Performance Analysis with Vampir

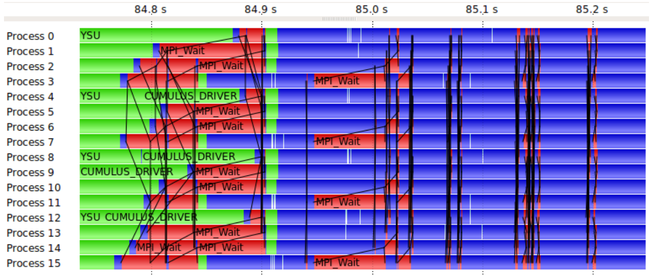
Bert Wesarg
Technische Universität Dresden

Event Trace Visualization with Vampir

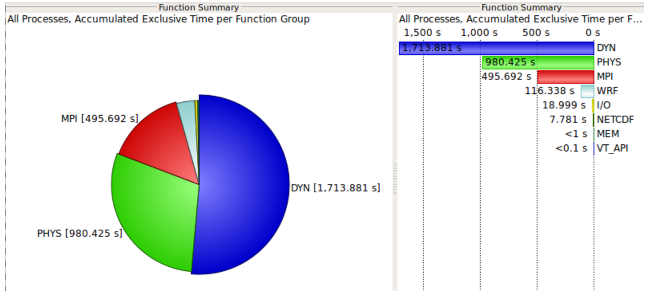
- Visualization of dynamic runtime behaviour at any level of detail along with statistics and performance metrics
- Alternative and supplement to automatic analysis

- **Typical questions that Vampir helps to answer**
  - What happens in my application execution during a given time in a given process or thread?
  - How do the communication patterns of my application execute on a real system?
  - Are there any imbalances in computation, I/O or memory usage and how do they affect the parallel execution of my application?

- **Timeline charts**
  - Application activities and communication along a time axis

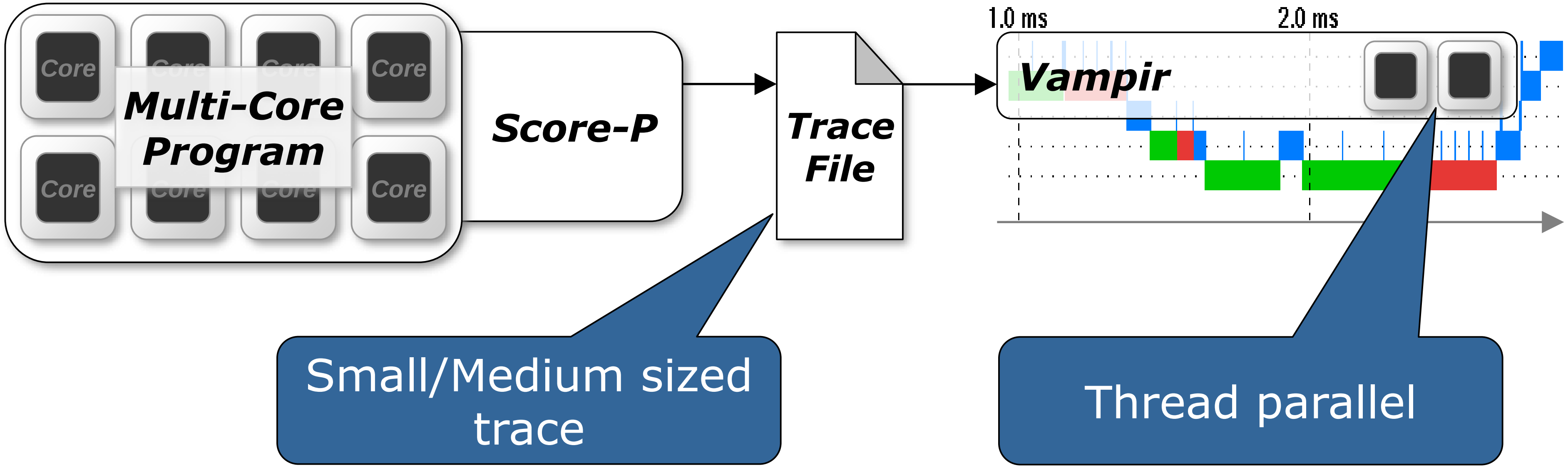


- **Summary charts**
  - Quantitative results for the currently selected time interval



Visualization Modes (1)
Directly on front end or local machine

```
% vampir
```



Small/Medium sized trace

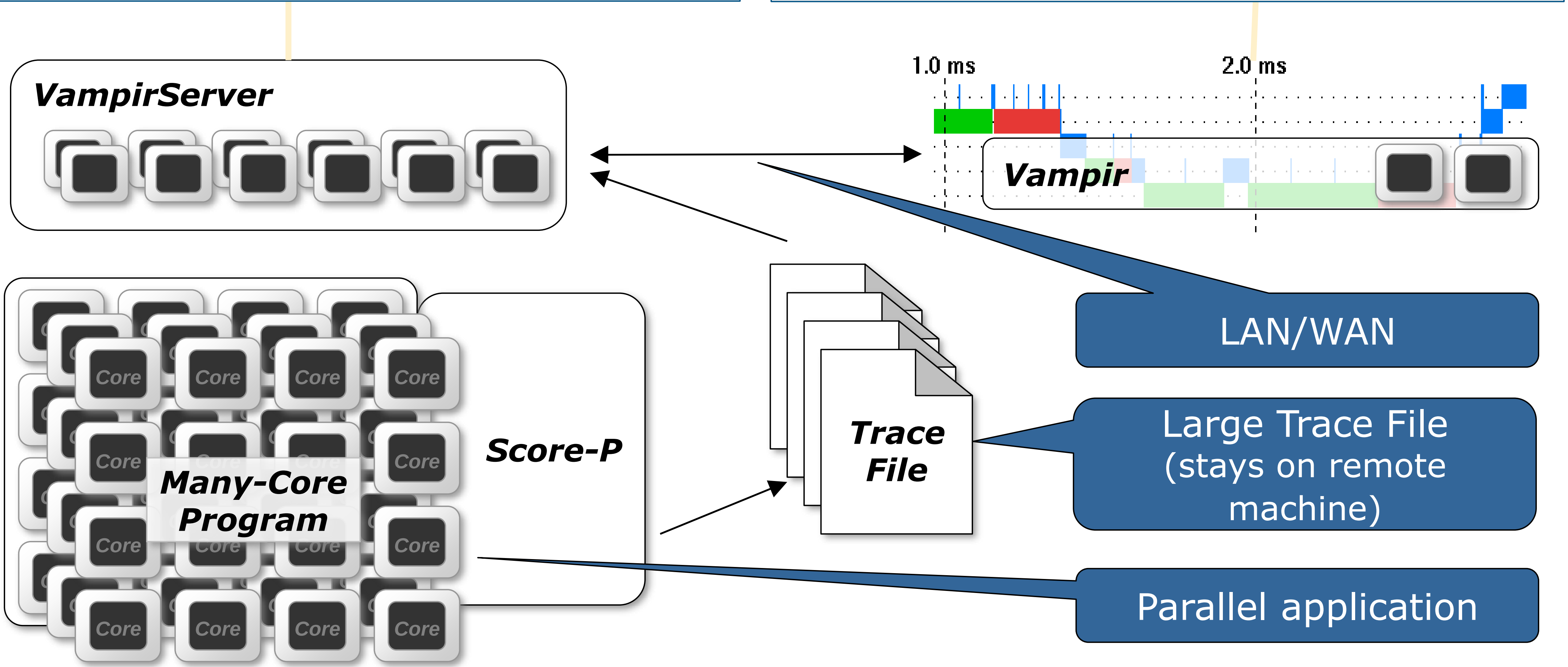
Thread parallel

Visualization Modes (2)

On local machine with remote VampirServer

```
% vampirserver start
```

```
% vampir
```



Main Performance Charts of Vampir

Timeline Charts

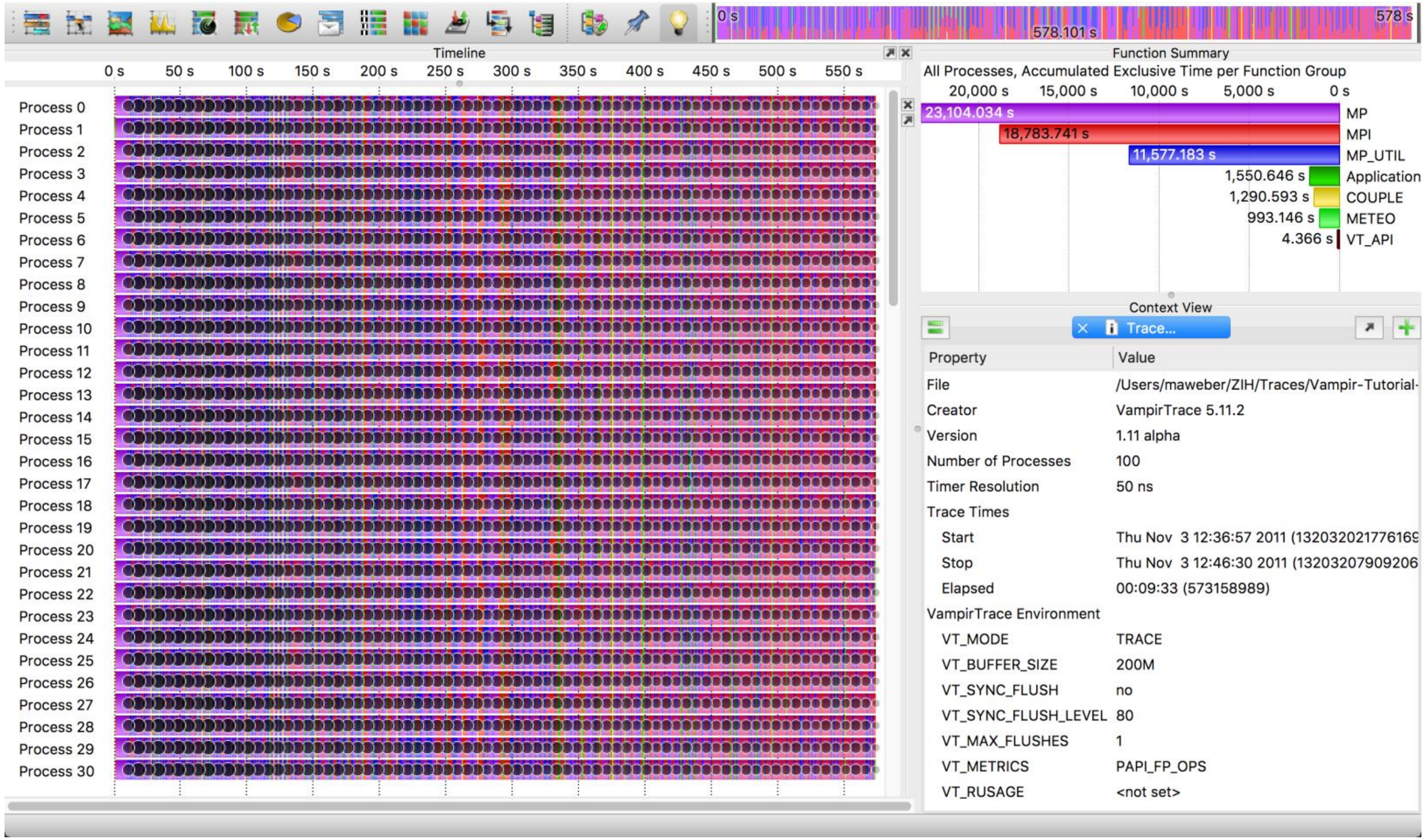
- Master Timeline → *all threads' activities*
- Process Timeline → *single thread's activities*
- Summary Timeline → *all threads' function call statistics*
- Performance Radar → *all threads' performance metrics*
- Counter Data Timeline → *single threads' performance metrics*
- I/O Timeline → *all threads' I/O activities*

Summary Charts

- Function Summary
- Message Summary
- I/O Summary
- Process Summary
- Communication Matrix View
- Call Tree

Vampir Case Study: Analyzing Load Imbalance in COSMO-SPECS

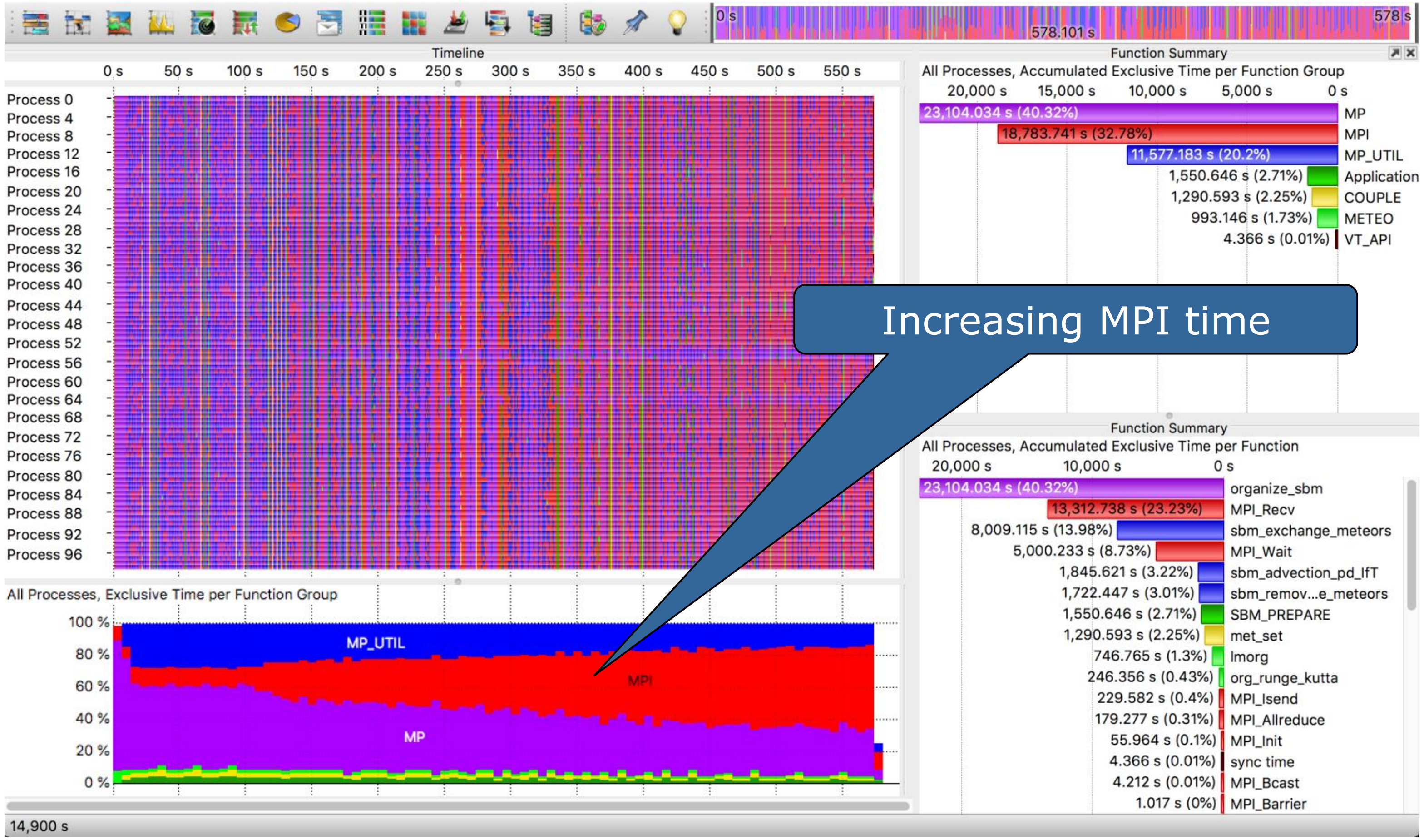
COSMO-SPECS



Property	Value
File	/Users/maweber/ZIH/Traces/Vampir-Tutorial-
Creator	VampirTrace 5.11.2
Version	1.11 alpha
Number of Processes	100
Timer Resolution	50 ns
Trace Times	
Start	Thu Nov 3 12:36:57 2011 (132032021776169
Stop	Thu Nov 3 12:46:30 2011 (13203207909206
Elapsed	00:09:33 (573158989)
VampirTrace Environment	
VT_MODE	TRACE
VT_BUFFER_SIZE	200M
VT_SYNC_FLUSH	no
VT_SYNC_FLUSH_LEVEL	80
VT_MAX_FLUSHES	1
VT_METRICS	PAPI_FP_OPS
VT_RUSAGE	<not set>

- Weather forecast code COSMO-SPECS
- Run with 100 processes
- COSMO: weather model (METEO group)
- SPECS: microphysics for accurate cloud calculation (MP and MP_UTIL group)
- Coupling of both models done in COUPLE group

COSMO-SPECS



Function Summary

All Processes, Accumulated Exclusive Time per Function Group

Time	Function Group
23,104.034 s (40.32%)	MP
18,783.741 s (32.78%)	MPI
11,577.183 s (20.2%)	MP_UTIL
1,550.646 s (2.71%)	Application
1,290.593 s (2.25%)	COUPLE
993.146 s (1.73%)	METEO
4.366 s (0.01%)	VT_API

Function Summary

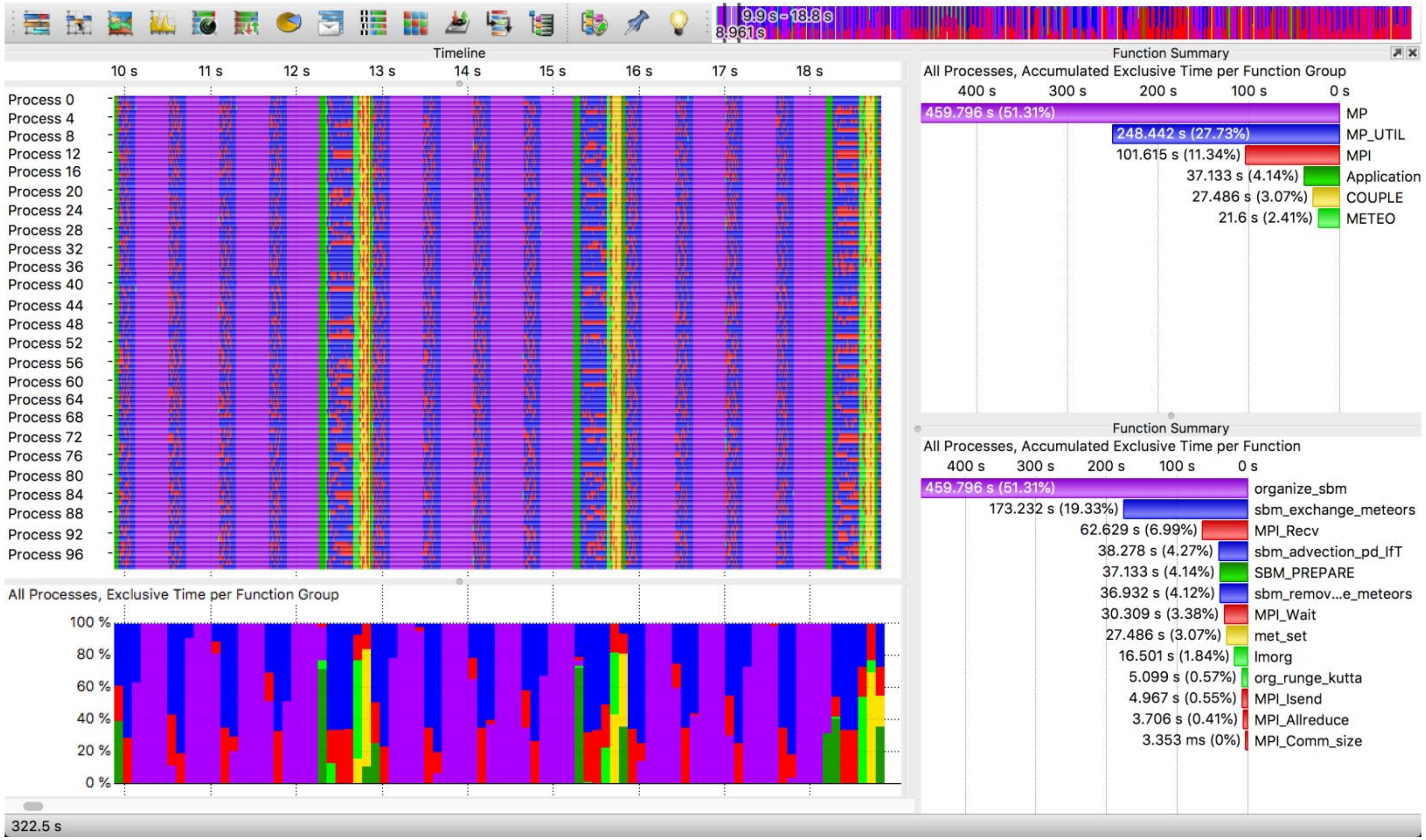
All Processes, Accumulated Exclusive Time per Function

Time	Function
23,104.034 s (40.32%)	organize_sbm
13,312.738 s (23.23%)	MPI_Recv
8,009.115 s (13.98%)	sbm_exchange_meteors
5,000.233 s (8.73%)	MPI_Wait
1,845.621 s (3.22%)	sbm_advection_pd_lfT
1,722.447 s (3.01%)	sbm_remov...e_meteors
1,550.646 s (2.71%)	SBM_PREPARE
1,290.593 s (2.25%)	met_set
746.765 s (1.3%)	lmorg
246.356 s (0.43%)	org_runge_kutta
229.582 s (0.4%)	MPI_Isend
179.277 s (0.31%)	MPI_Allreduce
55.964 s (0.1%)	MPI_Init
4.366 s (0.01%)	sync time
4.212 s (0.01%)	MPI_Bcast
1.017 s (0%)	MPI_Barrier

Increasing MPI time

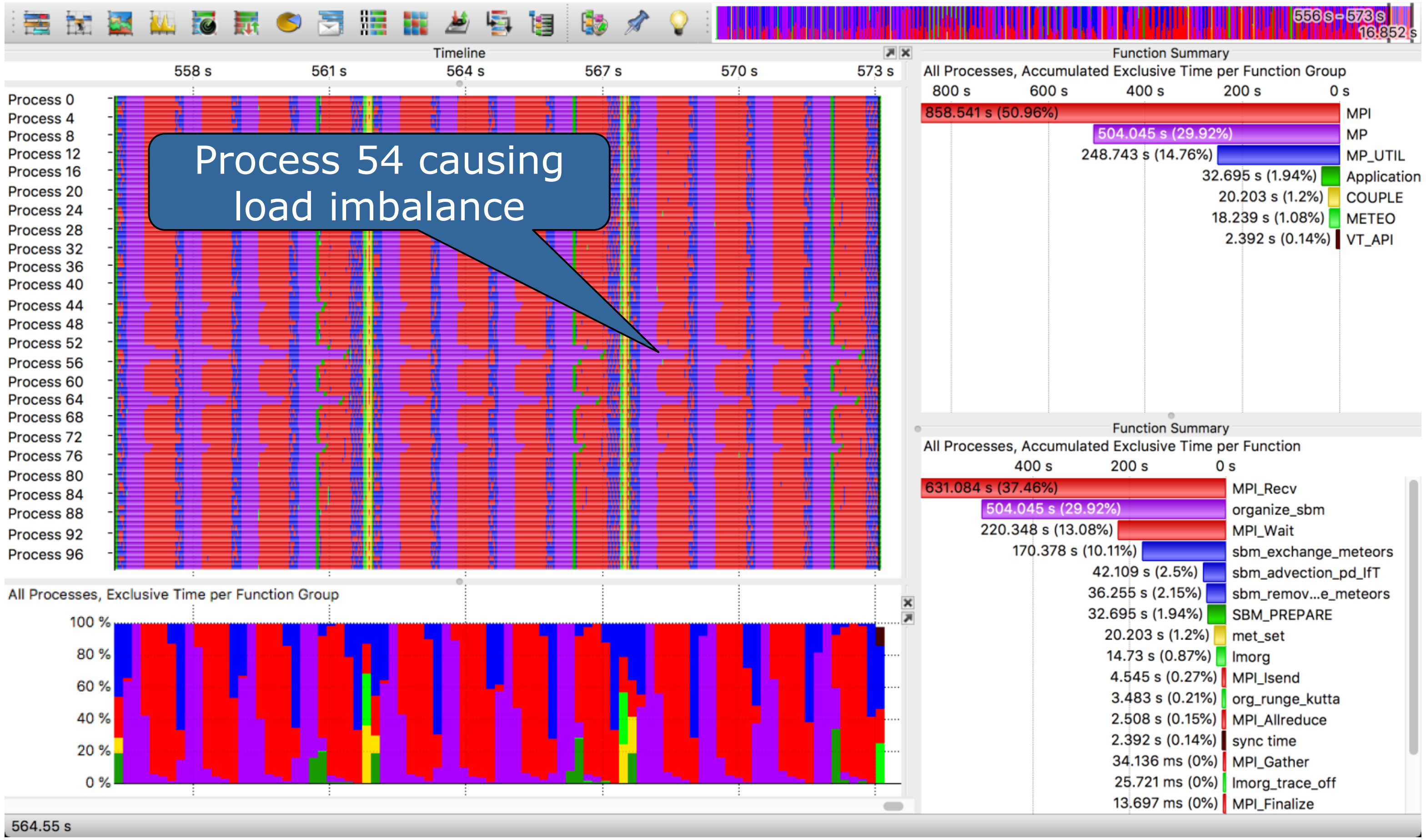
- Compared to METEO, MP and MP_UTIL are very compute intensive, however this is due to more complex calculations and no performance issue
- Problem: >32% of time spent in MPI
- MPI runtime share increases throughout the application run

COSMO-SPECS



- Zoom into the first three iterations
- MP/MP_UTIL perform four sub-steps in one iteration
- Low MPI time share
- Everything is balanced and looks okay

COSMO-SPECS



Process 54 causing load imbalance

Function Summary

All Processes, Accumulated Exclusive Time per Function Group

858.541 s (50.96%)	MPI
504.045 s (29.92%)	MP
248.743 s (14.76%)	MP_UTIL
32.695 s (1.94%)	Application
20.203 s (1.2%)	COUPLE
18.239 s (1.08%)	METEO
2.392 s (0.14%)	VT_API

Function Summary

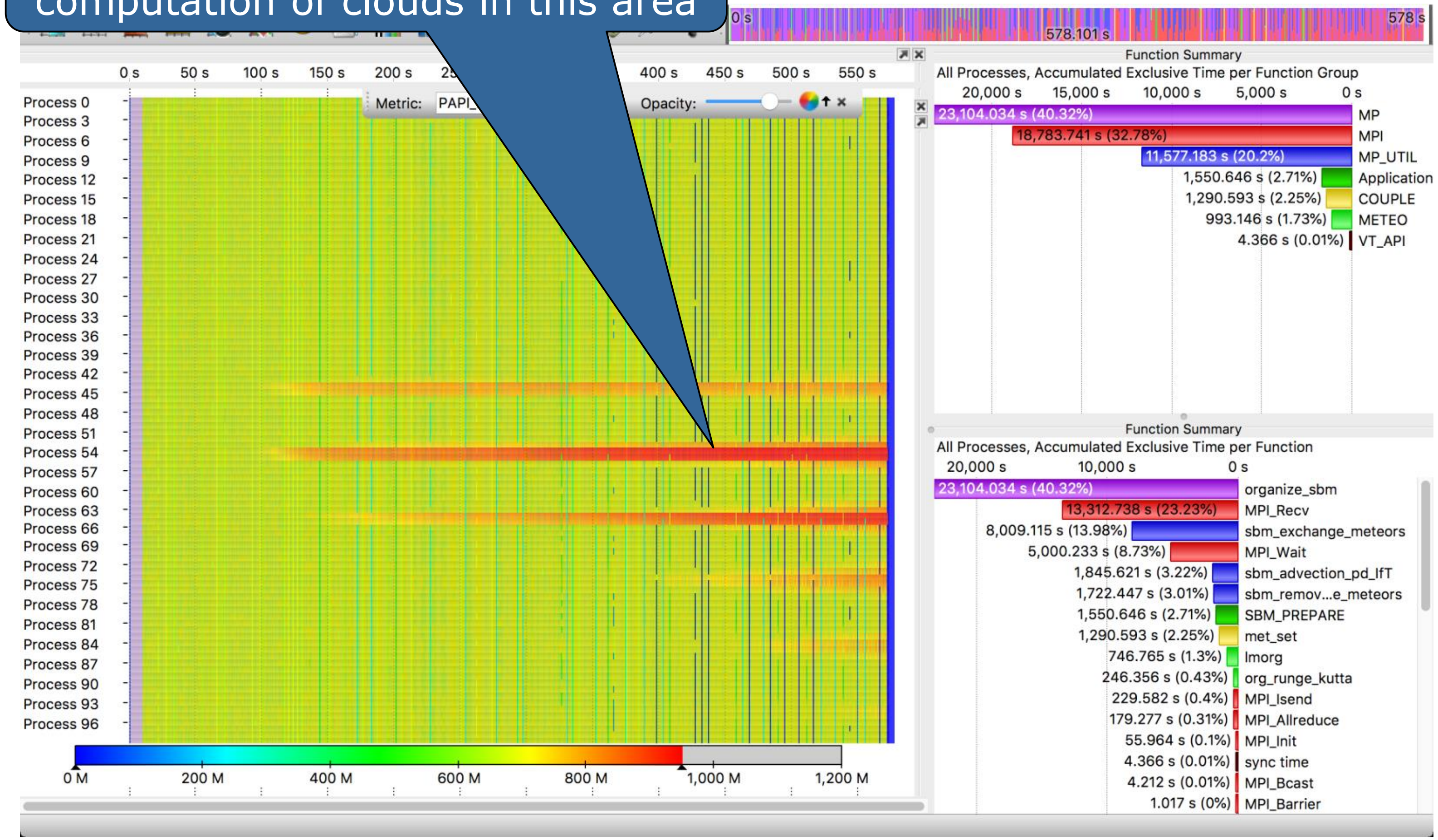
All Processes, Accumulated Exclusive Time per Function

631.084 s (37.46%)	MPI_Recv
504.045 s (29.92%)	organize_sbm
220.348 s (13.08%)	MPI_Wait
170.378 s (10.11%)	sbm_exchange_meteors
42.109 s (2.5%)	sbm_advection_pd_lfT
36.255 s (2.15%)	sbm_remov...e_meteors
32.695 s (1.94%)	SBM_PREPARE
20.203 s (1.2%)	met_set
14.73 s (0.87%)	lmorg
4.545 s (0.27%)	MPI_Isend
3.483 s (0.21%)	org_runge_kutta
2.508 s (0.15%)	MPI_Allreduce
2.392 s (0.14%)	sync time
34.136 ms (0%)	MPI_Gather
25.721 ms (0%)	lmorg_trace_off
13.697 ms (0%)	MPI_Finalize

- Zoom into the last three iterations
- Very high MPI time share (>50%)
- Large load imbalance caused by MP functions around **`Process 54`** and **`Process 64`**

COSMO-SPECS

High FLOPs rates due to computation of clouds in this area



Function Summary

All Processes, Accumulated Exclusive Time per Function Group

23,104.034 s (40.32%)	MP
18,783.741 s (32.78%)	MPI
11,577.183 s (20.2%)	MP_UTIL
1,550.646 s (2.71%)	Application
1,290.593 s (2.25%)	COUPLE
993.146 s (1.73%)	METEO
4.366 s (0.01%)	VT_API

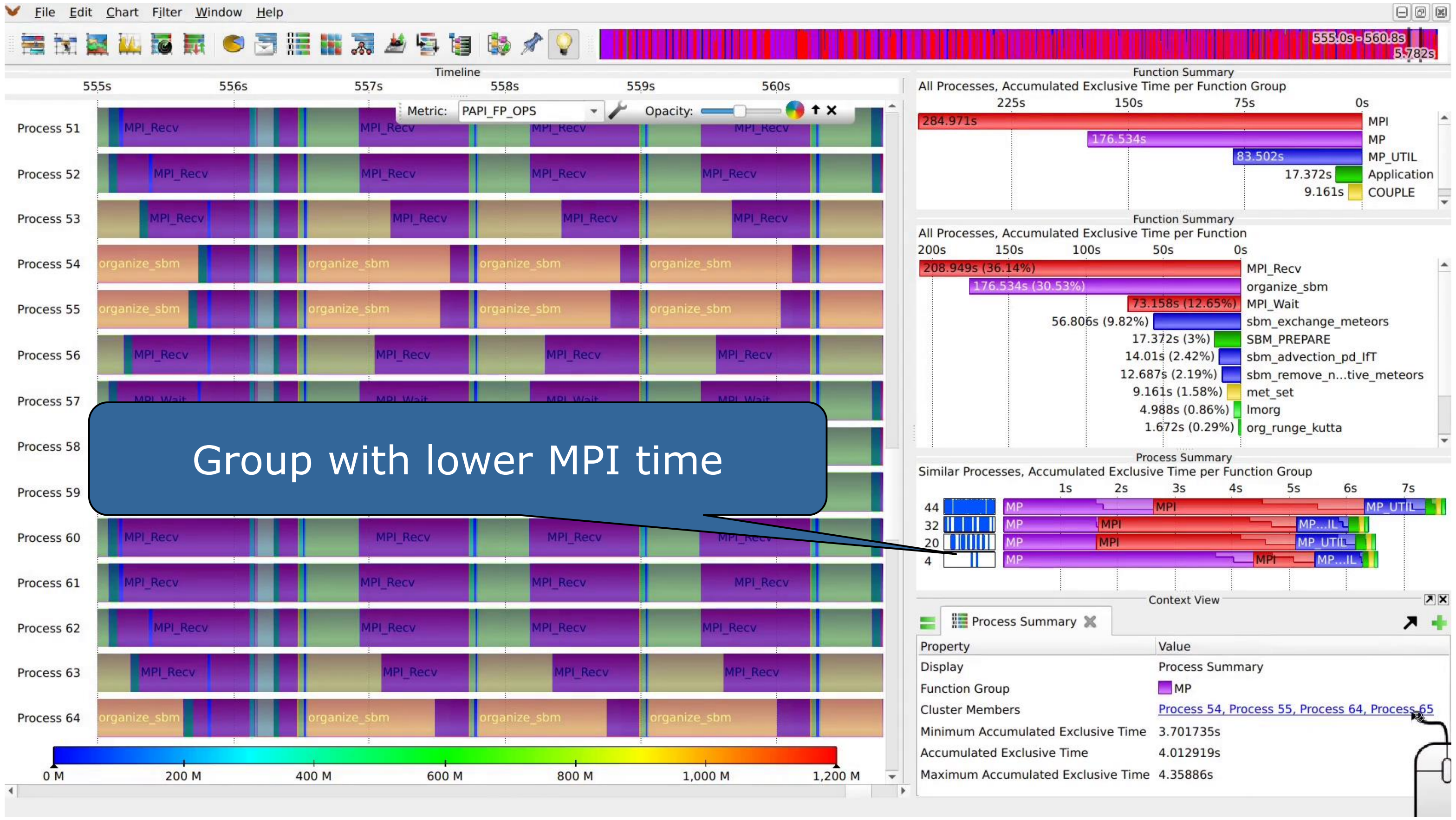
Function Summary

All Processes, Accumulated Exclusive Time per Function

23,104.034 s (40.32%)	organize_sbm
13,312.738 s (23.23%)	MPI_Recv
8,009.115 s (13.98%)	sbm_exchange_meteors
5,000.233 s (8.73%)	MPI_Wait
1,845.621 s (3.22%)	sbm_advection_pd_lfT
1,722.447 s (3.01%)	sbm_remov...e_meteors
1,550.646 s (2.71%)	SBM_PREPARE
1,290.593 s (2.25%)	met_set
746.765 s (1.3%)	lmorg
246.356 s (0.43%)	org_runge_kutta
229.582 s (0.4%)	MPI_Isend
179.277 s (0.31%)	MPI_Allreduce
55.964 s (0.1%)	MPI_Init
4.366 s (0.01%)	sync time
4.212 s (0.01%)	MPI_Bcast
1.017 s (0%)	MPI_Barrier

- **`PAPI_FP_OPS`** counter showing higher FLOPs rates on processes causing the imbalance
- Reason for imbalance: Static grid used for distribution of processes. Depending on the weather, expensive cloud computations (MP group) may be only necessary on some processes

COSMO-SPECS



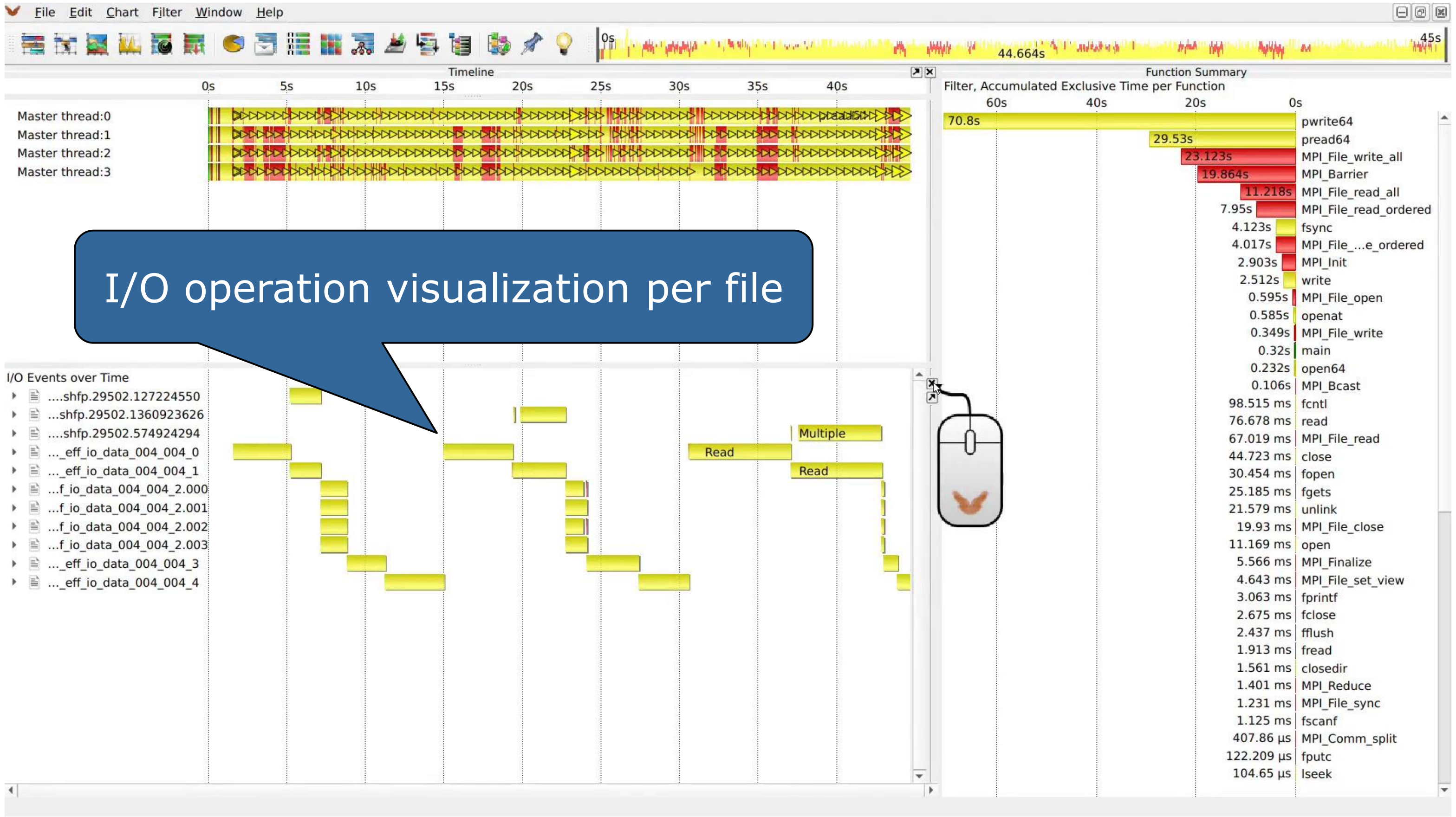
Group with lower MPI time

Property	Value
Display	Process Summary
Function Group	MP
Cluster Members	Process 54, Process 55, Process 64, Process 65
Minimum Accumulated Exclusive Time	3.701735s
Accumulated Exclusive Time	4.012919s
Maximum Accumulated Exclusive Time	4.35886s

- Process Summary helps finding outliers
- Groups processes by their behavior (similar call/duration profile)
- Number of expected groups is variable
- In this case 4 yields the best results

Vampir Showcase:
Analyzing Multilayer File I/O Applications

Multilayer File I/O Application


File Edit Chart Filter Window Help
0s
44.664s
45s
Timeline
0s 5s 10s 15s 20s 25s 30s 35s 40s
Master thread:0
Master thread:1
Master thread:2
Master thread:3
pread64
I/O Events over Time
....shfp.29502.127224550
...shfp.29502.1360923626
....shfp.29502.574924294
..._eff_io_data_004_004_0
..._eff_io_data_004_004_1
...f_io_data_004_004_2.000
...f_io_data_004_004_2.001
...f_io_data_004_004_2.002
...f_io_data_004_004_2.003
..._eff_io_data_004_004_3
..._eff_io_data_004_004_4
Multiple
Read
Read
Function Summary
Filter, Accumulated Exclusive Time per Function
60s 40s 20s 0s

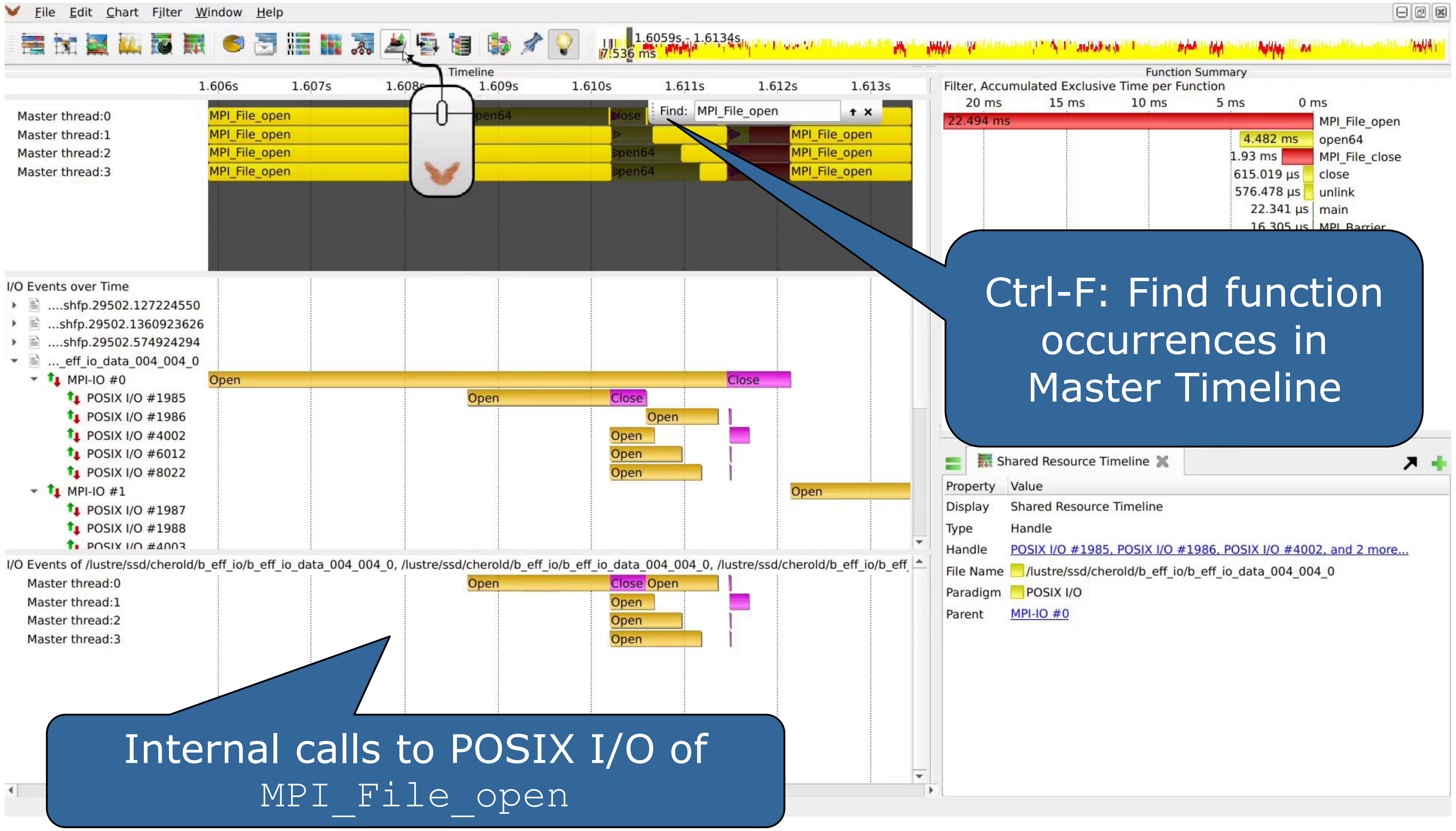
I/O operation visualization per file

70.8s	pwrite64
29.53s	pread64
23.123s	MPI_File_write_all
19.864s	MPI_Barrier
11.218s	MPI_File_read_all
7.95s	MPI_File_read_ordered
4.123s	fsync
4.017s	MPI_File_...e_ordered
2.903s	MPI_Init
2.512s	write
0.595s	MPI_File_open
0.585s	openat
0.349s	MPI_File_write
0.32s	main
0.232s	open64
0.106s	MPI_Bcast
98.515 ms	fcntl
76.678 ms	read
67.019 ms	MPI_File_read
44.723 ms	close
30.454 ms	fopen
25.185 ms	fgets
21.579 ms	unlink
19.93 ms	MPI_File_close
11.169 ms	open
5.566 ms	MPI_Finalize
4.643 ms	MPI_File_set_view
3.063 ms	fprintf
2.675 ms	fclose
2.437 ms	fflush
1.913 ms	fread
1.561 ms	closedir
1.401 ms	MPI_Reduce
1.231 ms	MPI_File_sync
1.125 ms	fscanf
407.86 µs	MPI_Comm_split
122.209 µs	fputc
104.65 µs	lseek

- IO bandwidth benchmark b_eff_io
- Measures achievable I/O bandwidth of parallel MPI-I/O applications

- Shared Resource Timeline offers a per file and per thread view on File I/O operations

Multilayer File I/O Application



Property	Value
Display	Shared Resource Timeline
Type	Handle
Handle	POSIX I/O #1985, POSIX I/O #1986, POSIX I/O #4002, and 2 more...
File Name	/lustre/ssd/cherold/b_eff_io/b_eff_io_data_004_004_0
Paradigm	POSIX I/O
Parent	MPI-IO #0

Ctrl-F: Find function occurrences in Master Timeline

Internal calls to POSIX I/O of `MPI_File_open`

- Visualization of I/O on multiple layers (MPI & POSIX)

- Example: behavior of **`MPI_File_open`**
- Internally uses POSIX **`open`** for opening the actual file on disk
- Multiple consecutive calls to **`open`** and **`close`** on master rank

Summary

- Vampir
  - Interactive trace visualization and analysis of:
    - MPI, OpenMP, CUDA applications
    - File I/O
    - Hardware performance counters
    - (Collective) communication
  - Intuitive browsing and zooming
  - Available for Linux, Windows, and macOS

- VampirServer
  - Scalable to large trace data sizes (20 TiByte)
  - Scalable to high parallelism (200,000 processes)

https://vampir.eu/