spatial-space <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.532% Preparing receives <= Update system boundaries (Vlasov post-translation) <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.52% Preparing receives <= Update system boundaries (Vlasov post-acceleration) <= Propagate <= Simulation <= main <= taupreload\_main <= .TAU application 0.518% Compute cells 0.509% Compute process inner cells

# 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





