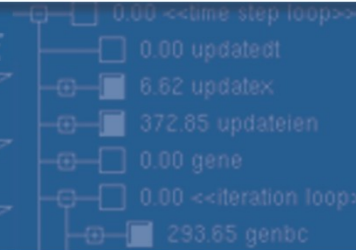


VI-HPS

SOFTWARE



FAST SOLUTIONS

- ☒ PAPI_L1_DCM
- ☒ PAPI_L1_ICM
- ☐ PAPI_L2_DCM
- ☒ PAPI_L2_ICM
- ☒ PAPI_L2_TCM
- ☐ PAPI_L2_TCM

PRODUCTIVITY

PFLOTRAN Case Study

Brian Wylie & Markus Geimer, Jülich Supercomputing Centre

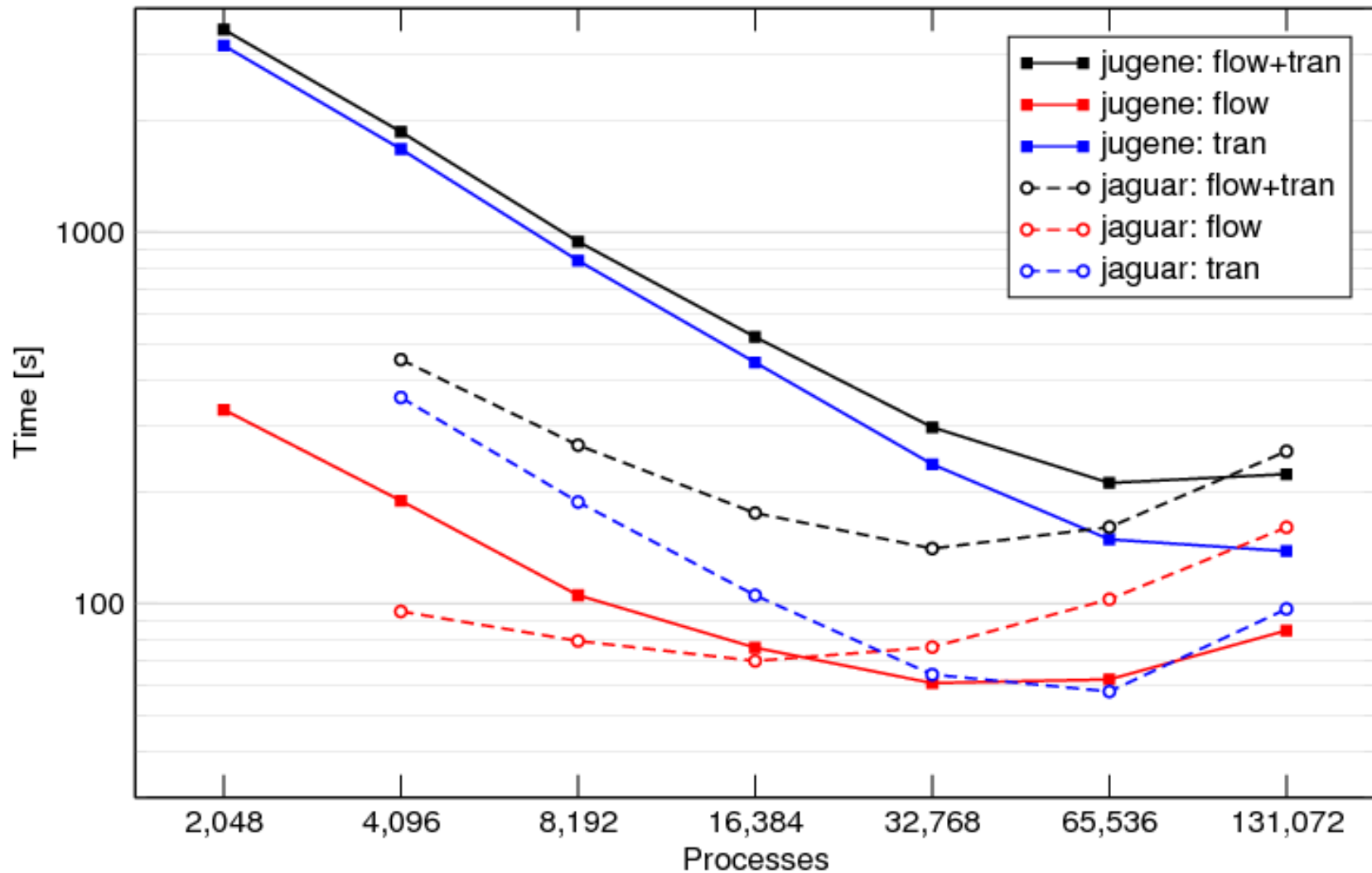
Tobias Hilbrich & Matthias Lieber, Technische Universität Dresden

Chee Wai Lee & Wyatt Spear, University of Oregon

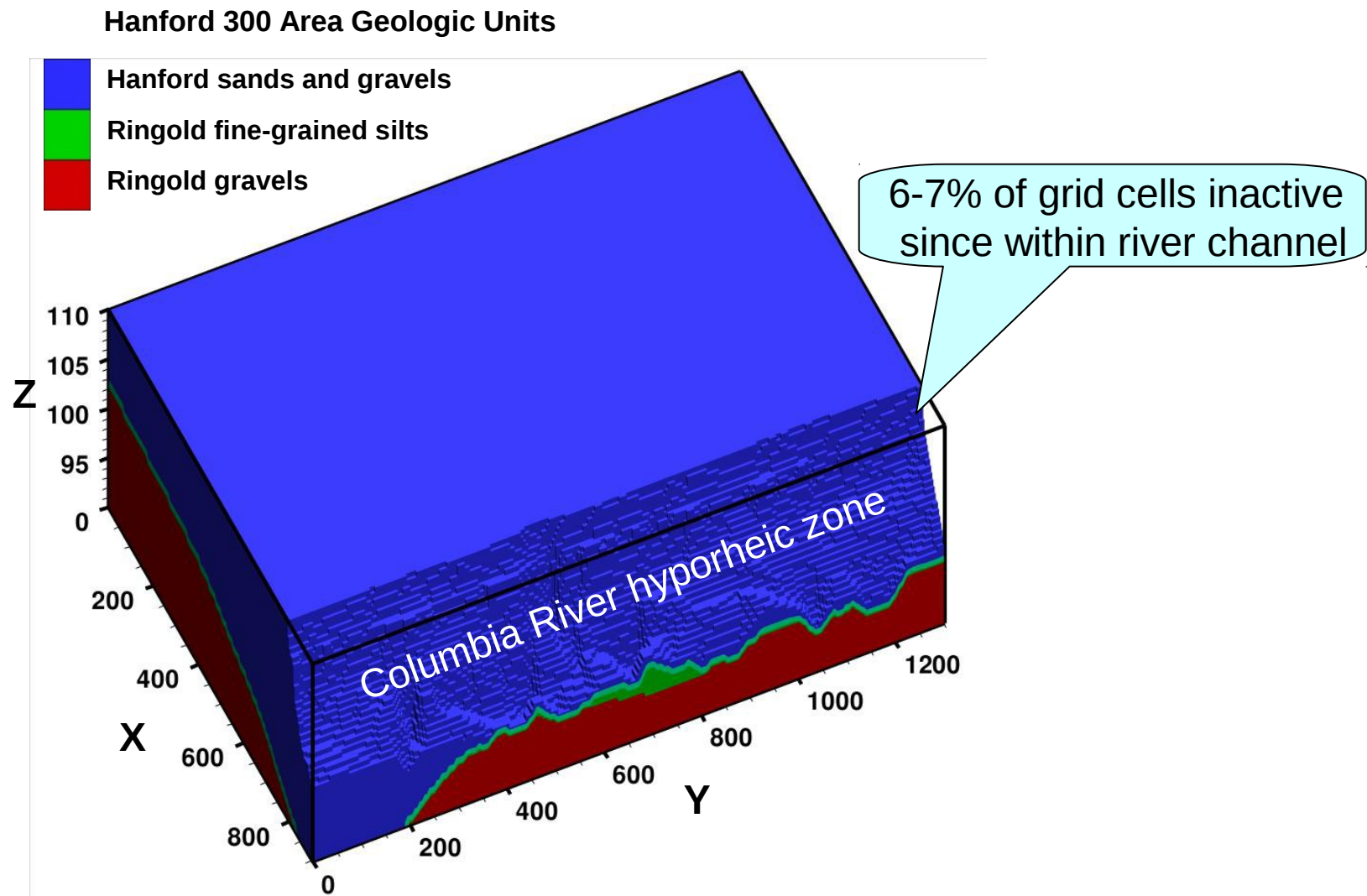
supported by the PFLOTRAN & VI-HPS development teams

- PFLOTRAN application
 - characteristics, scalability (BG/P & XT5)
- Scalasca
 - summary, summary+PAPI & trace analyses
- Vampir
 - timeline visualization, clustering, communication matrix, etc.
- TAU
 - PDT selective instrumentation
 - ParaProf experiment manager, PerfExplorer, 2D/3D profiles, histograms, callgraph, etc.

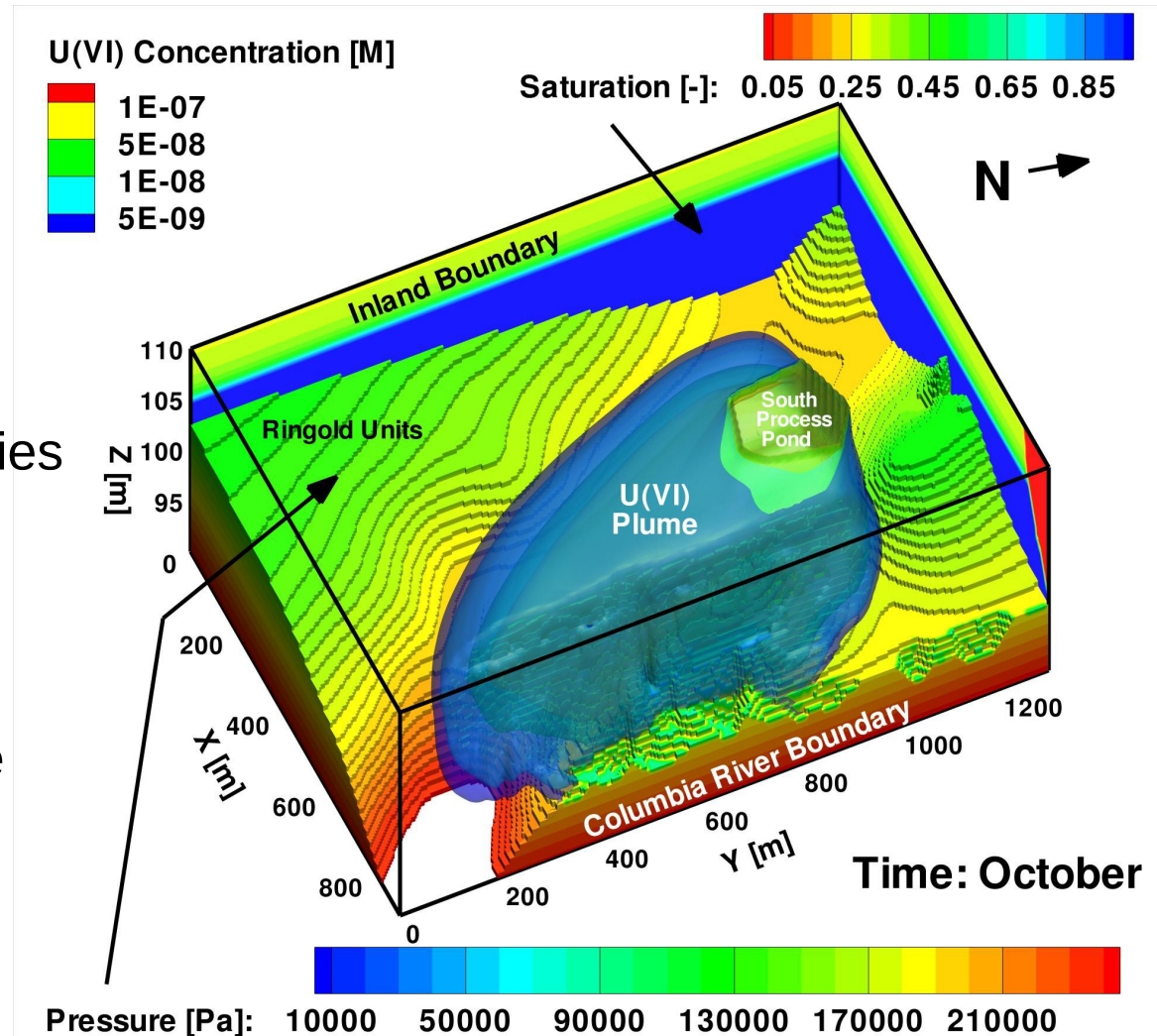
- 3D reservoir simulator developed by LANL/ORNL/PNNL
 - <http://ees.lanl.gov/source/orgs/ees/pflotran/>
 - approx. 80 thousand lines of Fortran90, combining
 - PFLOW non-isothermal, multi-phase groundwater flow solver
 - PTRAN reactive, multi-component contaminant transport solver
 - employs PETSc, LAPACK, BLAS & HDF5 I/O libraries
 - 87 PFLOTRAN source files (72 modules) + 789 PETSc
 - run with “2B” input dataset for 10 timesteps
 - most of run time in initialization phase, typically amortized
 - uses 3-dimensional (non-MPI) PETSc Cartesian grid
 - alternating FLOW & TRAN(sport) steps in each timestep
 - scaled on Jugene BG/P to 64k, Jaguar XT5 to 32k
 - TRAN(sport) step scaled much better than flow step
 - FLOW step generally faster, but crossover at larger scale



- Demonstrate performance measurement and analysis of PFLOTRAN using “2B” problem dataset and more than 10 thousand MPI processes
 - IBM BG/P (jugene.fz-juelich.de)
 - Cray XT5 (jaguar.nccs.ornl.gov)
- Challenge issued for Dagstuhl seminar 10181 on “Program development for extreme-scale computing” (3-7 May 2010)
 - two months notice with download/build/run instructions
 - <http://www.dagstuhl.de/10181>

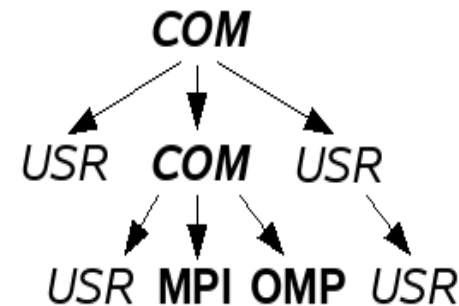


- Hanford 300 area
- 1-year simulation:
 - 900x1300x20m
 - $\Delta x/\Delta y = 5$ m
 - 1.87M grid cells
 - 15 chemical species
 - 28M DoF total
- 1-year simulation:
 - $\Delta t = 1$ hour
 - 5-10 hour runtime on Cray XT5 (4k cores)

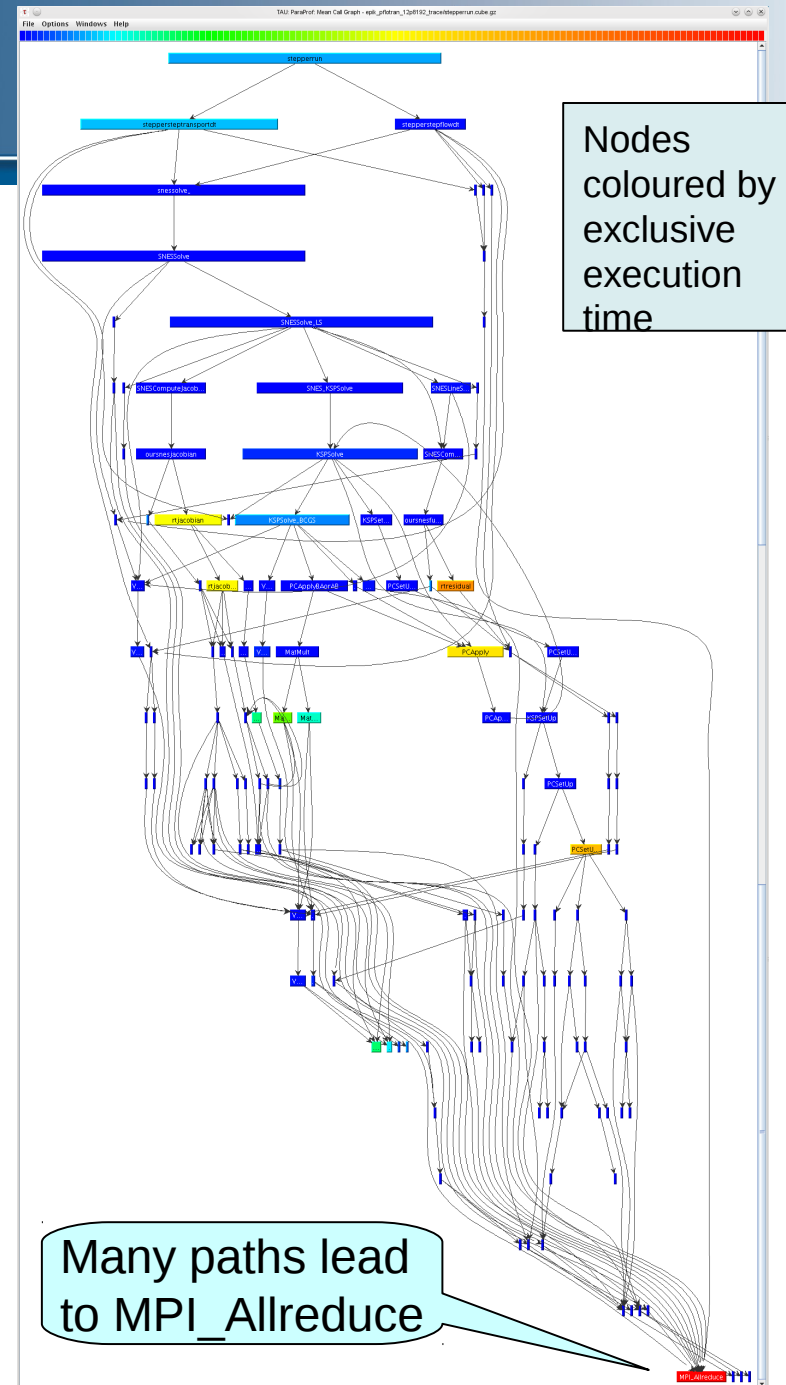
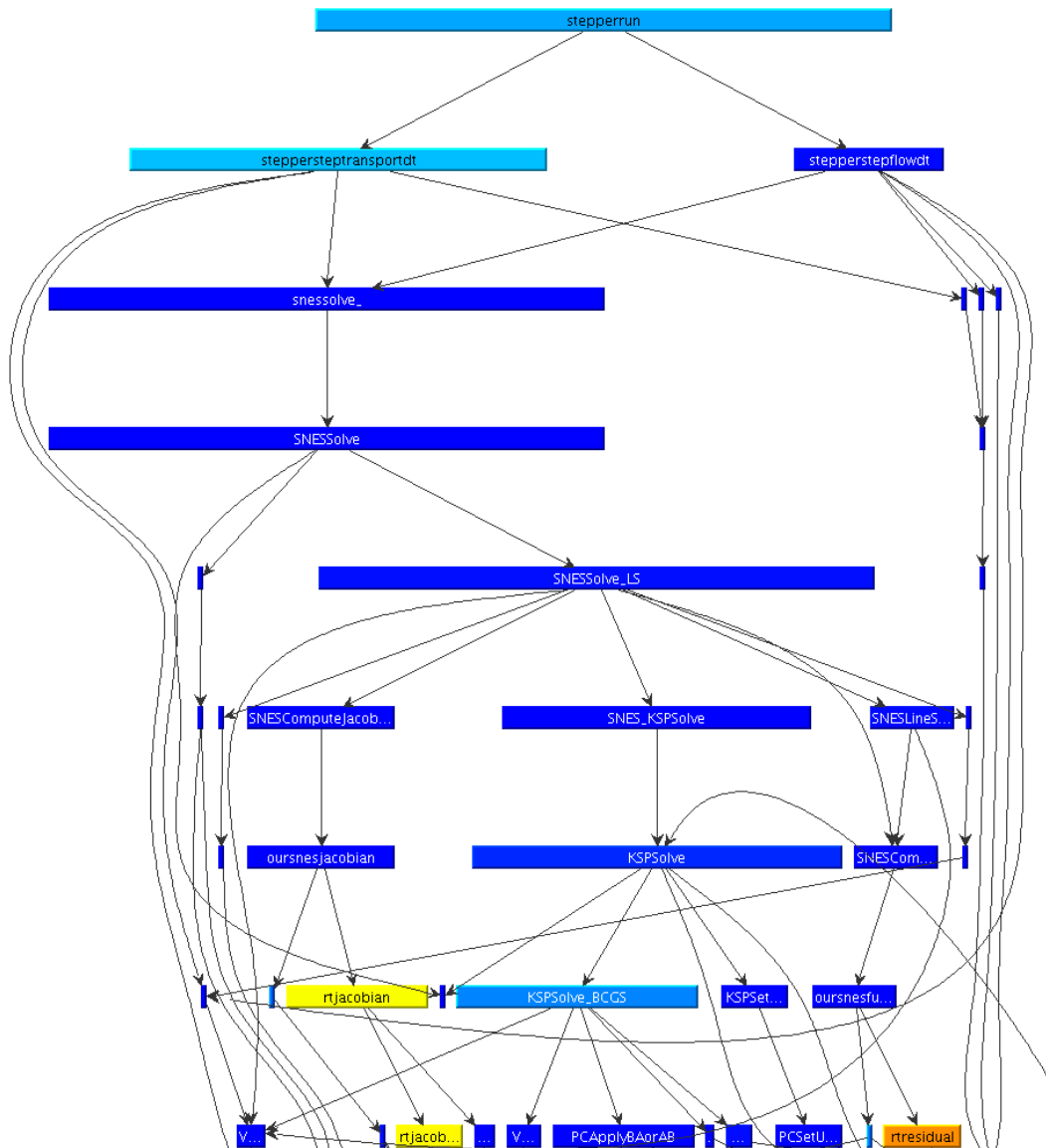


- Automatic application instrumentation
 - both PFLOTRAN application (Fortran) & PETSc library (C)
 - USR routines instrumented by IBM XL & Cray/PGI compilers
 - MPI routines via interposition of instrumented library (PMPI)
- Initial (small-scale) summary measurements used to define filter files specifying all purely computational routines
 - distinct filters required for IBM XL and PGI compilers
- Summary & trace experiments collected using filters
- Post-processing of analysis reports
 - cut to extract timestep loop; incorporation of 3D application topology
- Analysis report examination in GUI

- Determined by scoring summary expt using fully-instrumented executable
 - single timestep measurement sufficient
- 1146 PFLOTRAN+PETSc routines executed
 - plus 29 MPI library routines
 - 856 not on a callpath to MPI, purely local calculation (USR)
 - 291 on callpaths to MPI, mixed calculation & comm. (COM)
- Using measurement filter listing all USR routines
 - maximum callpath depth 22 frames
 - 1732 unique callpaths (399 in FLOW, 375 in TRAN)
 - 633 MPI callpaths (121 in FLOW, 114 in TRAN)
- Of 399 FLOW callpaths, 40 missing from TRAN, 20 only in TRAN, 14 with “similar” names
 - richardsjacobian vs rtjacobian, _MPIAIJ vs _MPIBAIJ, etc.



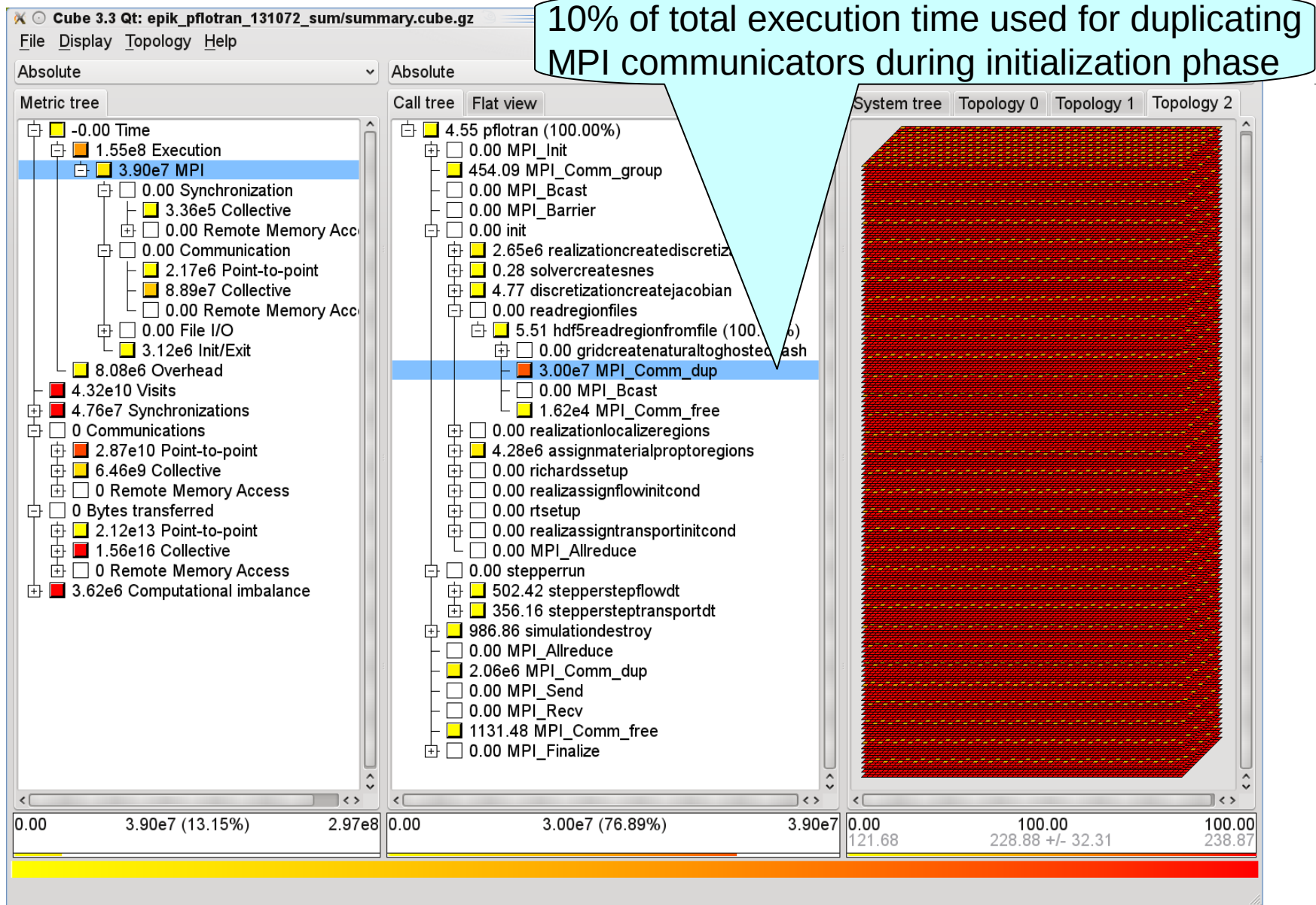
ParaProf view of stepperrun callgraph



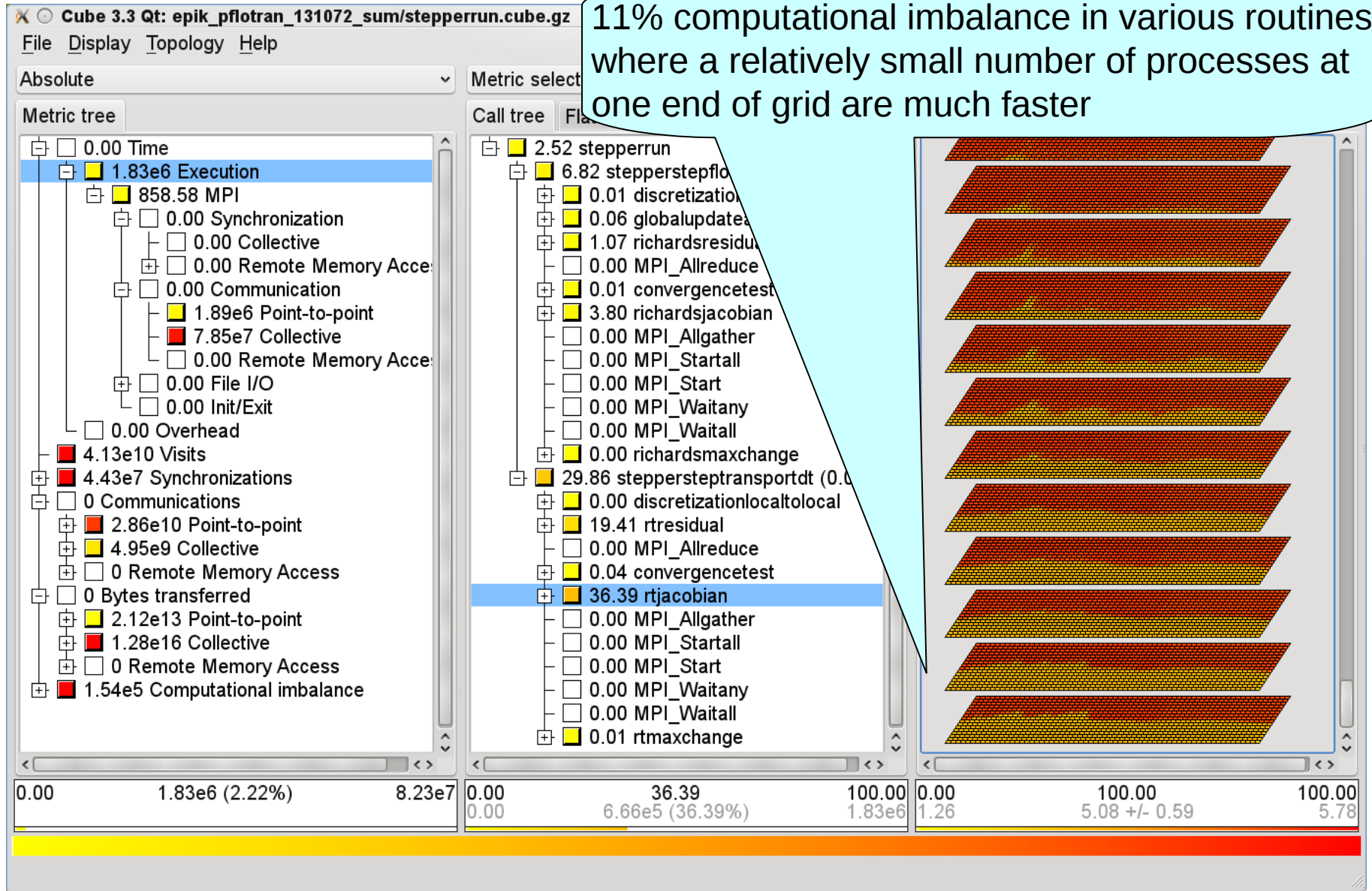
Nodes coloured by exclusive execution time

Many paths lead to MPI_Allreduce

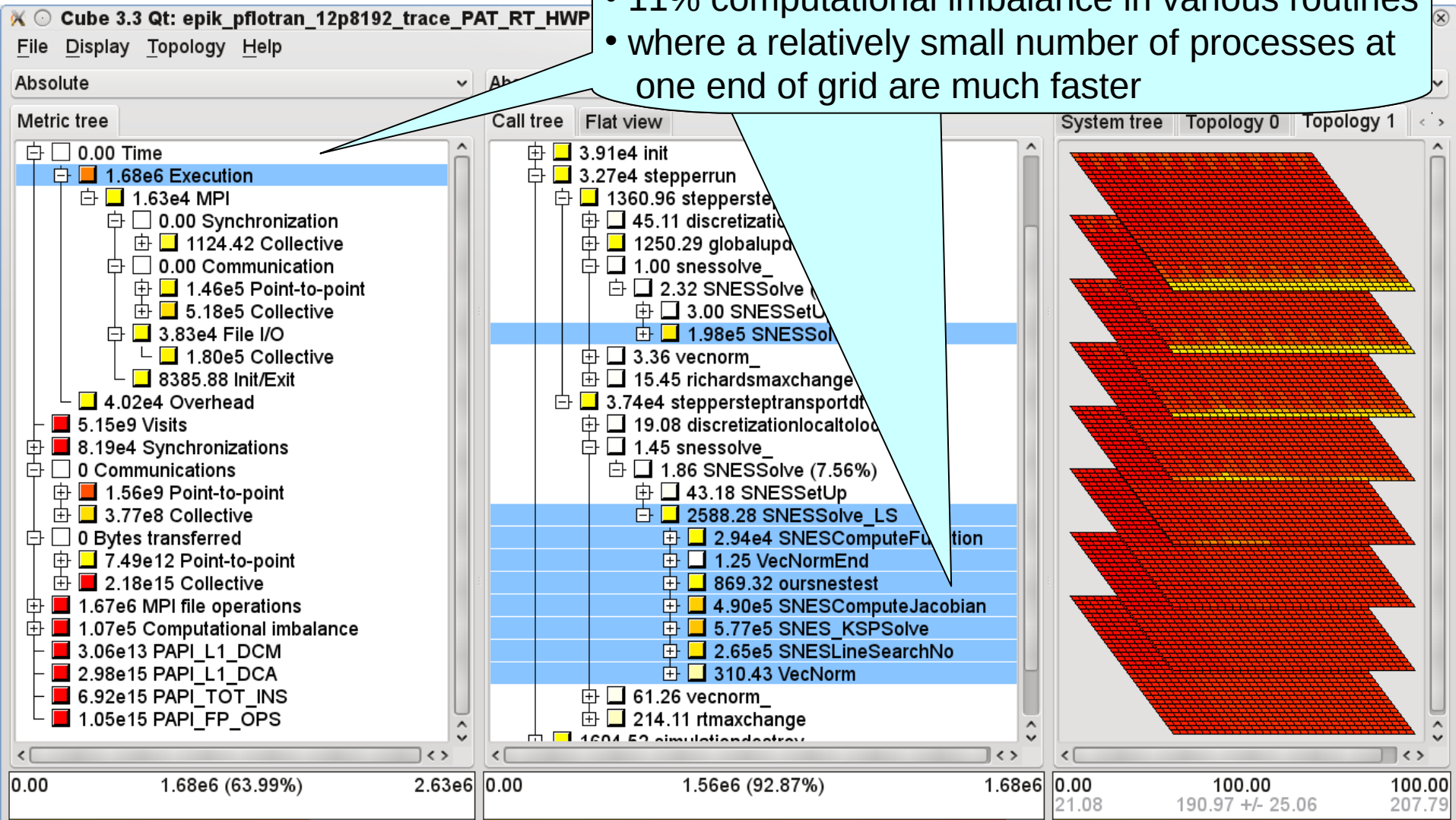
Scalasca summary analysis: entire program



Scalasca summary analysis: stepperrun

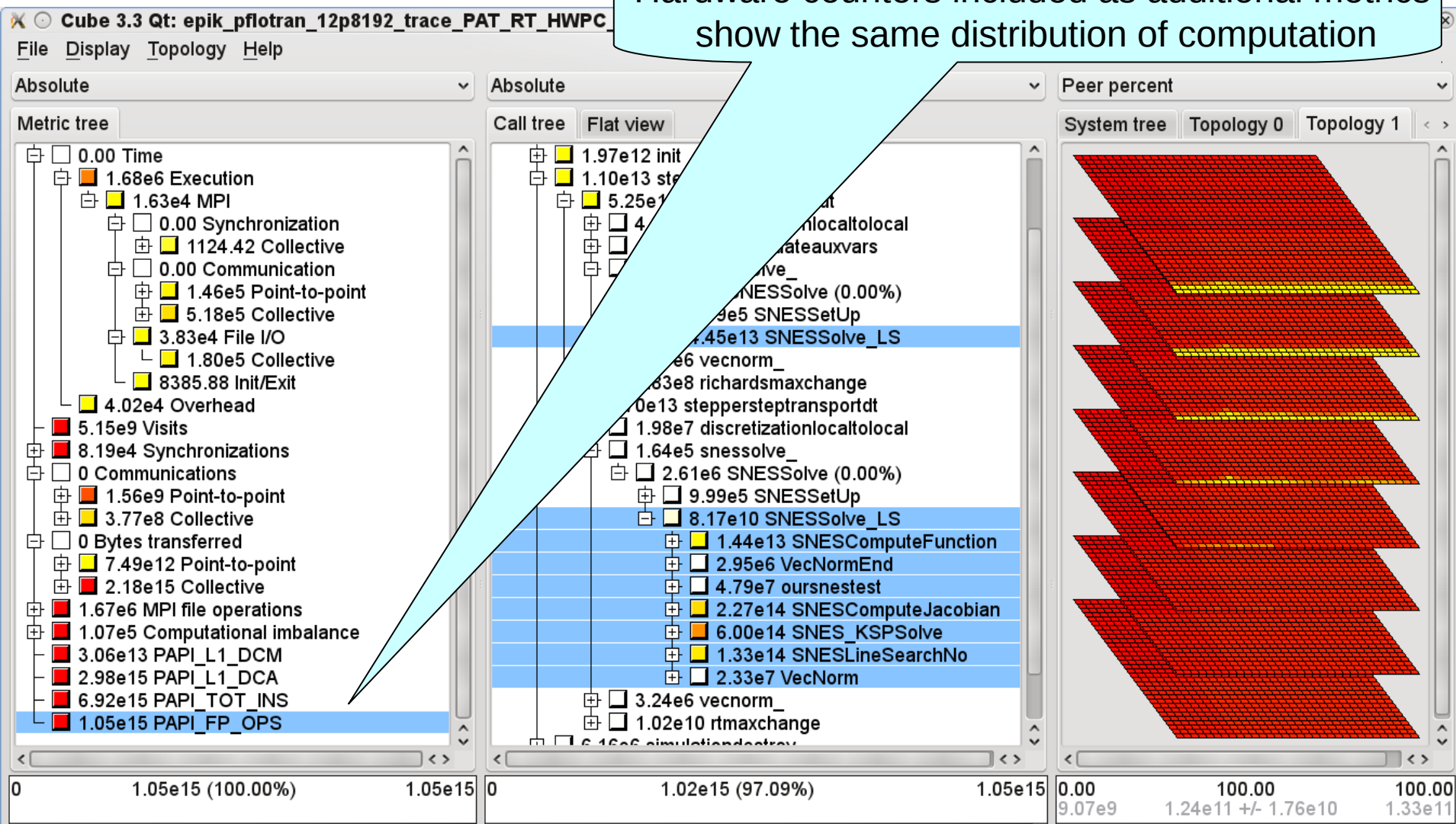


- 11% computational imbalance in various routines
- where a relatively small number of processes at one end of grid are much faster



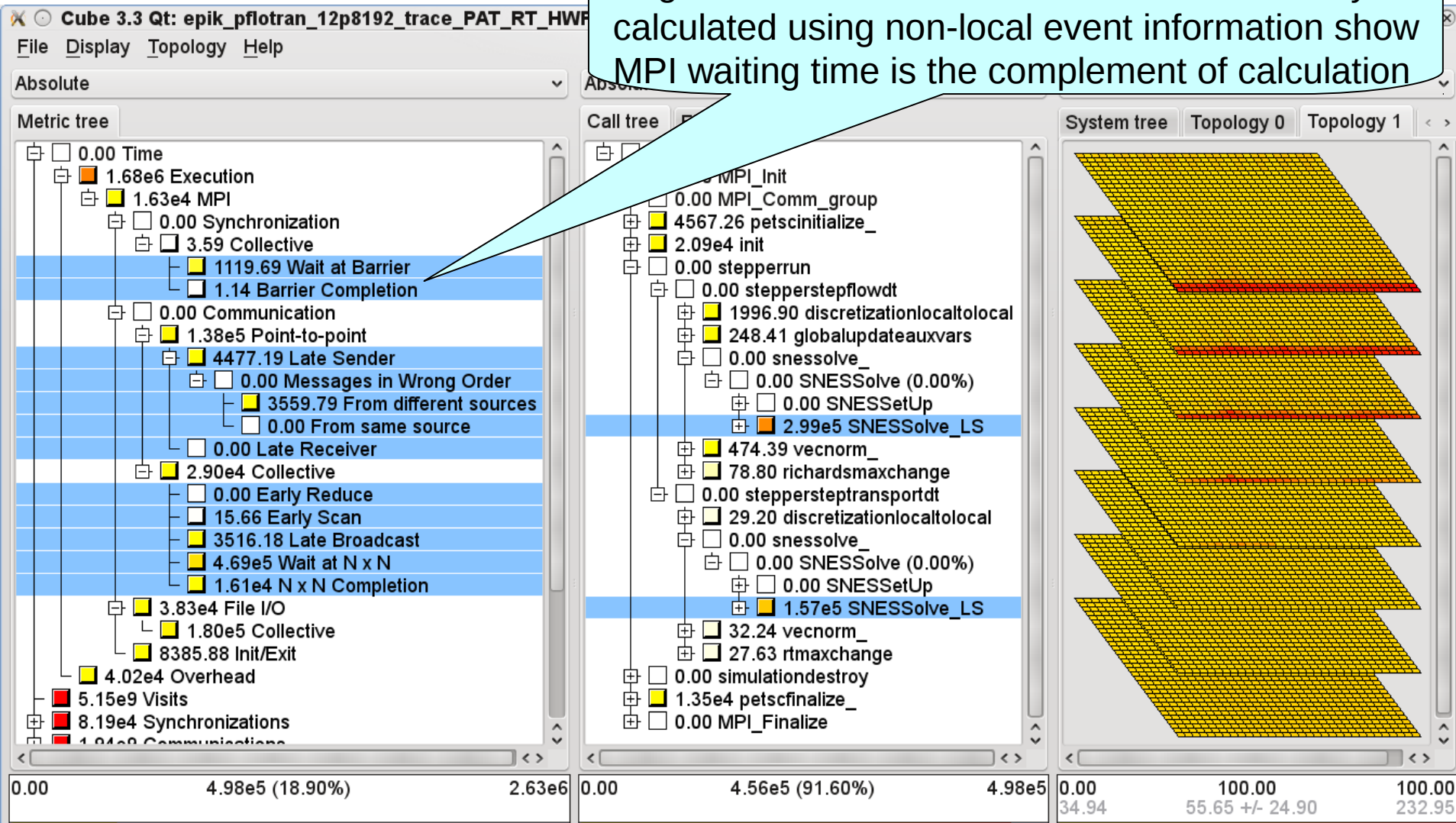
V-HPS

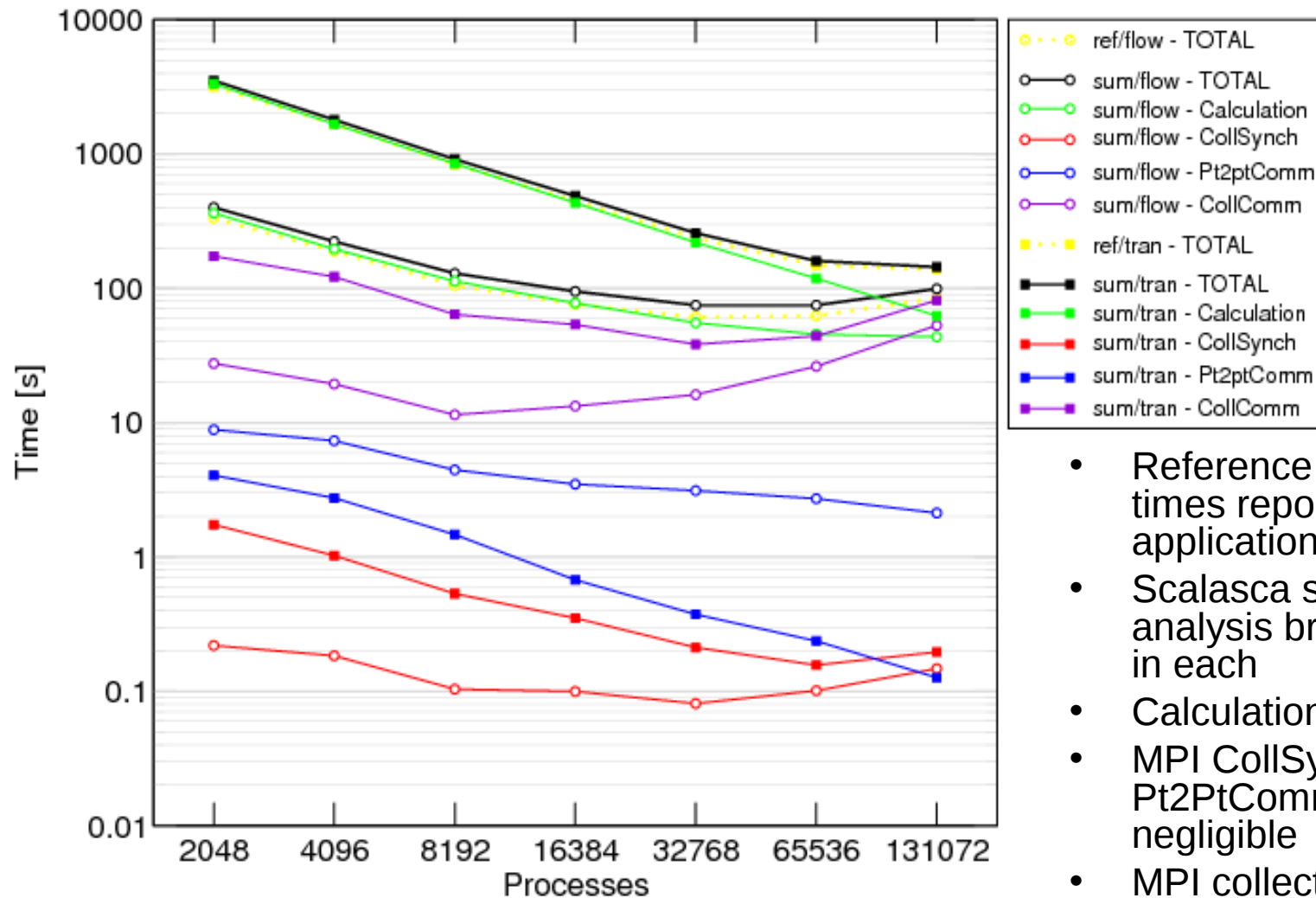
Hardware counters included as additional metrics show the same distribution of computation



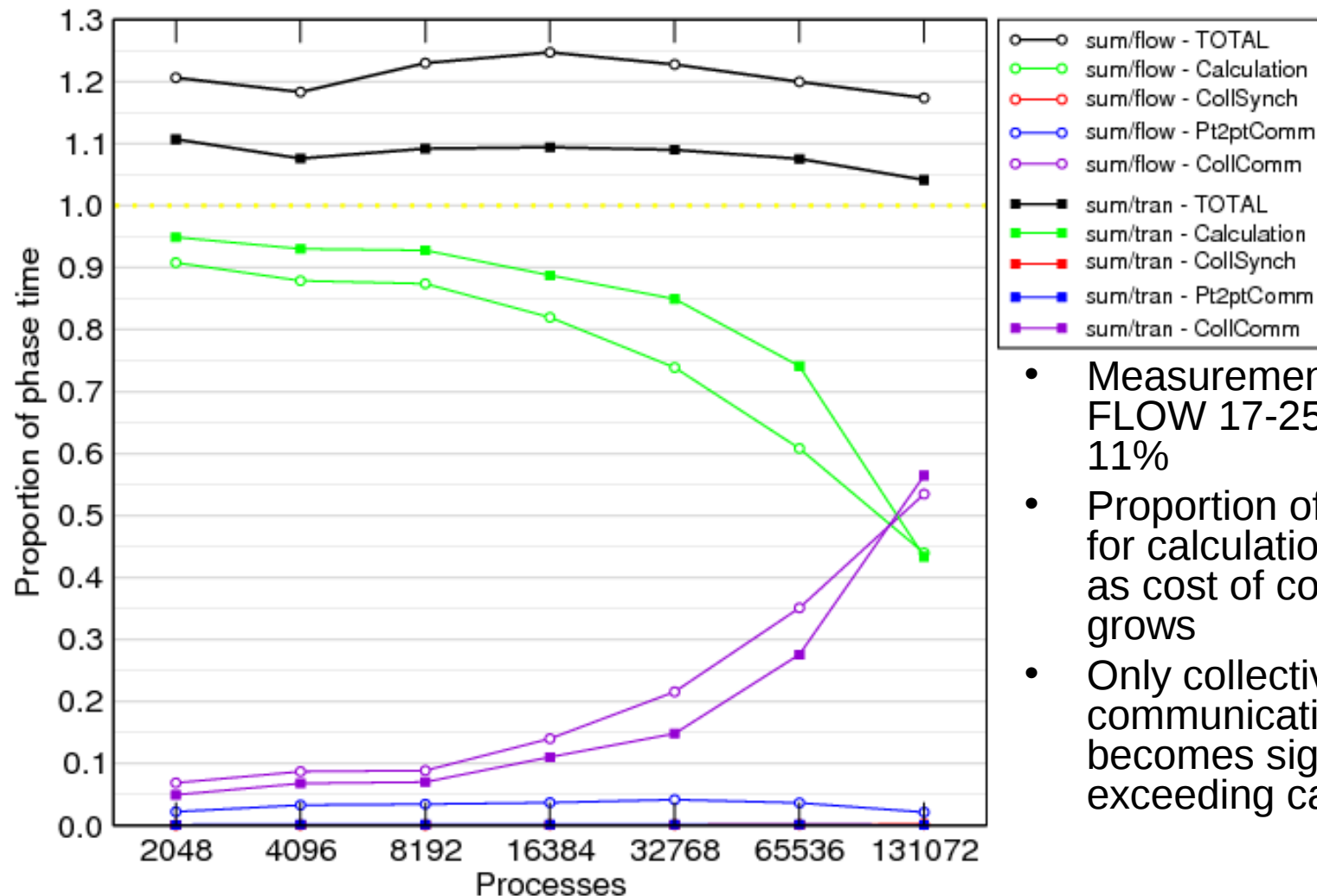
Scalasca trace analysis: MPI waiting time

Augmented metrics from automatic trace analysis calculated using non-local event information show MPI waiting time is the complement of calculation

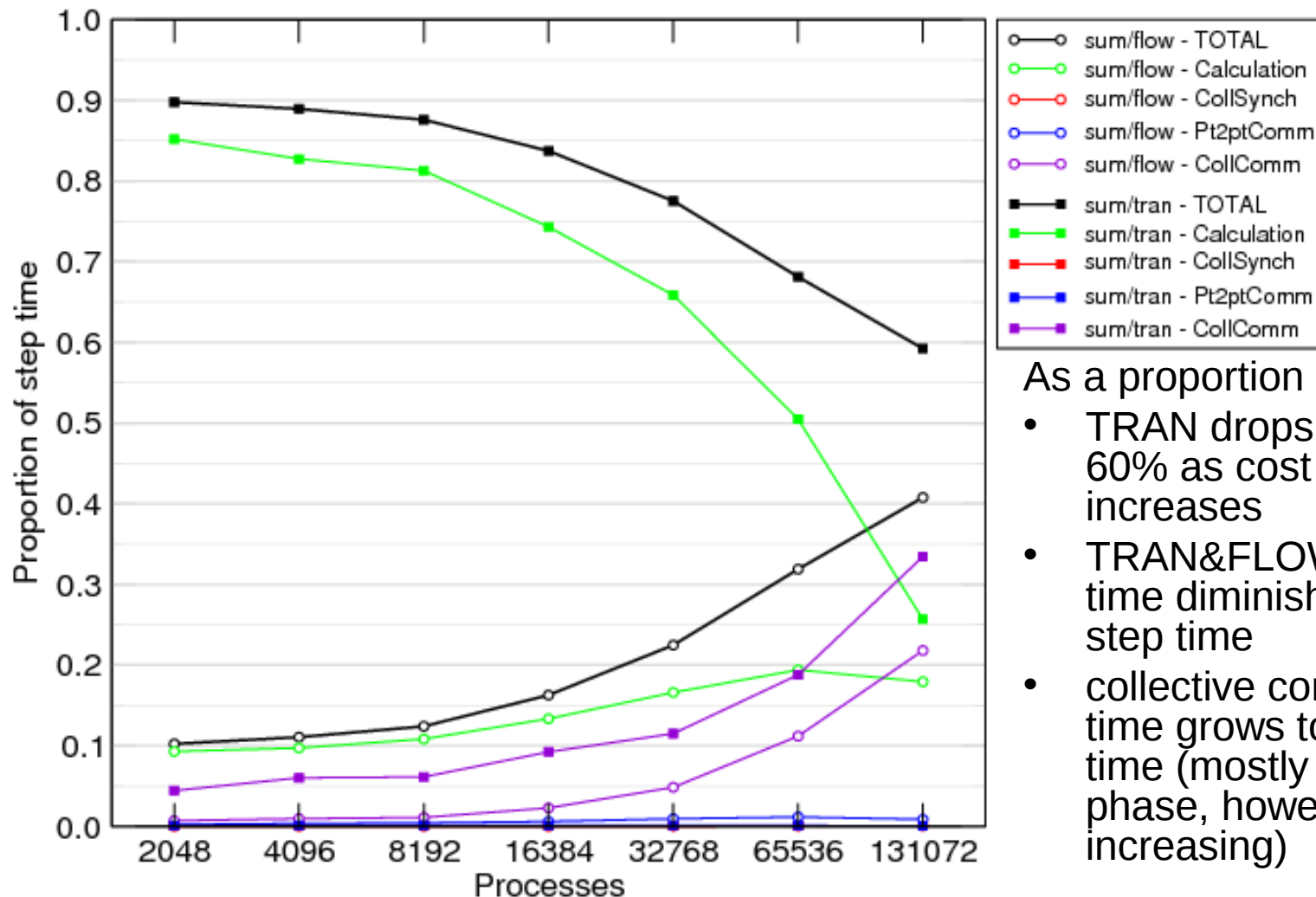




- Reference FLOW&TRAN times reported by application
- Scalasca summary analysis breakdown of time in each
- Calculation scales well
- MPI CollSynch & Pt2PtComm times are negligible
- MPI collective communication becomes a bottleneck



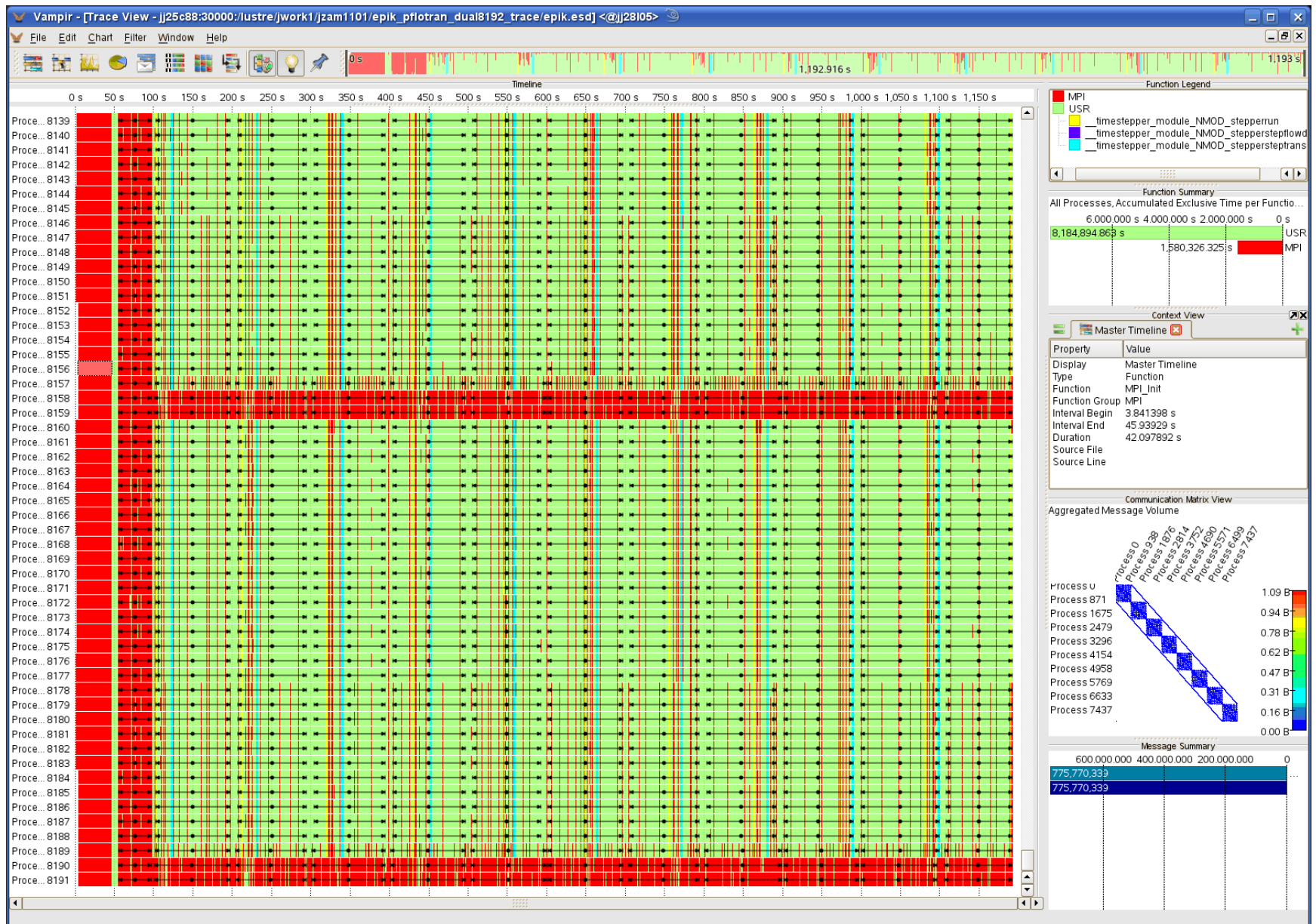
- Measurement dilation: FLOW 17-25% TRAN 4-11%
- Proportion of phase time for calculation diminishes, as cost of communication grows
- Only collective communication cost becomes significant, exceeding calculation times



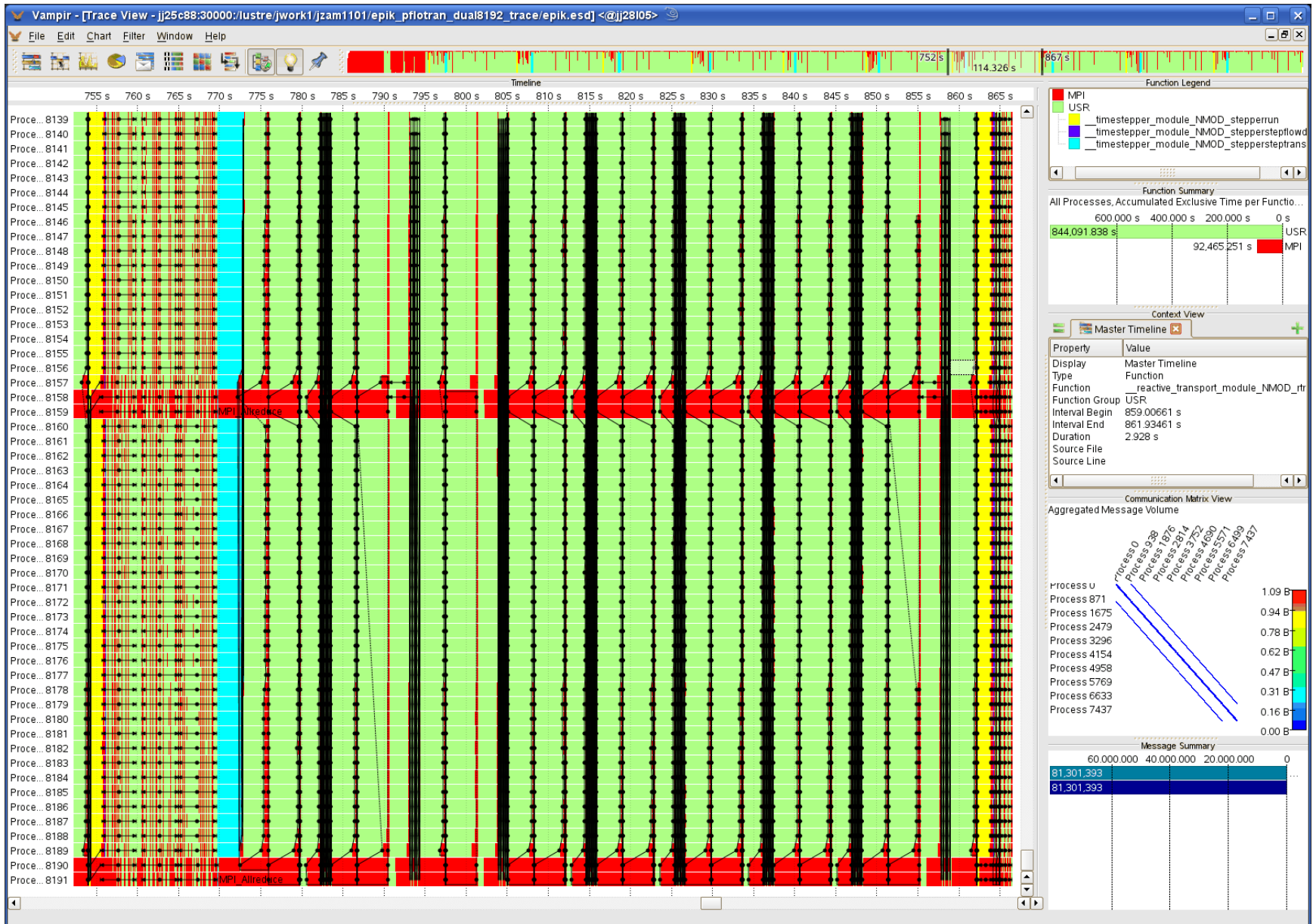
- As a proportion of step time
- TRAN drops from 90% to 60% as cost of flow increases
 - TRAN&FLOW calculation time diminish to 44% of step time
 - collective communication time grows to 55% of step time (mostly from TRAN phase, however, FLOW increasing)

- Collect traces with up to 16,384 MPI processes on BG/P for parallel analysis with Vampir7
- Use of manual instrumentation API for selective tracing
 - disable measurement during initialization phase and only enable trace of one timestep
 - reduces traces to a manageable size
- Interactive exploratory analysis of traces
 - cluster similar processes
 - zoom into time interval of interest
 - scroll/zoom timelines for processes
 - investigate communication patterns
 - examine message distributions

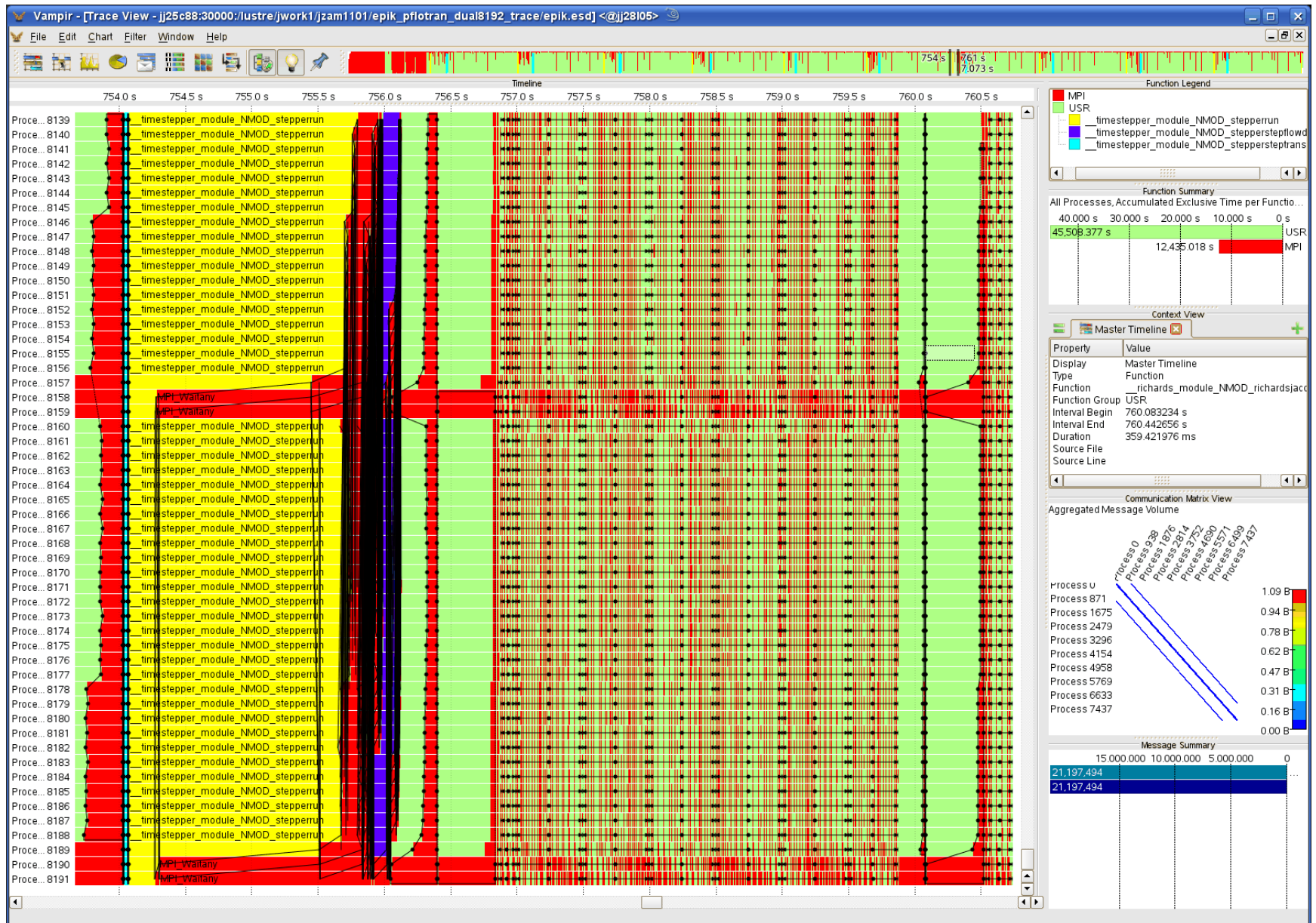
Full execution time-line



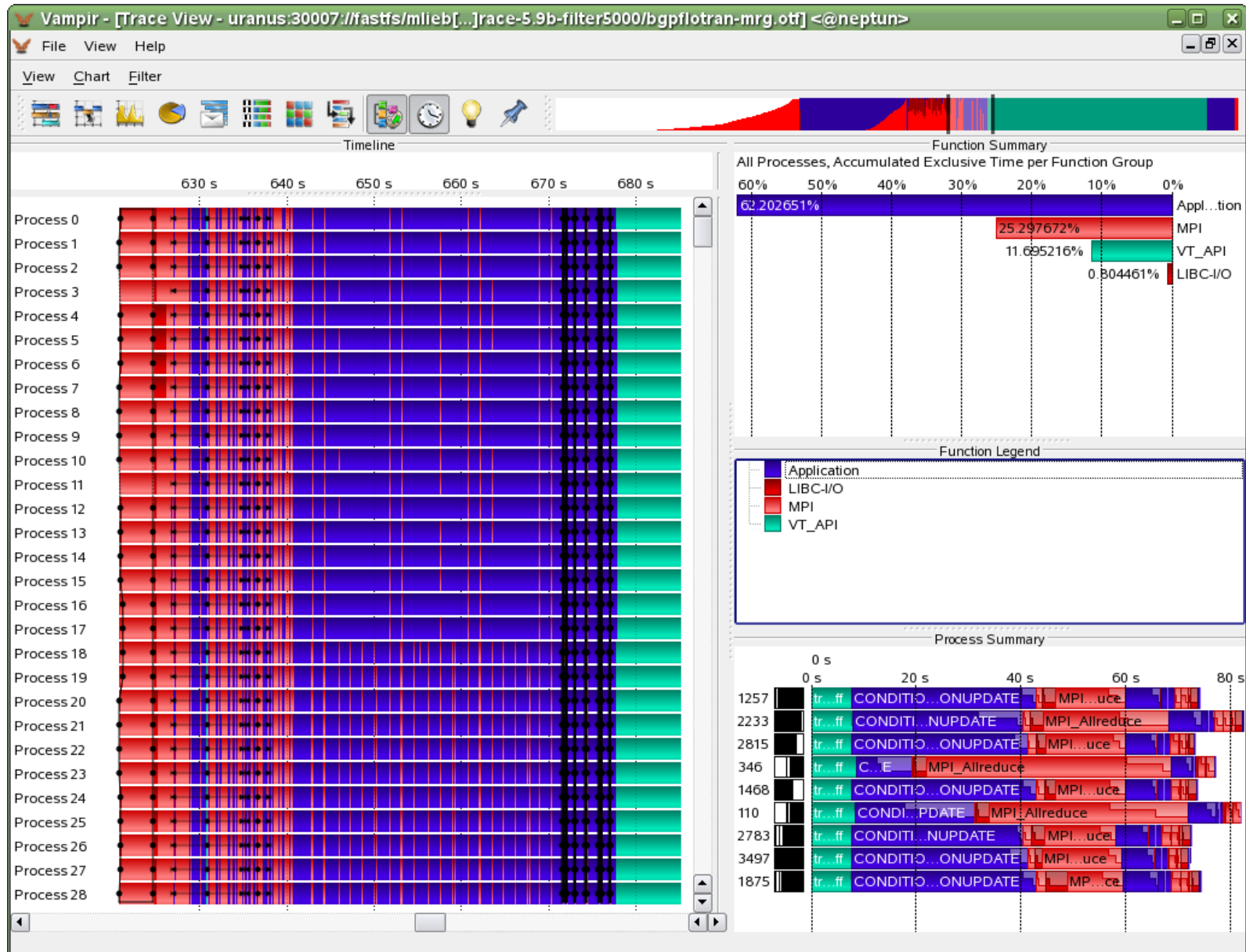
Timestep 7 zoom



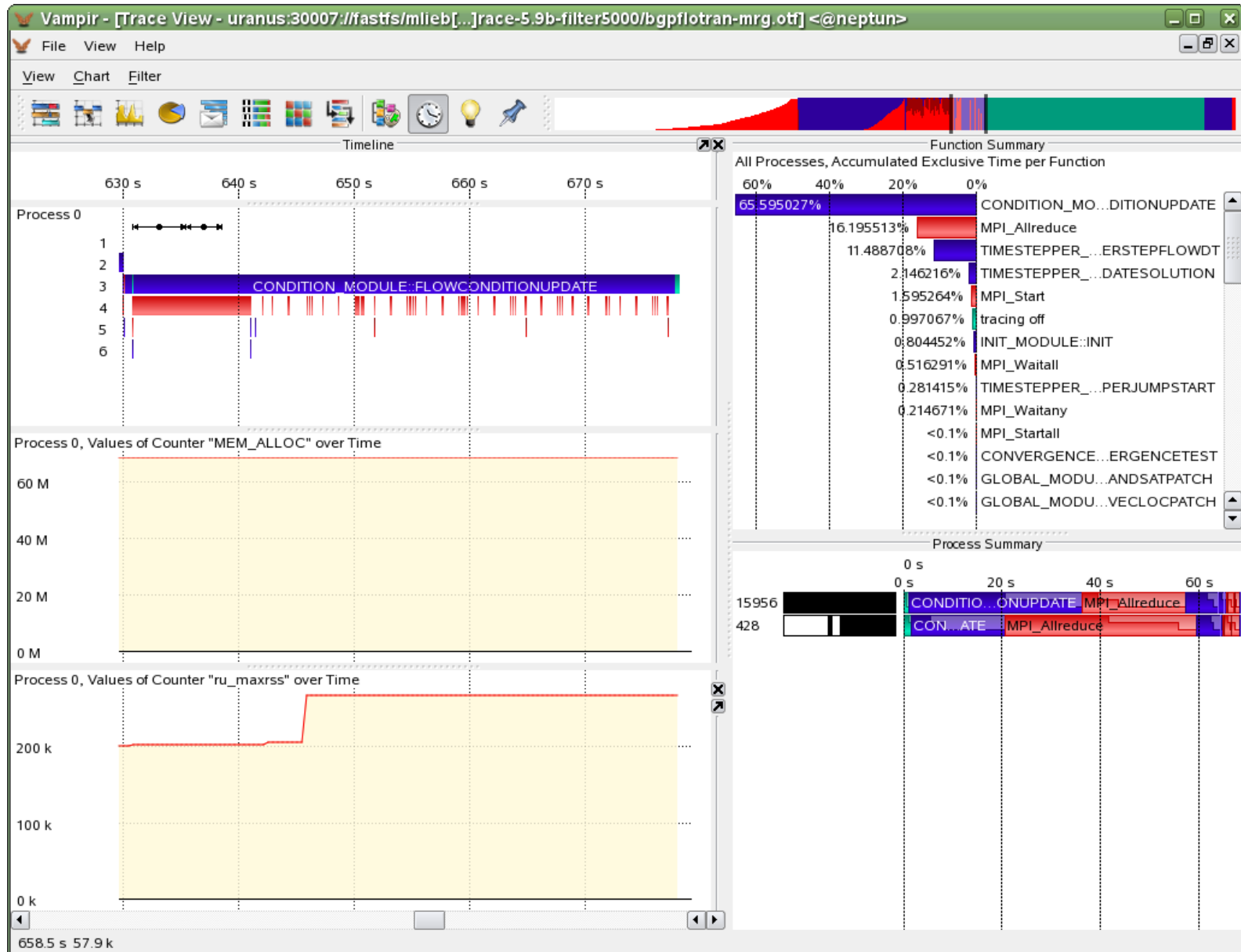
PFLOW phase zoom



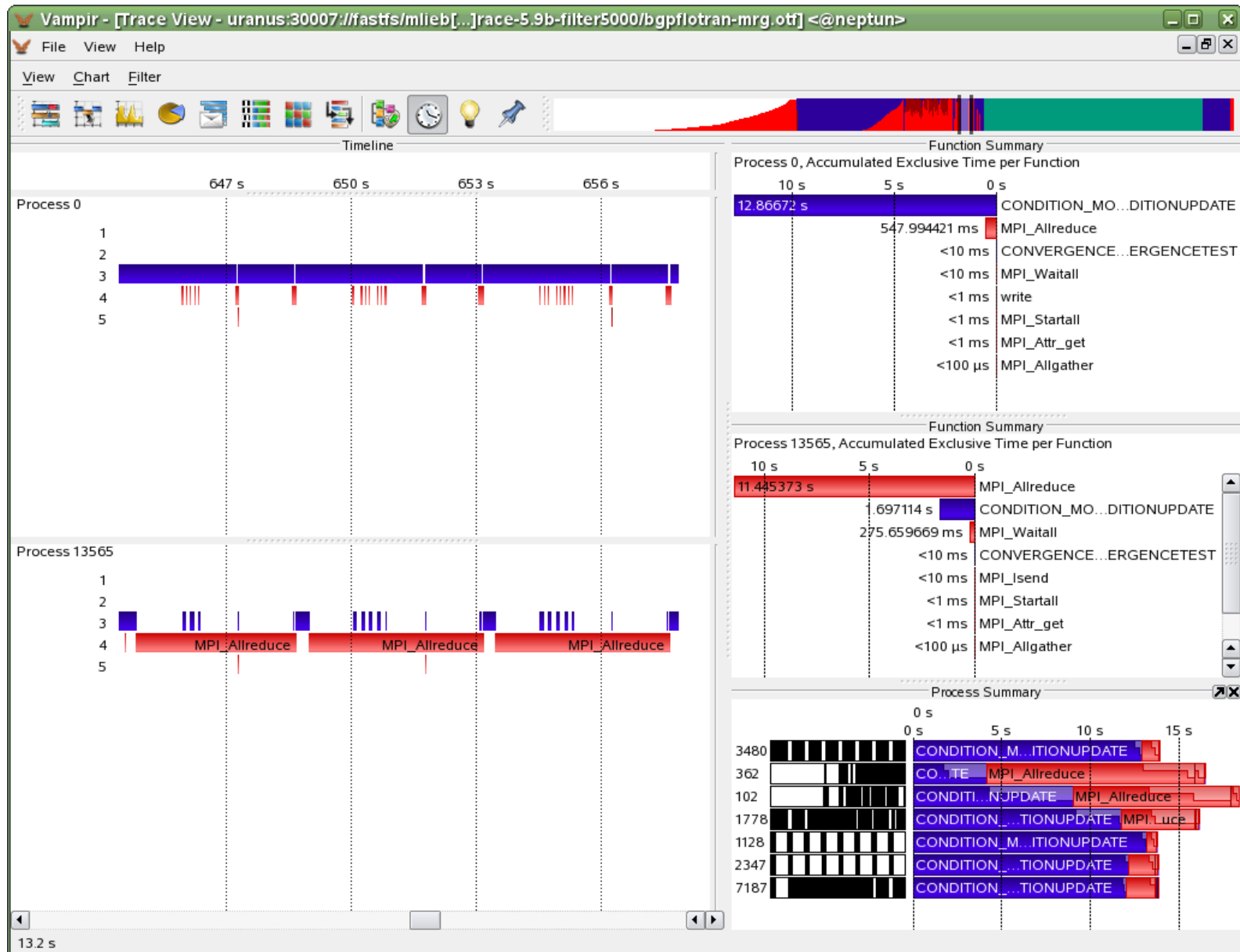
End of init phase + 1st timestep



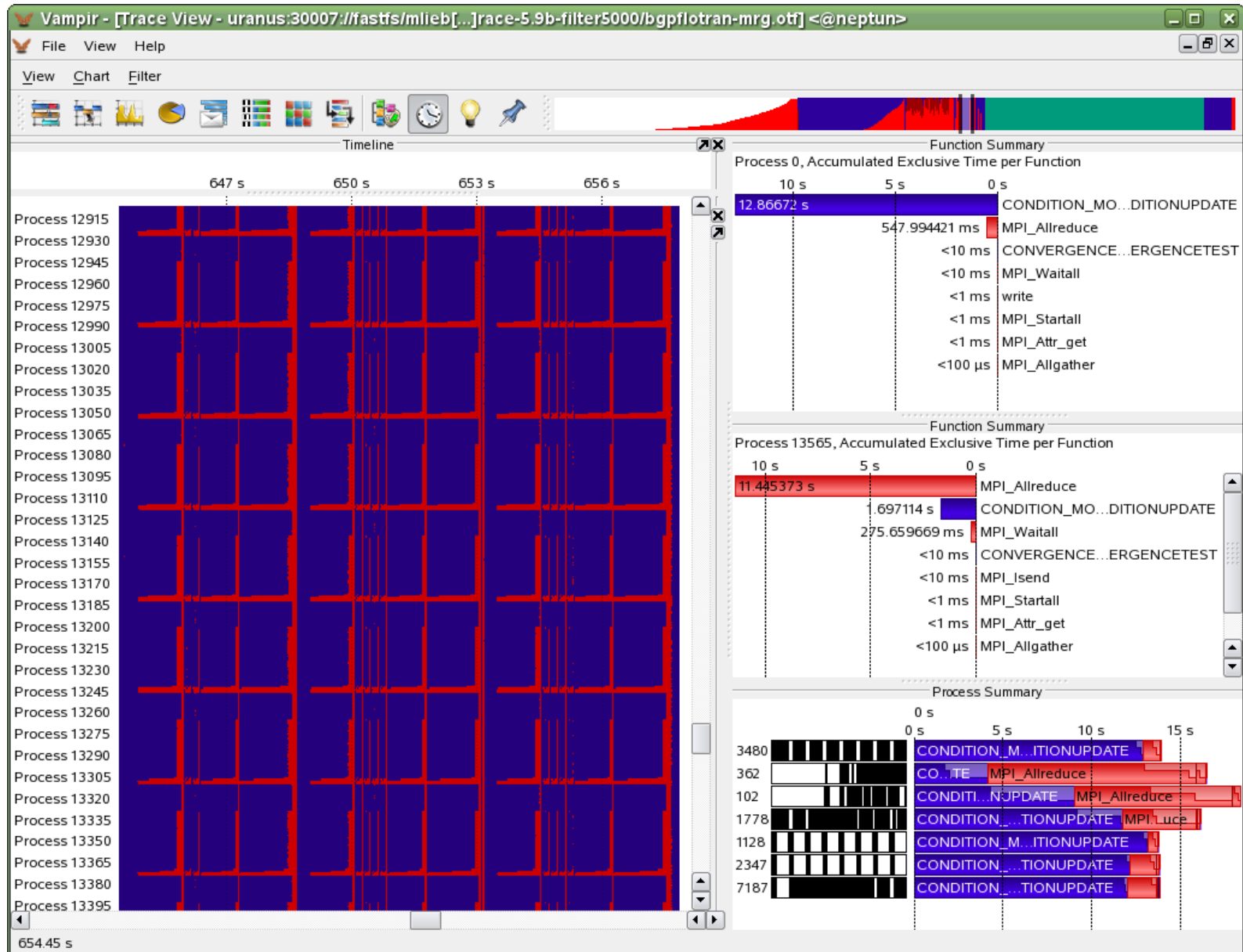
Clustering of processes



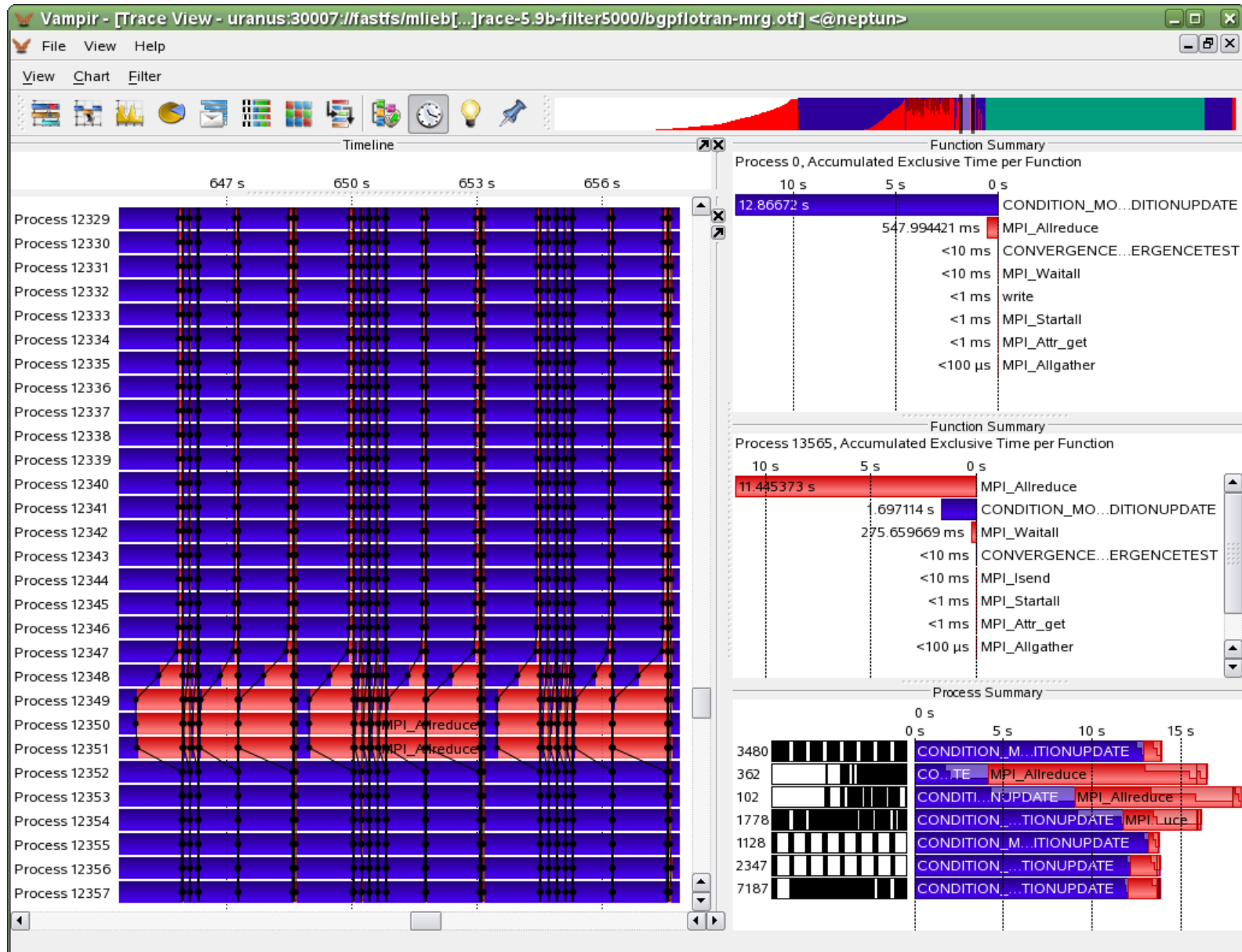
Comparing process timelines



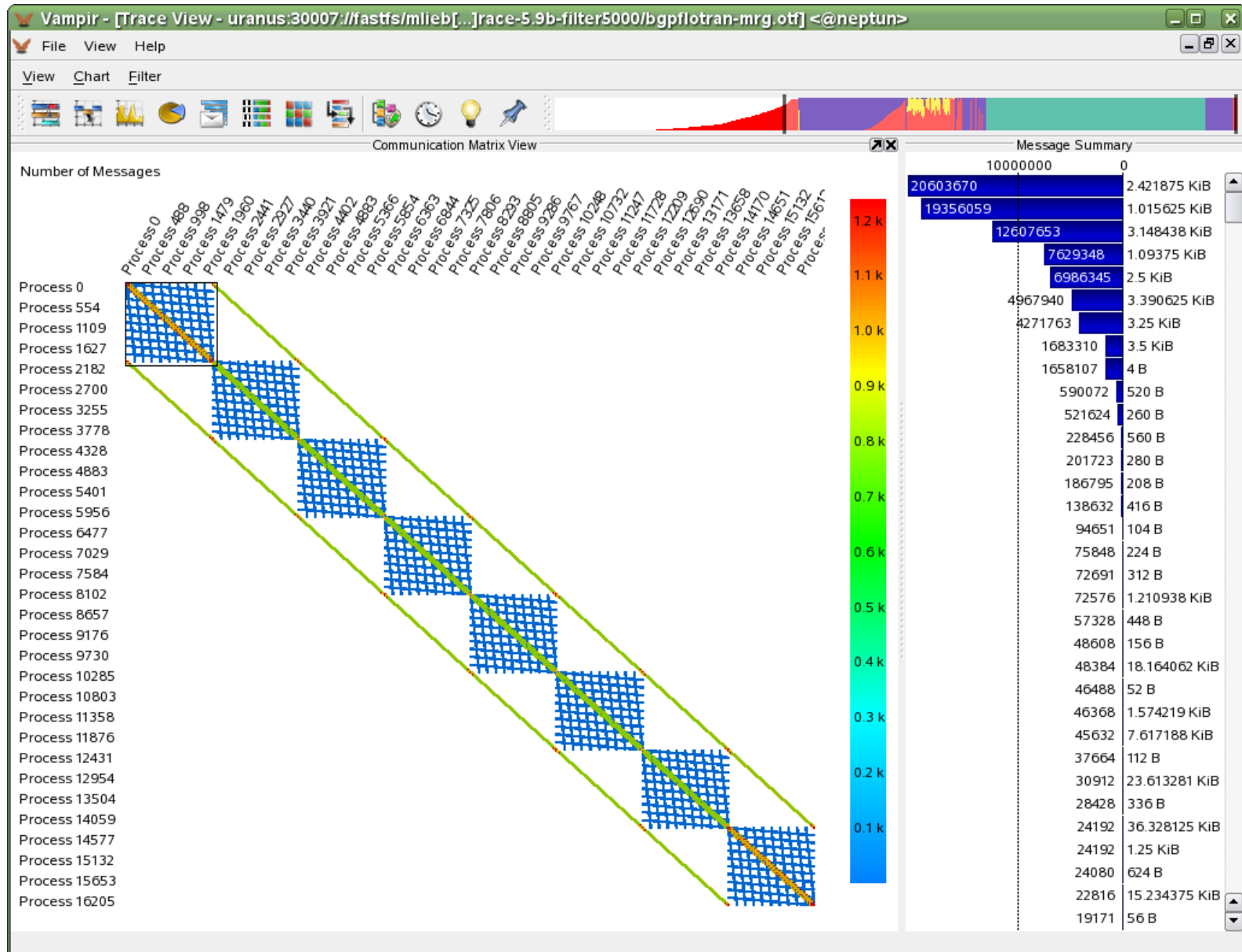
1st timestep: imbalance



1st timestep: imbalance (zoom)



Comm matrix + message hist



- Use PDT for selective instrumentation
 - Select user source routines taking more than 1% of exclusive execution time
 - 4 from PFLOTRAN, 19 from PETSc [+44 MPI]
- Collect profiles on Jaguar Cray XT5 from runs with varying numbers of MPI processes
 - Include PAPI preset hardware counters
- Use ParaProf Manager to browse experiments
 - Examine 2D & 3D graphical profile, histogram and callgraph presentations

ParaProf Manager (16380)



TAU: ParaProf Manager

File Options Help

Applications

- Standard Applications
 - Default App
 - Default Exp
 - pflotran_xt5_16380_reduced.xml
 - TIME
 - PAPI_FP_OPS
 - PAPI_TOT_INS
 - PAPI_L1_DCA
 - PAPI_L1_DCM
 - PAPI_RES_STL
 - PAPI_TOT_CYC
 - PAPI_L2_TCA
 - PAPI_L2_TCM
- Default (jdbc:derby:/home/users/wsppear/tau2/x86_64/lib/perfdmf)
- peri_s3d (jdbc:postgresql://apollo.cs.uoregon.edu:5432/peri_s3d)
- scott (jdbc:derby:/home/users/scottb/.ParaProf/perfdmf)
- peri_pflotran (jdbc:postgresql://apollo.cs.uoregon.edu:5432/peri_pflotran)

TrialField	Value
Name	pflotran_xt5_16380_reduced.xml
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	6
CPU MHz	2600.000
CPU Type	6-Core AMD Opteron(tm) Processor 23 ...
CPU Vendor	AuthenticAMD
CWD	/lustre/widow1/scratch/wsppear/pflotran...
Cache Size	512 KB
Executable	/var/spool/alps/1887061/pflotran
File Type Index	10
File Type Name	TAU Snapshot
Hostname	nid10413
Local Time	2010-05-03T04:49:09-04:00
MPI Processor Name	nid10413
Memory Size	16385788 kB
Node Name	nid10413
OS Machine	x86_64
OS Name	Linux
OS Release	2.6.16.60-0.39_1.0102.4787.2.2.41-...
OS Version	#1 SMP Thu Nov 12 17:58:04 CST 2009
Starting Timestamp	1272875793196482
TAU Architecture	craycnl
TAU Config	-pdt=/ccs/home/wsppear/bin/pdtoolkit...
TAU Makefile	/ccs/home/wsppear/bin/tau2/craycnl/lib...
TAU MetaData Merge Time	0.000305 seconds
TAU Profile Merge Time	1.208 seconds
TAU Unification Time	0.001299 seconds
TAU Version	2.18.2-cvs
TAU_CALLPATH	off
TAU_CALLPATH_DEPTH	2
TAU_COMM_MATRIX	off
TAU_COMPENSATE	off
TAU_PROFILE	on
TAU_PROFILE_FORMAT	merged

.TAU_application

MPI

PETSc

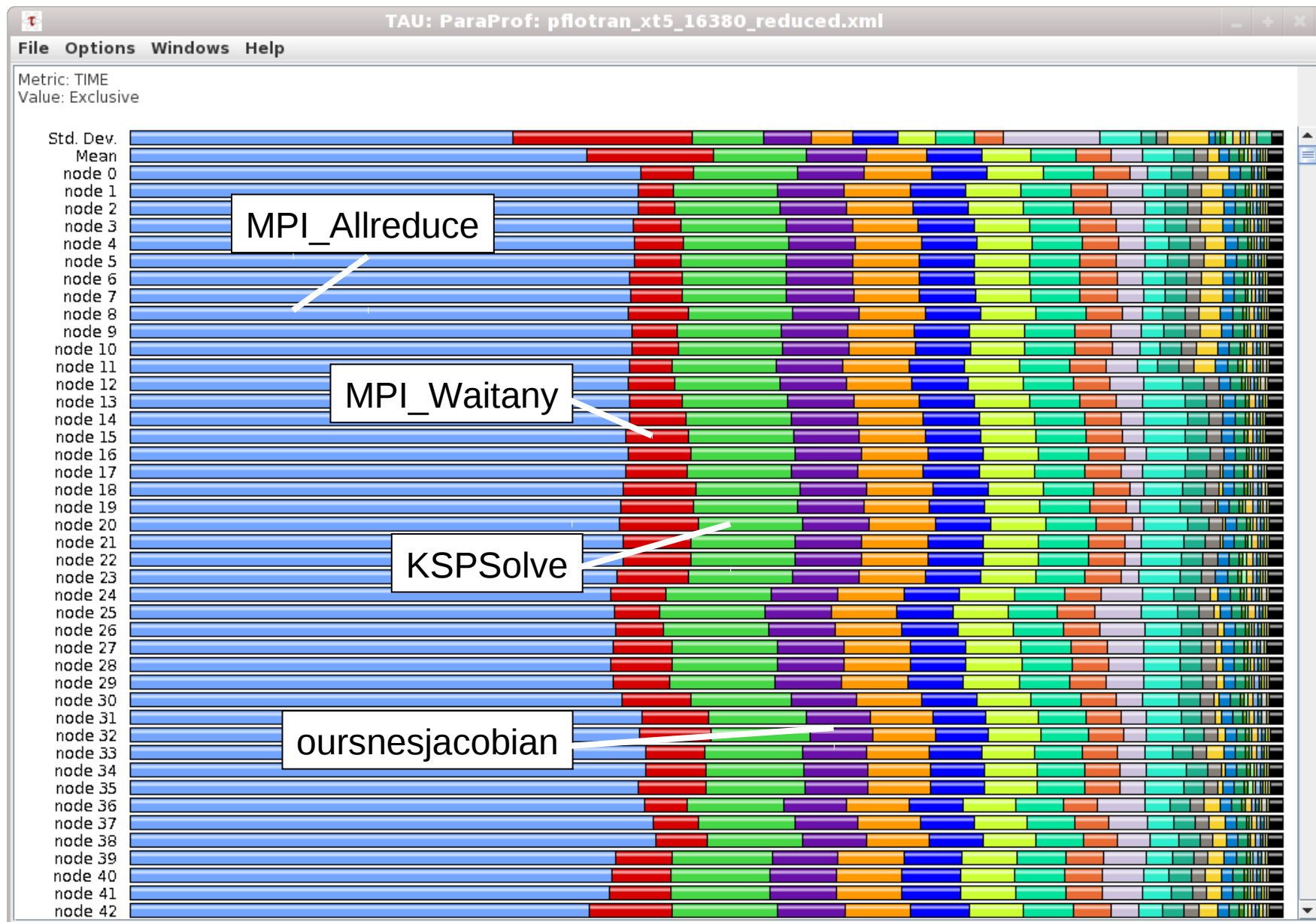
PFLOTRAN

TAU: ParaProf: Function Legend: pflotran_xt5_16380_reduced.xml/

File Filter Windows Help

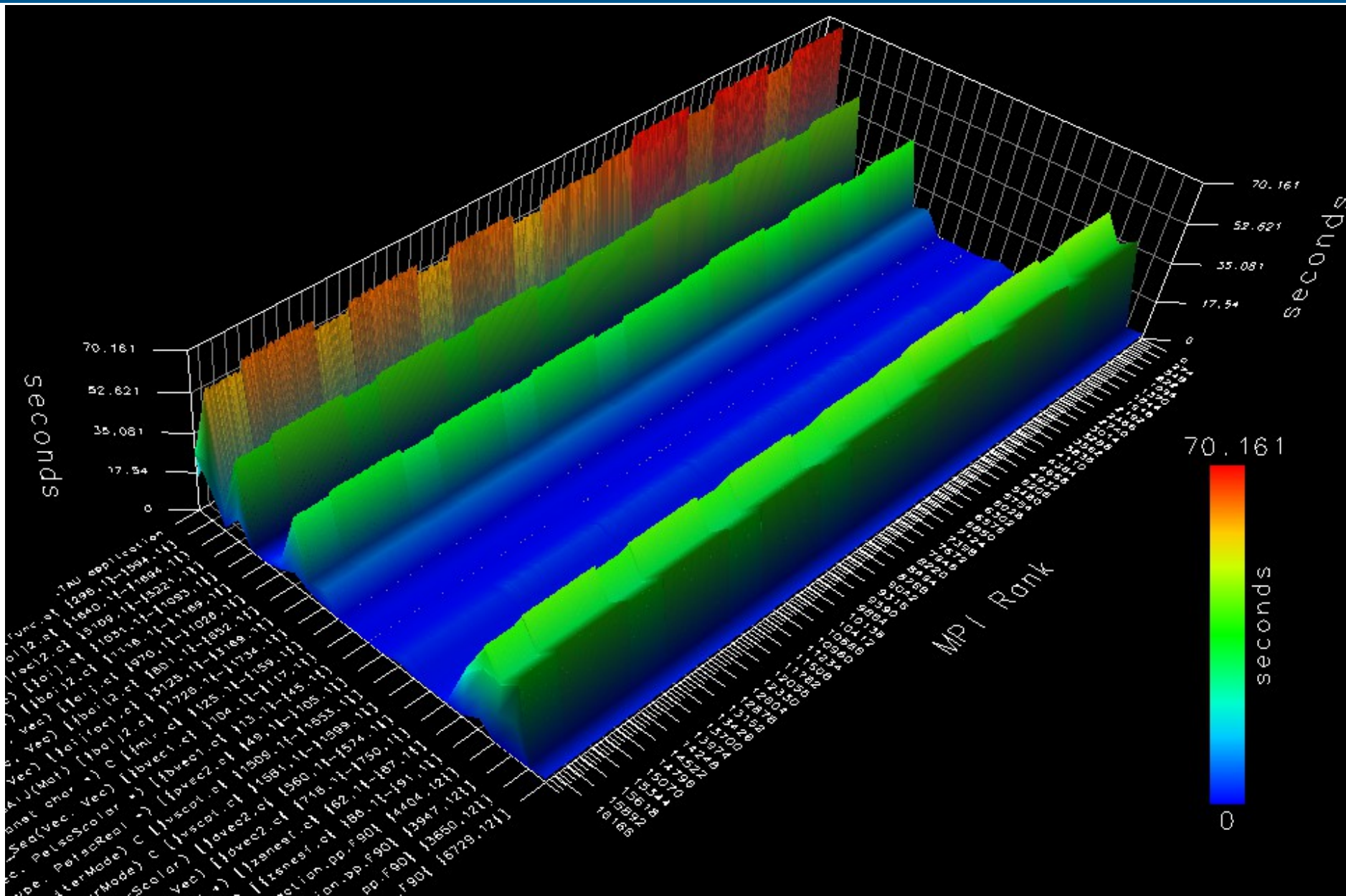
- TAU application
- MPI_Allgather()
- MPI_Allreduce()
- MPI_Attr_delete()
- MPI_Attr_get()
- MPI_Attr_put()
- MPI_Barrier()
- MPI_Bcast()
- MPI_Comm_dup()
- MPI_Comm_free()
- MPI_Comm_group()
- MPI_Comm_rank()
- MPI_Comm_size()
- MPI_Comm_split()
- MPI_Exscan()
- MPI_File_close()
- MPI_File_get_size()
- MPI_File_open()
- MPI_File_read_at()
- MPI_Finalize()
- MPI_Gather()
- MPI_Get_count()
- MPI_Get_elements()
- MPI_Init()
- MPI_Irecv()
- MPI_Isend()
- MPI_Keyval_create()
- MPI_Keyval_free()
- MPI_Op_create()
- MPI_Op_free()
- MPI_Recv()
- MPI_Recv_init()
- MPI_Request_free()
- MPI_Scan()
- MPI_Scatterv()
- MPI_Send()
- MPI_Send_init()
- MPI_Start()
- MPI_Startall()
- MPI_Type_commit()
- MPI_Type_contiguous()
- MPI_Type_free()
- MPI_Type_size()
- MPI_Waitall()
- MPI_Waitany()
- PetscErrorCode KSPSolve(KSP, Vec, Vec) C [{itfunc.c} {298.1}-{594.1}]
- PetscErrorCode MatDiagonalScale_SeqBAIJ(Mat, Vec, Vec) [{bai2.c} {1640.1}-{1694.1}]
- PetscErrorCode MatLUFactorNumeric_SeqBAIJ_N(Mat, Mat, const MatFactorInfo *) [{bai2.c} {5109.1}-{5221.1}]
- PetscErrorCode MatMultAdd_SeqBAIJ(Mat, Vec, Vec, Vec) [{aij.c} {1031.1}-{1093.1}]
- PetscErrorCode MatMultAdd_SeqBAIJ_N(Mat, Vec, Vec, Vec) [{bai2.c} {1116.1}-{1169.1}]
- PetscErrorCode MatMult_SeqBAIJ(Mat, Vec, Vec) [{aij.c} {970.1}-{1026.1}]
- PetscErrorCode MatMult_SeqBAIJ_N(Mat, Vec, Vec) [{bai2.c} {601.1}-{652.1}]
- PetscErrorCode MatSolve_SeqBAIJ_NaturalOrdering(Mat, Vec, Vec) [{aijfact.c} {3125.1}-{3169.1}]
- PetscErrorCode MatZeroEntries_SeqBAIJ(Mat) [{bai2.c} {1726.1}-{1734.1}]
- PetscErrorCode PetscMallocValidat(int, const char *, const char *, const char *) C [{mtr.c} {125.1}-{159.1}]
- PetscErrorCode VecCopy_Seq(Vec, Vec) [{bvec1.c} {104.1}-{117.1}]
- PetscErrorCode VecDot_Seq(Vec, Vec, PetscScalar *) [{bvec1.c} {13.1}-{45.1}]
- PetscErrorCode VecNorm_MPI(Vec, NormType, PetscReal *) [{pvec2.c} {49.1}-{105.1}]
- PetscErrorCode VecScatterBegin(VecScatter, Vec, Vec, InsertMode, ScatterMode) C [{vscat.c} {1509.1}-{1553.1}]
- PetscErrorCode VecScatterEnd(VecScatter, Vec, Vec, InsertMode, ScatterMode) C [{vscat.c} {1581.1}-{1599.1}]
- PetscErrorCode VecSet_Seq(Vec, PetscScalar) [{dvec2.c} {560.1}-{574.1}]
- PetscErrorCode VecWAXPY_Seq(Vec, PetscScalar, Vec, Vec) [{dvec2.c} {718.1}-{750.1}]
- PetscErrorCode oursnesfunction(SNES, Vec, Vec, void *) [{zsnesf.c} {62.1}-{67.1}]
- PetscErrorCode oursnesjacobian(SNES, Vec, Mat *, Mat *, MatStructure *, void *) [{zsnesf.c} {86.1}-{91.1}]
- REACTION_MODULE::RMULTIRATESORPTION [{reaction.pp.F90} {4404.12}]
- REACTION_MODULE::RTOTAL [{reaction.pp.F90} {3947.12}]
- TIMESTEPPEP_MODULE::STEPPERSTEPSTRANSPORTOT [{timestepper.pp.F90} {3650.12}]
- TIMESTEPPEP_MODULE::STEPPERUPDATETRANSPORTSOLUTION [{timestepper.pp.F90} {6729.12}]

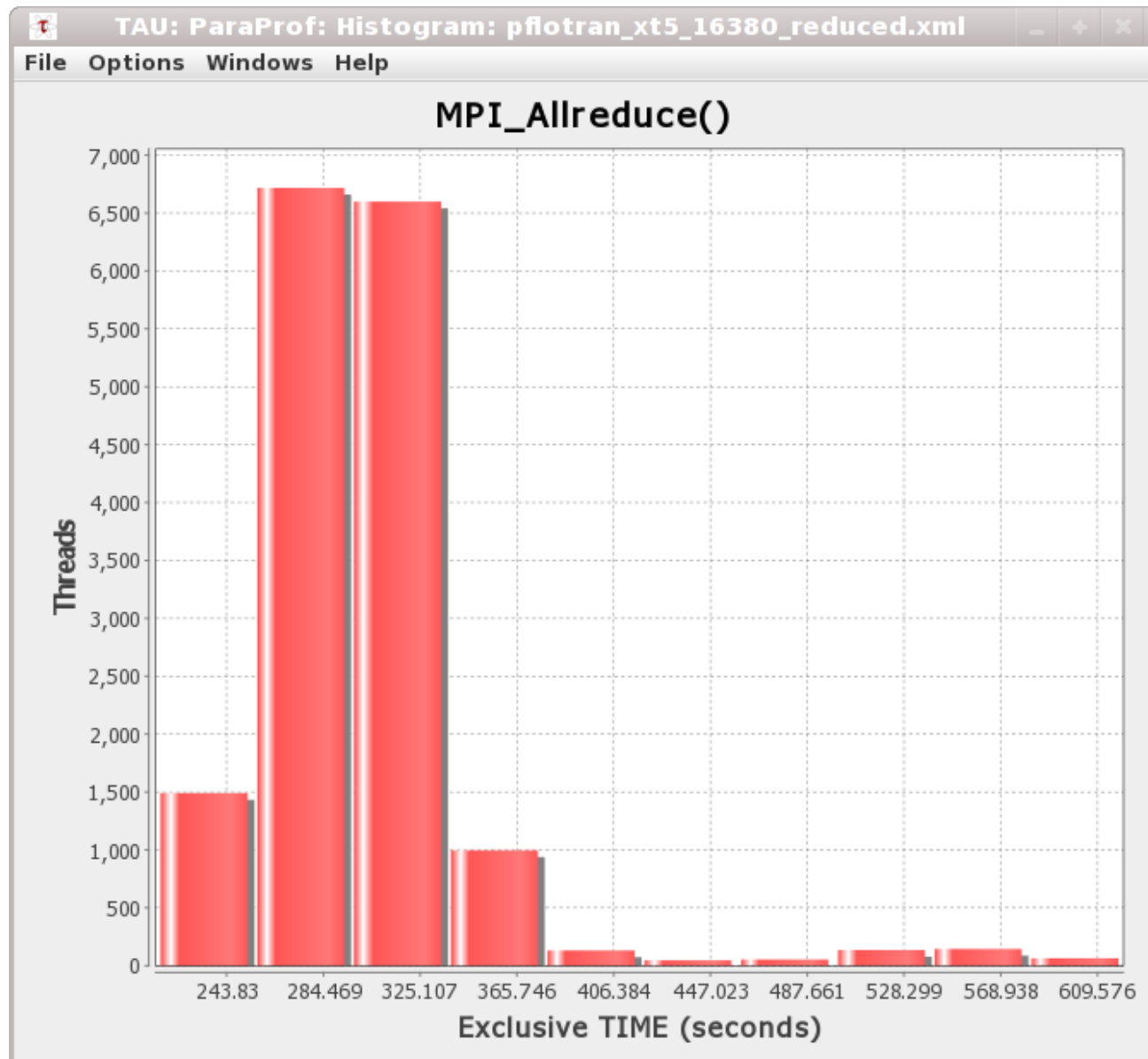
ParaProf full profile (16380)

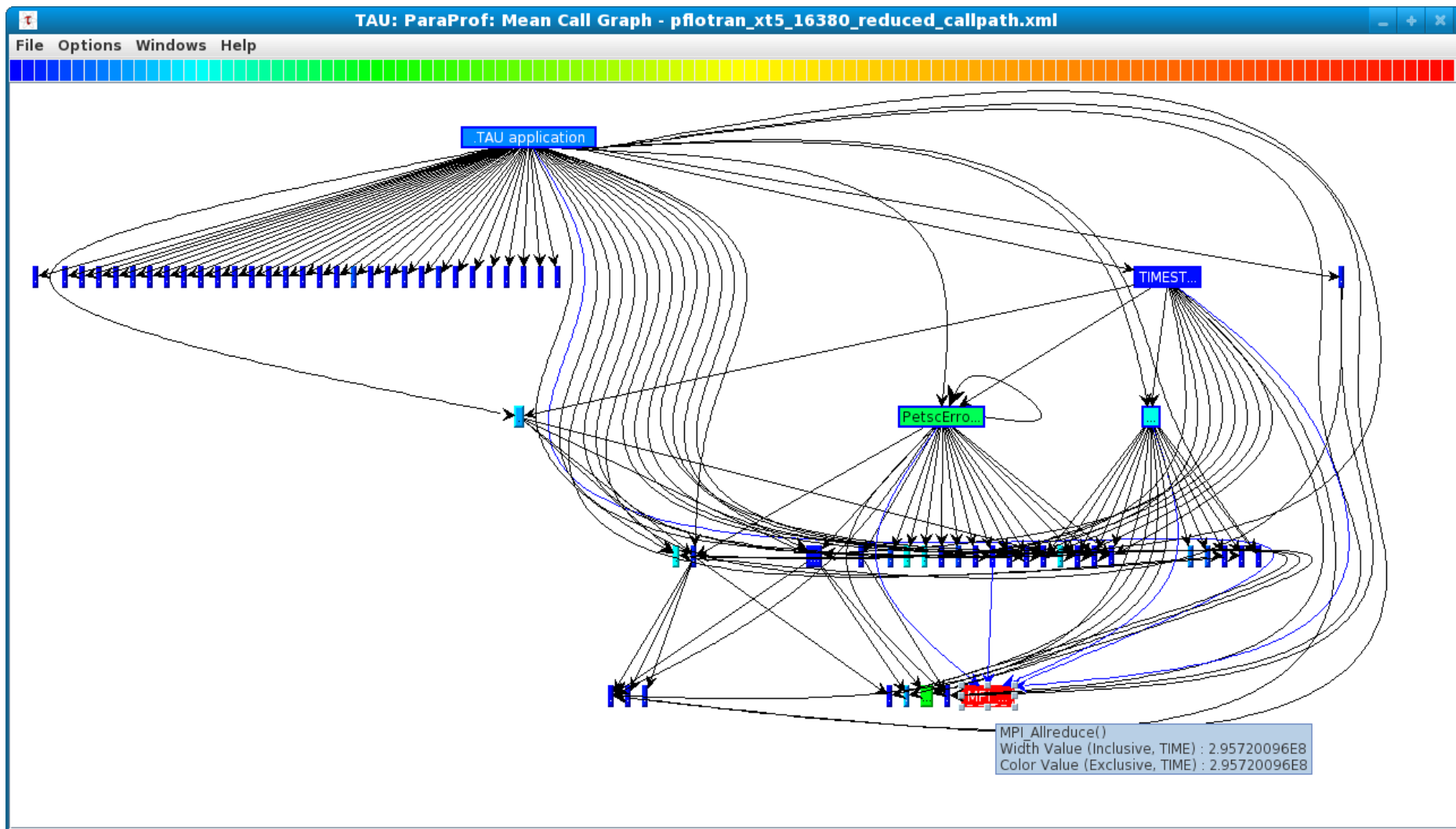




VI-HPS



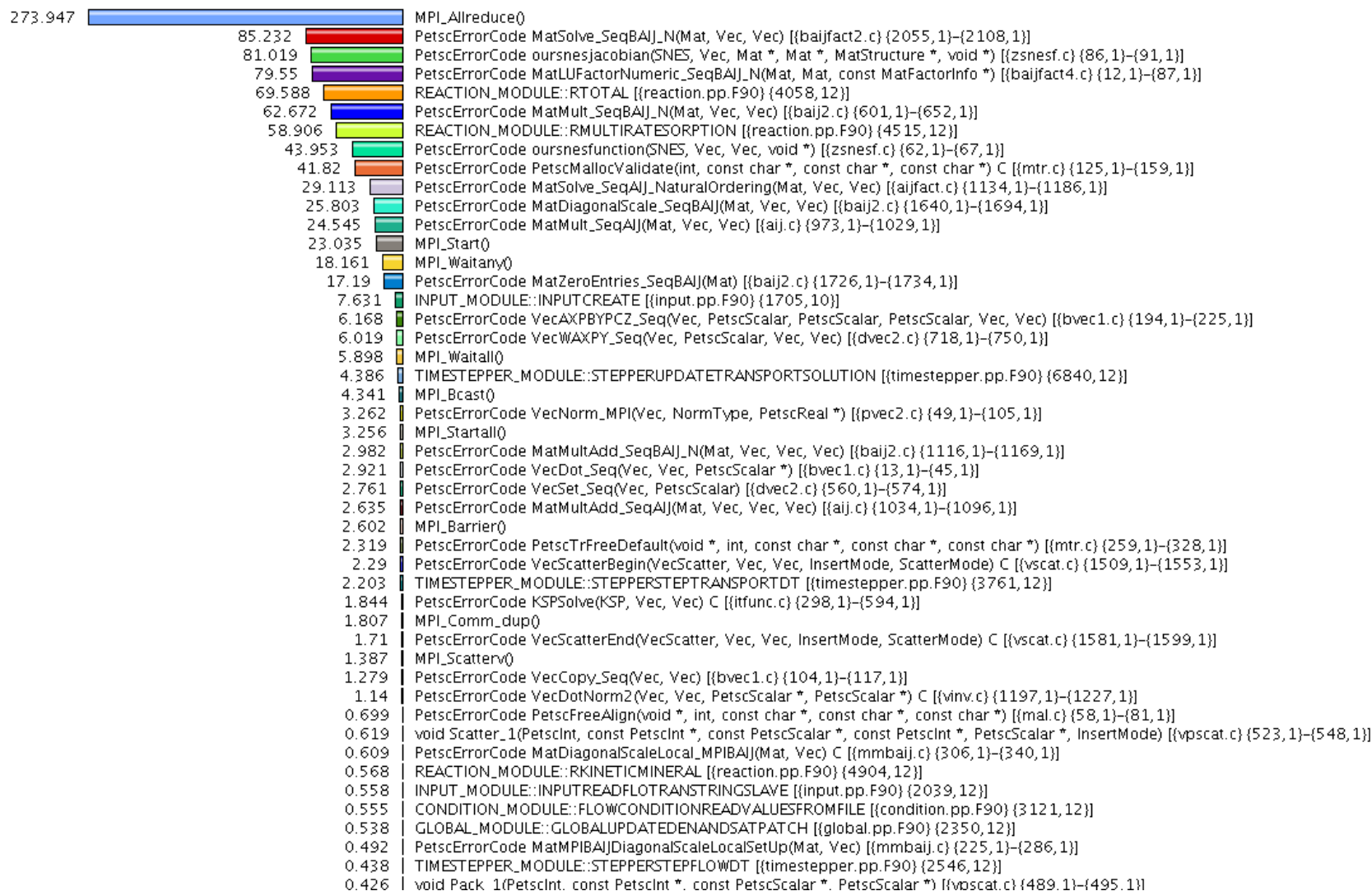


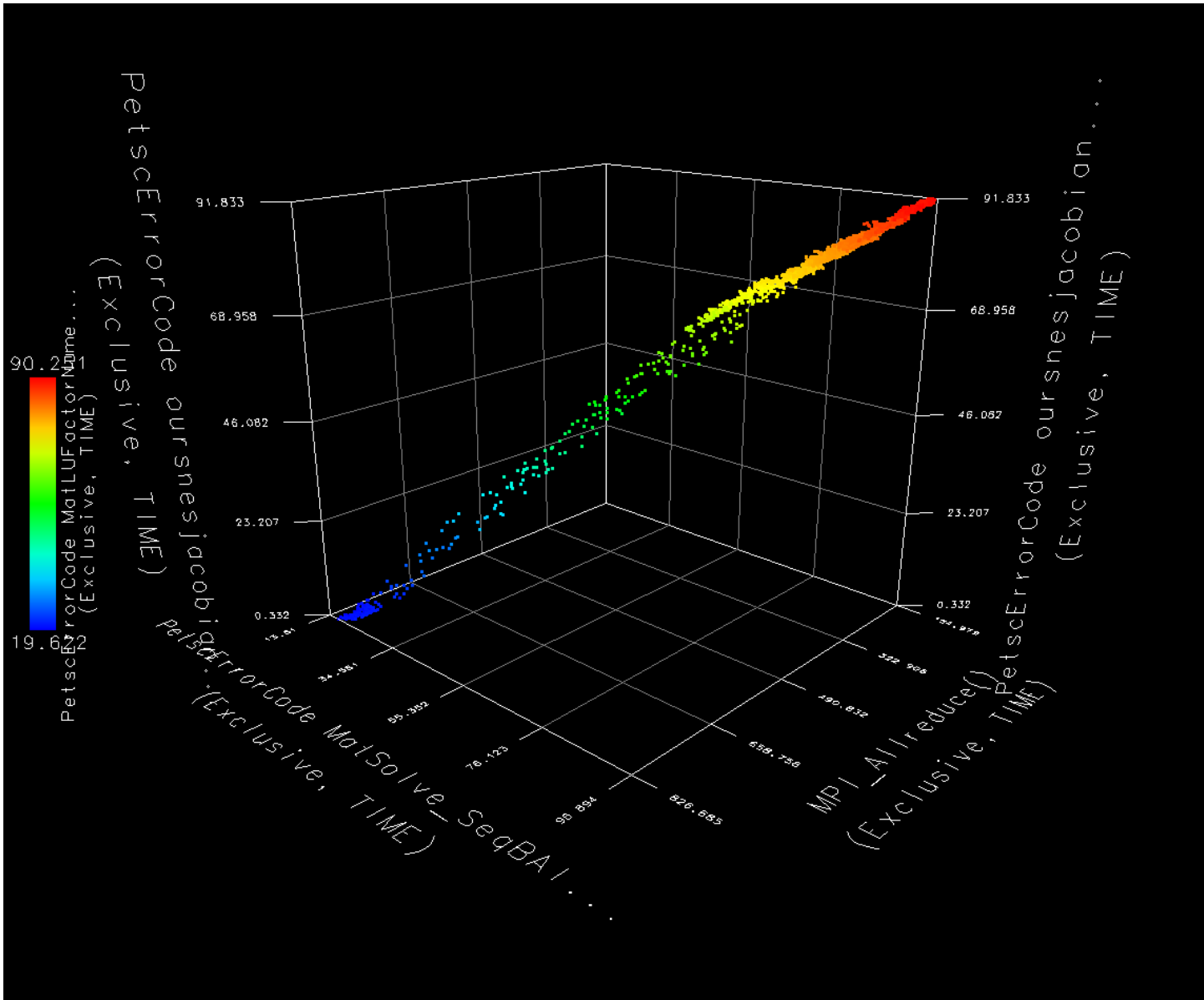


ParaProf mean profile (8184)

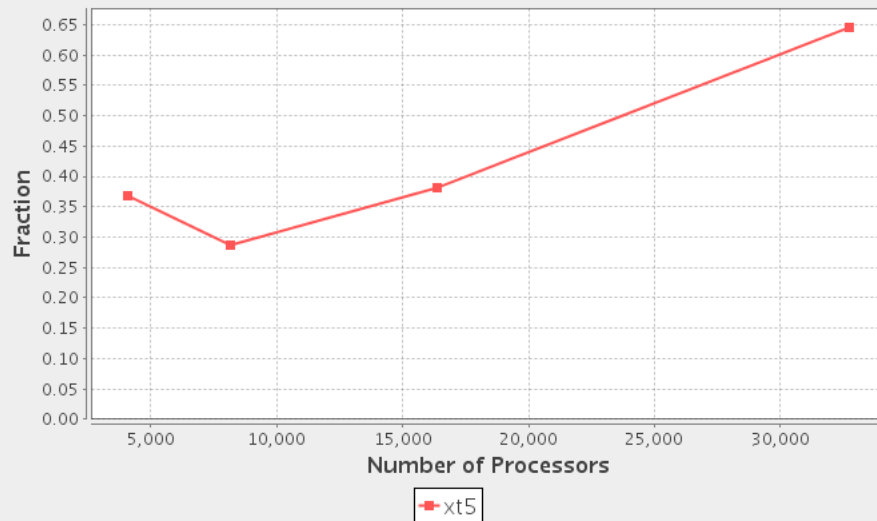


Metric: TIME
Value: Exclusive
Units: seconds

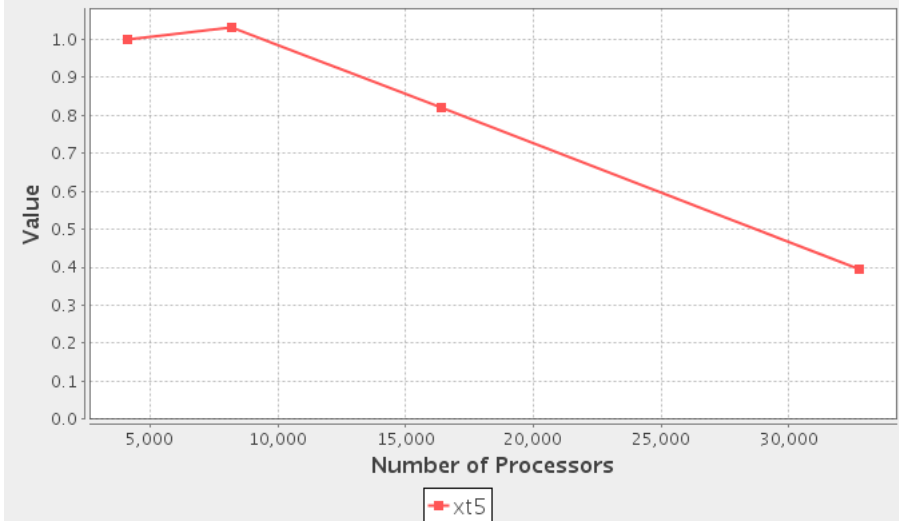




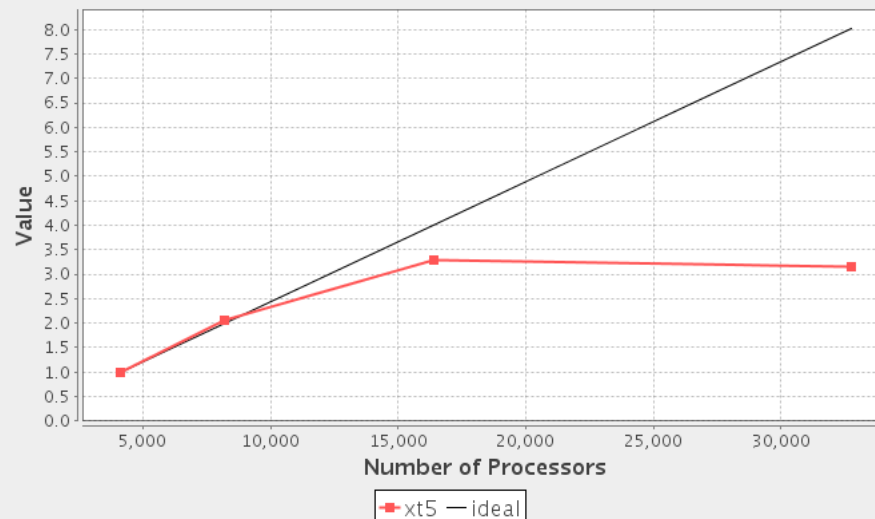
MPI TIME / Total TIME – pflotran_2b:xt5



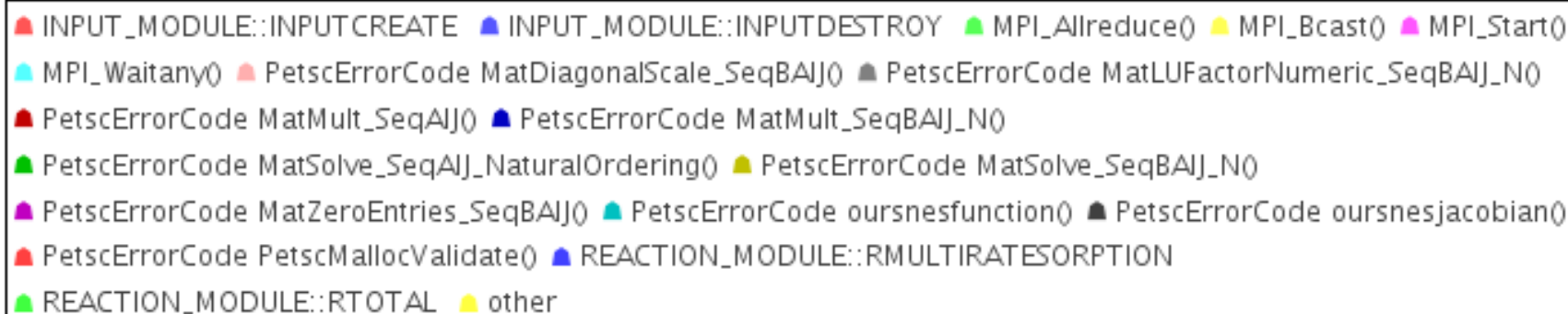
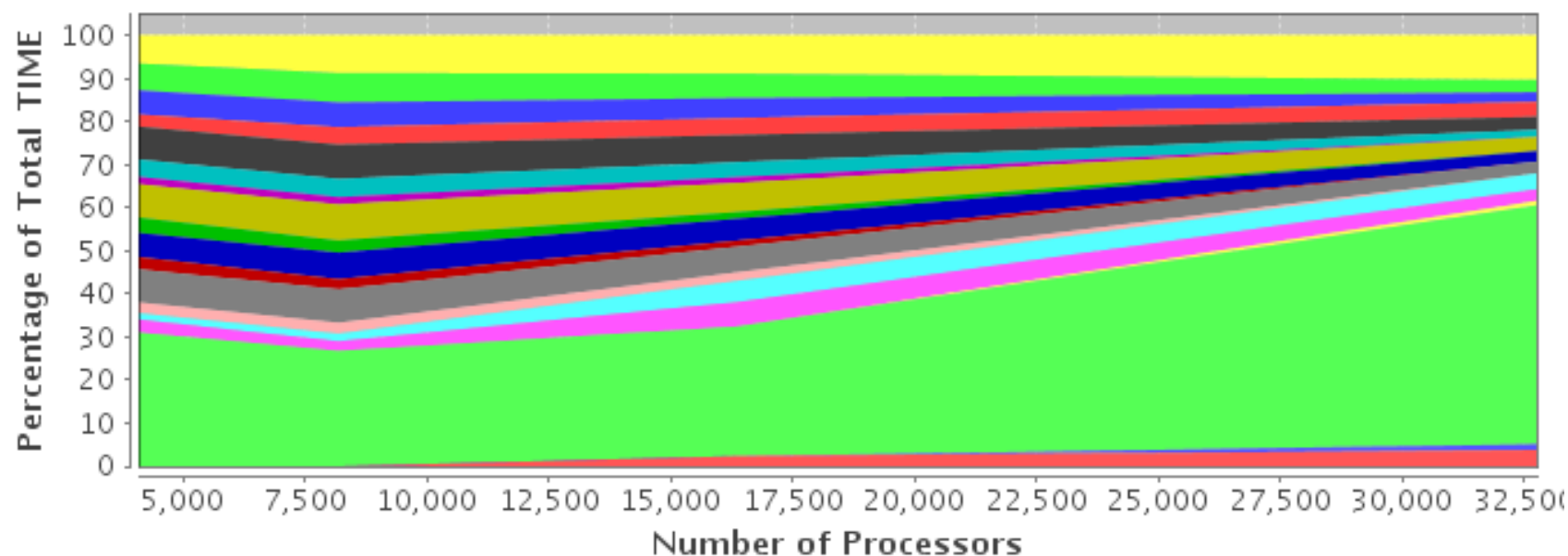
Relative Efficiency – pflotran_2b:xt5:TIME

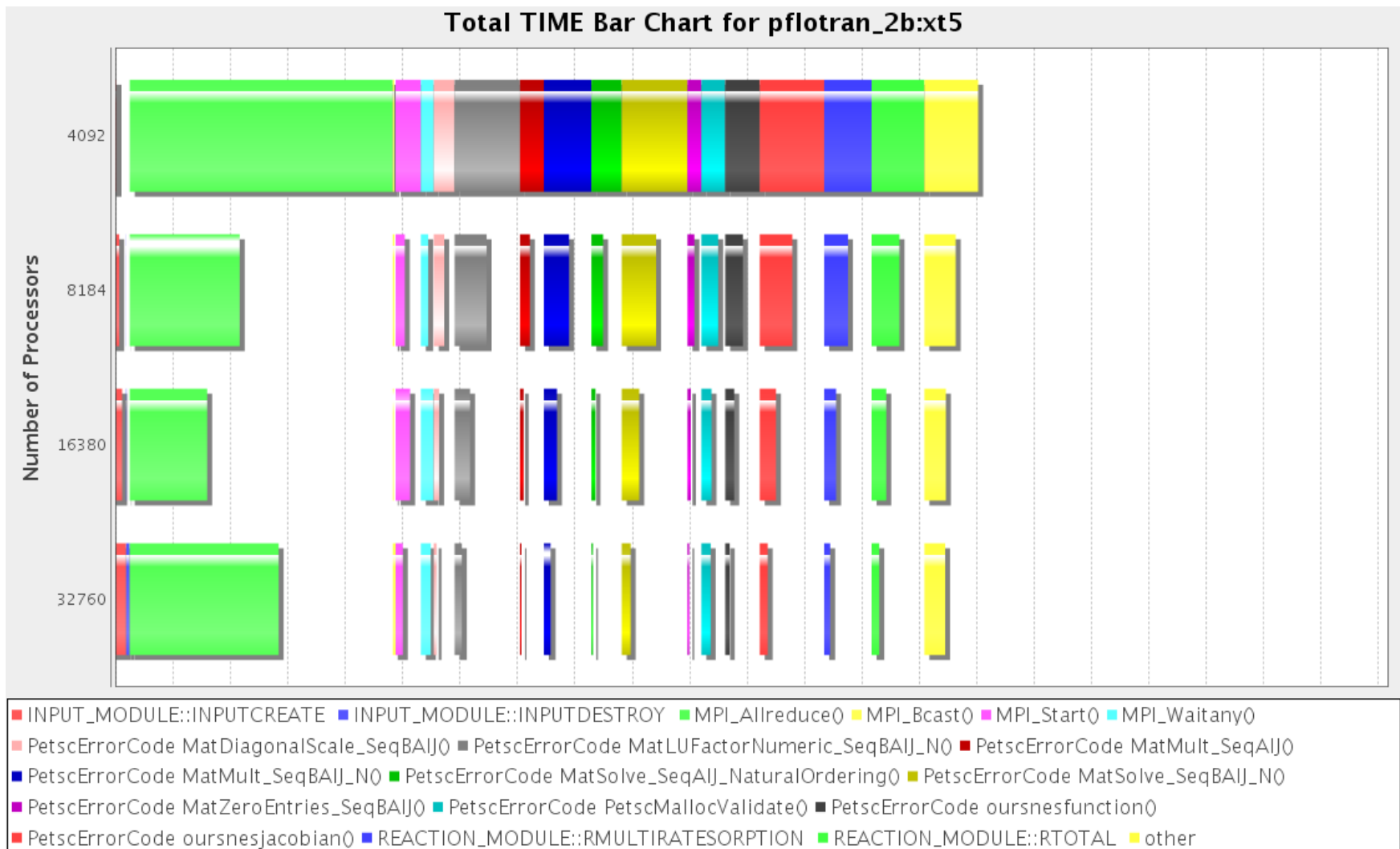


Relative Speedup – pflotran_2b:xt5:TIME



Total TIME Breakdown for pflotran_2b:xt5





TAU: ParaProf Manager

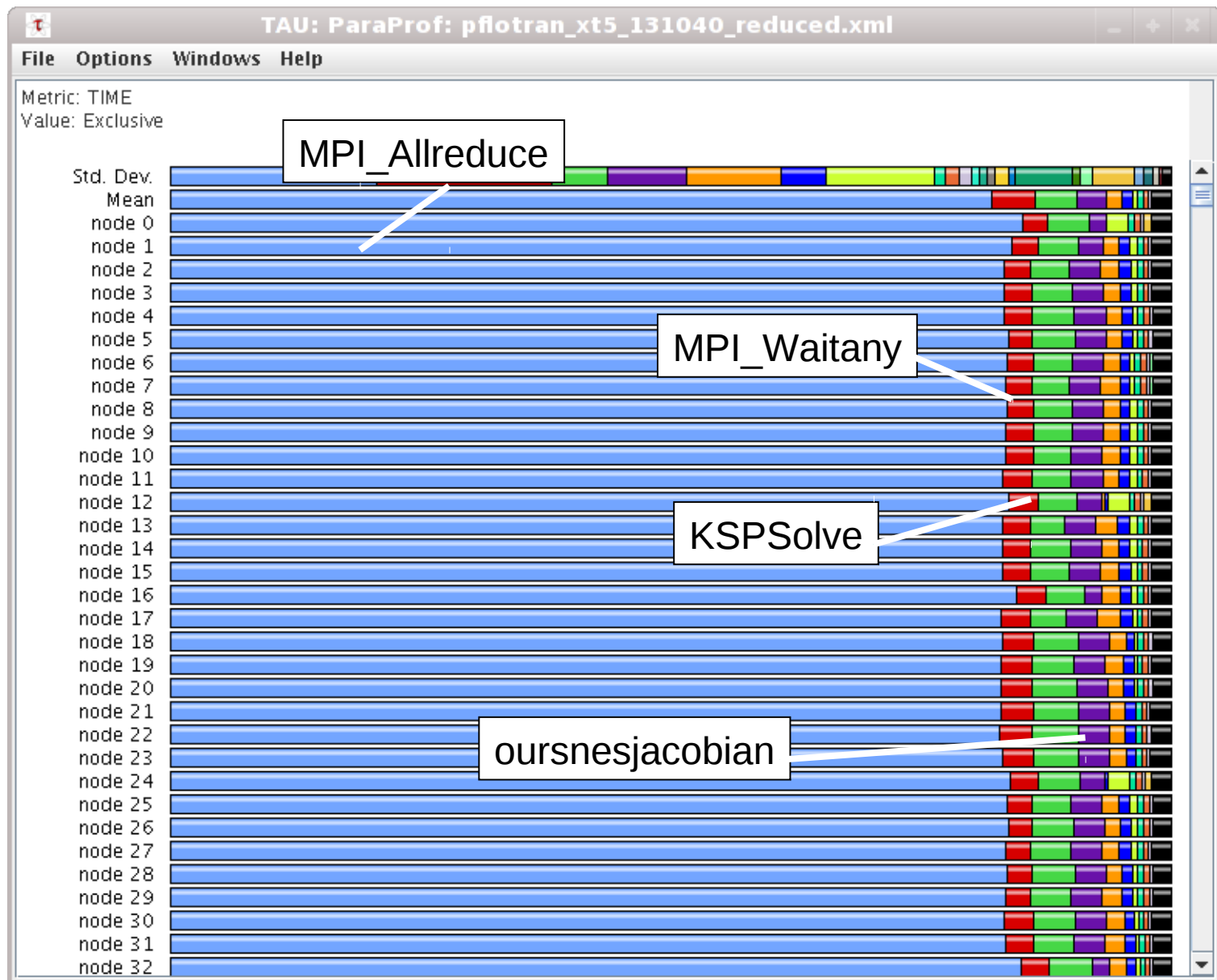
File Options Help

Applications

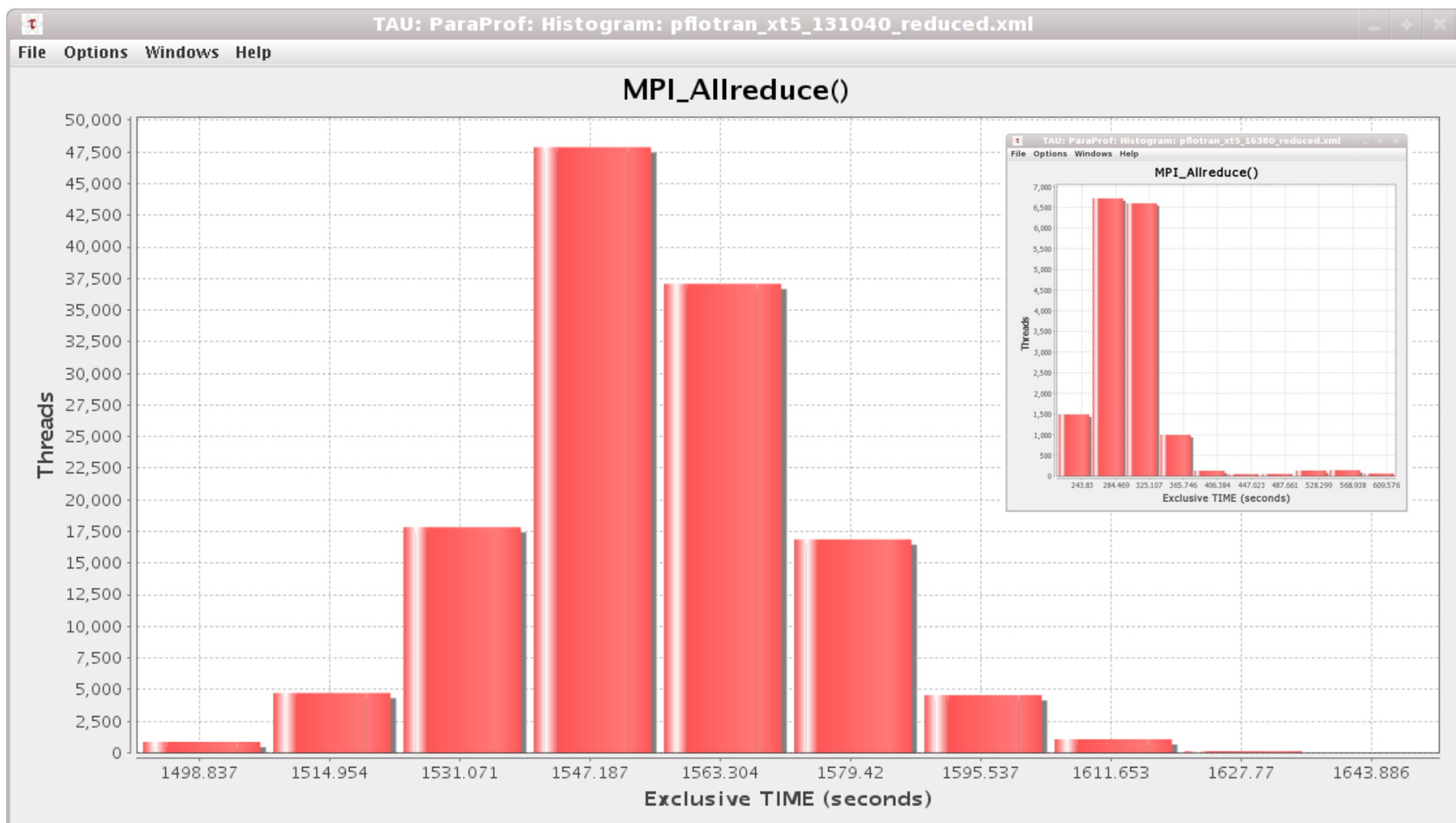
- Standard Applications
 - Default App
 - Default Exp
 - pflotran_xt5_131040_reduced.xml
 - TIME
 - PAPI_FP_OPS
 - PAPI_TOT_INS
 - PAPI_L1_DCA
 - PAPI_L1_DCM
 - PAPI_RES_STL
 - PAPI_TOT_CYC
 - PAPI_L2_TCA
 - PAPI_L2_TCM
- Default (jdbc:derby:/home/users/wsppear/tau2)
- peri_s3d (jdbc:postgresql://apollo.cs.uoregon)
- scott (jdbc:derby:/home/users/scottb/.ParaPr
- peri_pflotran (jdbc:postgresql://apollo.cs.uore

TrialField	Value
Name	pflotran_xt5_131040_reduced.xml
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	6
CPU MHz	2600.000
CPU Type	6-Core AMD Opteron(tm) Processor 23 (D0)
CPU Vendor	AuthenticAMD
CWD	/lustre/widow1/scratch/wsppear/pflotran_2b_hug...
Cache Size	512 KB
Executable	/var/spool/alps/1891721/pflotran
File Type Index	10
File Type Name	TAU Snapshot
Hostname	nid11823
Local Time	2010-05-03T20:45:44-04:00
MPI Processor Name	nid11823
Memory Size	16385788 kB
Node Name	nid11823
OS Machine	x86_64
OS Name	Linux
OS Release	2.6.16.60-0.39_1.0102.4787.2.2.41-cn1
OS Version	#1 SMP Thu Nov 12 17:58:04 CST 2009
Starting Timestamp	1272932052236170
TAU Architecture	craycn1
TAU Config	-pdt=/ccs/home/wsppear/bin/pdtoolkit/ -pdt_c...
TAU Makefile	/ccs/home/wsppear/bin/tau2/craycn1/lib/Makefil...
TAU MetaData Merge Time	0.00041 seconds
TAU Profile Merge Time	12.96 seconds
TAU Unification Time	0.02815 seconds
TAU Version	2.18.2-cvs
TAU_CALLPATH	off
TAU_CALLPATH_DEPTH	2
TAU_COMM_MATRIX	off
TAU_COMPENSATE	off
TAU_PROFILE	on
TAU_PROFILE_FORMAT	merged
TAU_SAMPLING	off
TAU_THROTTLE	on
TAU_THROTTLE_NUMCALLS	100000
TAU_THROTTLE_PERCALL	10
TAU_TRACE	off
TAU_TRACK_HEADROOM	off
TAU_TRACK_HEAP	off
TAU_TRACK_MESSAGE	off
Timestamp	1272933945055505
UTC Time	2010-05-04T00:45:44Z
pid	32472
username	wsppear

ParaProf full profile (131040)



ParaProf histogram (131040)



- Duplication of MPI Communicators by HDF5 dominates initialization at scale (particularly on XT5)
- MPI_Allreduce collective communication remains a severe bottleneck in the timestep loop
 - for both FLOW & TRAN phases
 - particularly MatZeroRowsLocal/PetscMaxSum
- Computational imbalance from inactive grid cells evident for processes at one end of 3D process grid
 - widespread in the computational routines
 - since only a minority of processes, little to gain

- Performance analysis of complex applications at large-scale requires care
 - full automatic instrumentation is convenient, but may produce more detail than desirable
 - selective measurement and/or instrumentation may be used to reduce overhead and size of event traces
 - even basic reduced execution profiles for many thousands of processes rapidly become awkwardly large
 - powerful analyses and interactive customizable visualizations required for an effective initial overview leading to in-depth refinement of performance issues
- VI-HPS experts are available to assist

- Applications can be prepared for measurement using manual, selective and automatic instrumentation and PAPI counters
- Scalasca runtime summary & automatic event trace analyses quantify and isolate locations of MPI communication & synchronization overheads
- Vampir visual trace analysis supports interactive exploration of detailed process interaction and clustering of similar traces
- TAU experiment management, graphical profile displays and extensive instrumentation options facilitate customization of measurements and analysis presentations
- VI-HPS tools can be used independently, however, maximum benefit offered by exploiting their complementary integrated capabilities

- PFLOTRAN developers for providing an interesting large-scale test case and assistance building/running their code
- Dagstuhl seminar organizers for setting the tools challenge and hosting the associated discussions
- NIC/JSC & NCCS/ORNL for use of compute-time allocations on their large-scale facilities
- Support and development teams for the various tools for their prompt support