

Performance Analysis Tools

Christian Iwainsky

Iwainsky@rz.rwth-aachen.de

25.3.2010

1. **Performance Analysis Tools in Aachen, an Overview**
2. **Vampir in Detail**
3. **How to use Vampir in Aachen**

How does one gain insight?

Data has to be presented to the developer in an understandable way in order to gain insight!

Data can be

- **broken up by time**

Example: Function entries and exits are displayed on a timeline

- **attributed to graphs representing properties of the source code**

Example: Display function execution times on the call graph

- **etc...**

Tools Overview

Serial	Oracle Performance Analyzer	✓
	GNU gprof	✗
	Intel VTune	✗
	Acumem	LATER
OpenMP / Shared Memory	Oracle Performance Analyzer	✓
	Intel Threading Tools	✗
	Intel Parallel Studio	LATER
	Vampir	LATER
MPI	Intel Trace Collector and Analyzer	✓
	Vampir	LATER
	Oracle Performance Analyzer	✓
	Microsoft Event Tracing for MS-MPI	LATER
	Scalasca	LATER

Facts about Acumem:

- Commercial (www.acumem.com)
- Sampling based
- Tailored towards optimizing memory access
- Currently available for only Linux `module load acumem`
- (min) Three steps
 - sample application
e.g.: `sample -r -o sample.smp jacobi < input`
 - analyze application for specific hardware
e.g.: `report -o myReport -i sample.smp`
 - review results
e.g.: `firefox myReport.html`

Also graphical launcher:
`acumem`

Acumem ThreadSpotter™

Acumem ThreadSpotter™ is a tool to quickly analyze an application for a range of performance problems, particularly related to multicore optimization.

[Read more...](#) [Manual](#)

Open the Report



Your application

Application: jacobi.exe



Memory Bandwidth

The memory bus transports data between the main memory and the processor. The capacity of the memory bus is limited. Abuse of this resource limits application scalability.

[Manual: Bandwidth](#)



Memory Latency

The regularity of the application's memory accesses affects the efficiency of the hardware prefetcher. Irregular accesses causes cache misses, which forces the processor to wait a lot for data to arrive.

[Manual: Cache misses](#) [Manual: Prefetching](#)



Data Locality

Failure to pay attention to data locality has several negative effects. Caches will be filled with unused data, and the memory bandwidth will waste transporting unused data.

[Manual: Locality](#)



Thread Communication / Interaction

Several threads contending over ownership of data in their respective caches causes the different processor cores to stall.

[Manual: Multithreading](#)

This means that your application shows opportunities to:

Avoid major processor stalls due to irregular access patterns

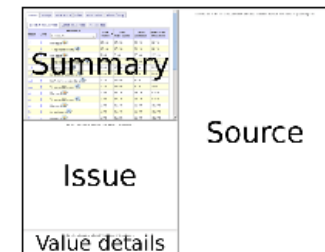
[Read more...](#)

Next Steps

The prepared report is divided into sections.

- Select the tab **Summary** to see global statistics for the entire application.
- Select the tabs **Bandwidth Issues**, **Latency Issues** and **MT Issues** to browse through the detected problems.
- Select the tab **Loops** to browse through statistics and detected problems loop by loop.

The Issue and Source windows contain details and annotated source code for the detected problems.



Resources

Manual

[Table of Contents](#)

[Overview](#)

[Optimization Workflow](#)

[Concepts](#)

[Reading the Report](#)

[Issue Reference](#)

Acumem Web Site

Report Layout

Issues | **Loops** | **Summary** | **Files** | **Execution** | **About/Help**

Bandwidth Issues | **Latency Issues** | **Multi-Threading Issues**

#	Issue type	% of bandwidth	% of fetches	% of write-backs	ut
8	Fetch utilization	9.7%	11.4%	52.8%	0.0
9	Loop fusion	9.7%	11.4%	52.8%	0.0
10	Non-temporal store possible	9.7%	11.4%	52.8%	0.0
1	Fetch hot-spot	3.3%	9.5%	0.0%	10
2	Fetch utilization	3.1%	7.3%	0.0%	51

Issue #8: Fetch utilization

This instruction group also show symptoms of: Fetch hot-spot, Write back hot-spot.

Statistics for instructions of this issue

Instructions involved in this issue

Instructions previously writing to related data

Loop statistics

Loop instructions

Copyright (c) 2007-2009 Acumem AB. All Rights Reserved.
Patents pending.

Select an issue or a loop in the issue or loop tables.

Summary/Overview

```
/*
*****
* Subroutine HelmholtzJ
* Solves poisson equation on rectangular grid assuming
* (1) Uniform discretization in each direction, and
* (2) Dirichlet boundary conditions
*
* Jacobi method is used in this routine
*
* Input : n,m   Number of grid points in the X/Y directions
*         dx,dy Grid spacing in the X/Y directions
*         alpha Helmholtz eqn. coefficient
*         omega Relaxation factor
*         f(n,m) Right hand side function
*         u(n,m) Dependent variable/Solution
*         tolerance Tolerance for iterative solver
*         maxit Maximum number of iterations
*
* Output : u(n,m) - Solution
*****
*/
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "jacobi.h"

#define U(j,i) afU[(j) - data->iRowFirst] * data->iCol:
#define F(j,i) afF[(j) - data->iRowFirst] * data->iCol:
```

Report Layout

The screenshot displays the PPces Performance Analysis Tools interface. The top navigation bar includes tabs for Issues, Loops, Summary, Files, Execution, and About/Help. The 'Issues' tab is active, showing a table of performance issues. The table has columns for Issue type, % of bandwidth, % of fetches, % of write-backs, and a 'ut' column. The issues are listed in a table with a filter set to 'All'.

#	Issue type	% of bandwidth	% of fetches	% of write-backs	ut
8	Fetch utilization	9.7%	11.4%	52.8%	0.0
9	Loop fusion	9.7%	11.4%	52.8%	0.0
10	Non-temporal store possible	9.7%	11.4%	52.8%	0.0
1	Fetch hot-spot	3.3%	9.5%	0.0%	10
2	Fetch utilization	3.1%	7.3%	0.0%	51

Issue #8: Fetch utilization ?
This instruction group also show symptoms of: Fetch hot-spot, Write back hot-spot.

- Statistics for instructions of this issue** ?
- Instructions involved in this issue** ?
- Instructions previously writing to related data** ?
- Loop statistics** ?
- Loop instructions** ?

Copyright (c) 2007-2009 Acumem AB. All Rights Reserved.
Patents pending.

Select an issue or a loop in the issue or loop tables.

Selected issue

```
Line Fetches
1 /*
2 *****
3 * Subroutine HelmholtzJ
4 * Solves poisson equation on rectangular grid assuming
5 * (1) Uniform discretization in each direction, and
6 * (2) Dirichlet boundary conditions
7 *
8 * Jacobi method is used in this routine
9 *
10 * Input : n,m   Number of grid points in the X/Y directions
11 *          dx,dy Grid spacing in the X/Y directions
12 *          alpha Helmholtz eqn. coefficient
13 *          omega Relaxation factor
14 *          f(n,m) Right hand side function
15 *          u(n,m) Dependent variable/Solution
16 *          tolerance Tolerance for iterative solver
17 *          maxit Maximum number of iterations
18 *
19 * Output : u(n,m) - Solution
20 *****
21 */
22 #include <math.h>
23 #include <stdio.h>
24 #include <stdlib.h>
25 #include "jacobi.h"
26
27 #define U(j,i) afU[((j) - data->iRowFirst) * data->iCol:
28 #define F(j,i) afF[((j) - data->iRowFirst) * data->iCol:
```


Report Layout

Issues

Loops

Summary

Files

Execution

About/Help

Bandwidth Issues

Latency Issues

Multi-Threading Issues

#	Issue type	% of bandwidth	% of fetches	% of write-backs	u
8	Fetch utilization	9.7%	11.4%	52.8%	0
9	Loop fusion	9.7%	11.4%	52.8%	0
10	Non-temporal store possible	9.7%	11.4%	52.8%	0
1	Fetch hot-spot	3.3%	9.5%	0.0%	1
2	Fetch utilization	3.1%	7.3%	0.0%	5

Issue #8: Fetch utilization

This instruction group also show symptoms of: Fetch hot-spot, Write back hot-spot.

Code view

Instructions of this issue

Instructions previously writing to related data

Loop statistics

Loop instructions

Copyright (c) 2007-2009 Acumem AB. All Rights Reserved.
Patents pending.

Select an issue or a loop in the issue or loop tables.

/rwthfs/rz/cluster/home/ci841521

/veranstaltungen/PPCES2010/acumem

/jacobi/./jacobi.c

Line

Fetches

Source

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

/*

* Subroutine HelmholtzJ

* Solves poisson equation on rectangular grid assuming

* (1) Uniform discretization in each direction, and

* (2) Dirichlect boundary conditions

*

* Jacobi method is used in this routine

*

* Input : n,m Number of grid points in the X/Y direct:

* dx,dy Grid spacing in the X/Y directions

* alpha Helmholtz eqn. coefficient

* omega Relaxation factor

* f(n,m) Right hand side function

* u(n,m) Dependent variable/Solution

* tolerance Tolerance for iterative solver

* maxit Maximum number of iterations

*

* Output : u(n,m) - Solution

*/

#include <math.h>

#include <stdio.h>

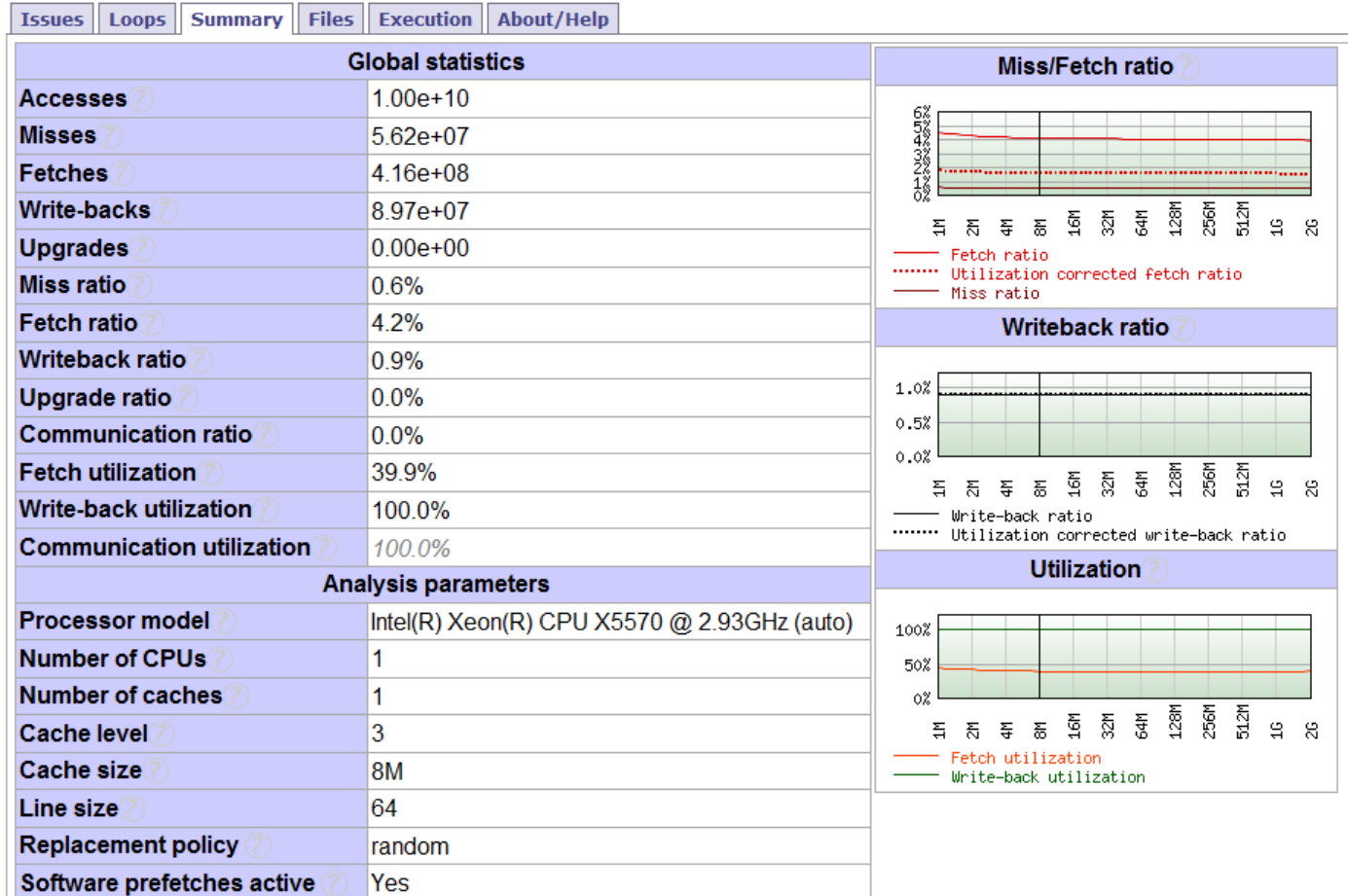
#include <stdlib.h>

#include "jacobi.h"

#define U(j,i) afU[((j) - data->iRowFirst) * data->iCol:

#define F(j,i) afF[((j) - data->iRowFirst) * data->iCol:

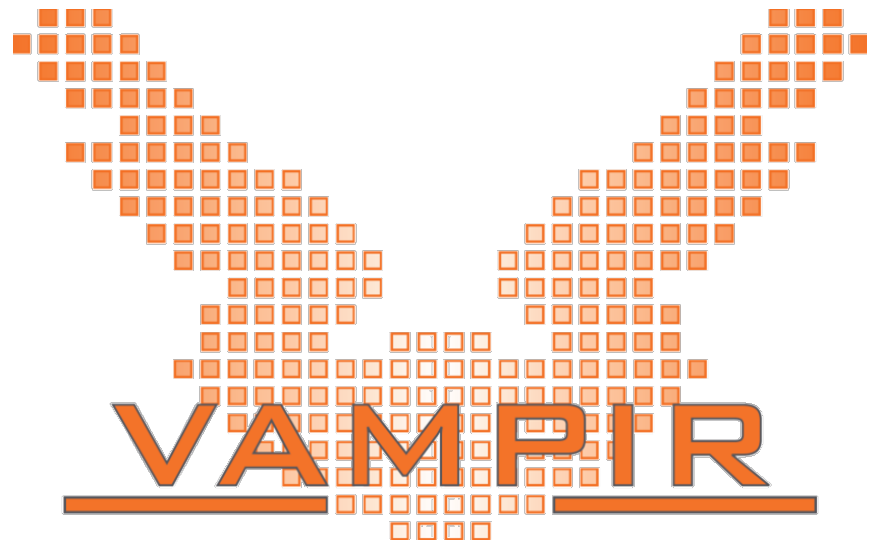
The Summary-View



Copyright (c) 2007-2009 Acumem AB. All Rights Reserved.
Patents pending.

Introduction to Vampir

- Overview of Vampir
- How to use Vampir



Facts about Vampir

- **Actually a tool collection**
 - Vampir Trace freely available
 - Vampir commercial
 - Vampir Server / Vampir Next Generation commercial
- **Developed at the TU Dresden**
- **Many additional recording capabilities**
 - Memory Usage
 - I/O Activity Tracing
 - CPU-counters

Prepare your environment

- **Vampir-trace**

```
module load UNITE vampirtrace
```

- **Vampir (Classic GUI)**

```
module load UNITE vampir
```

- **Vampir Next Generation (Server Client Model)**

```
module load UNITE vampirserver
```

Note:

- **Vampir-Trace is not necessary for visualization of data**
- **Either Vampir or Vampir Next Generation must be loaded for analysis**

To use Vampir the following steps are necessary:

1. Select the correct compiler wrapper

C:

```
vtcc -vt:cc
```

C++:

```
vtcxx -vt:cxx
```

Fortran:

```
vtf90 -vt:f90
```

```
vtf77 -vt:f77
```

2. Instrument your application by recompiling it

```
vtcc -vt:cc gcc hello.c -o hello.exe
```

3. Execute as usual

```
$MPIEXEC -np 2 hello.exe
```

➡ Results in a OTF-file

4. Analyze application

```
vampir vtrace.otf
```

Vampir has two graphical data exploration tools:

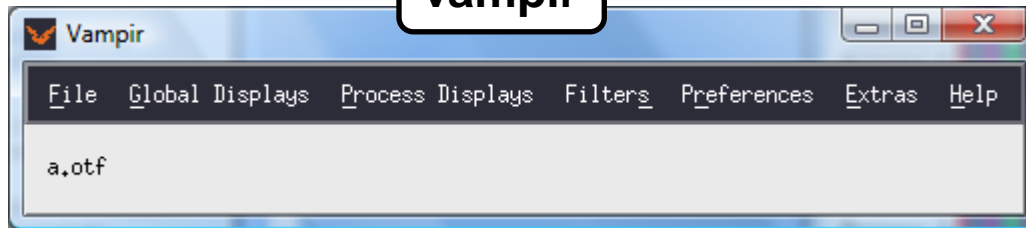
- **vampir**
local execution, no server required
- **vampir server / vampir next generation**
server-client based:
 1. Start the vampir server (default with 5 processes)
`vngd-start.sh`
→ output: server listens on hostname:port
 2. Start the vampir client and connect to the server
`vngd-start.sh [-a hostname -p port] [vtrace.otf]`

Missing arguments will be queried by the GUI

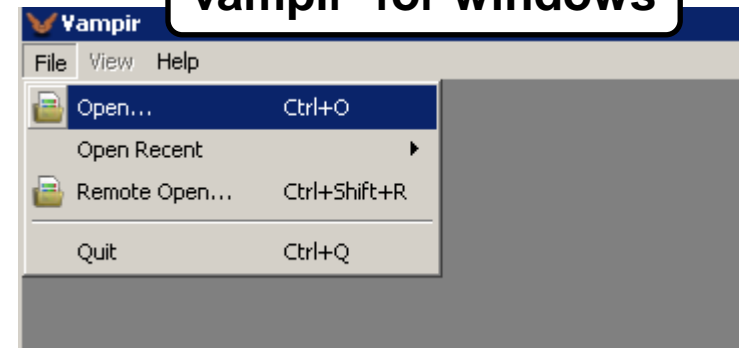
Main Window

- No functional difference
- Vampir Server can handle larger traces more efficiently
- Vampir Server has slightly enhanced graphics

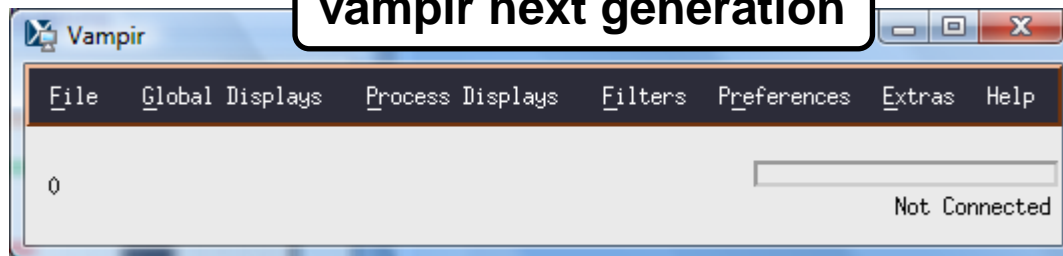
vampir



vampir for windows

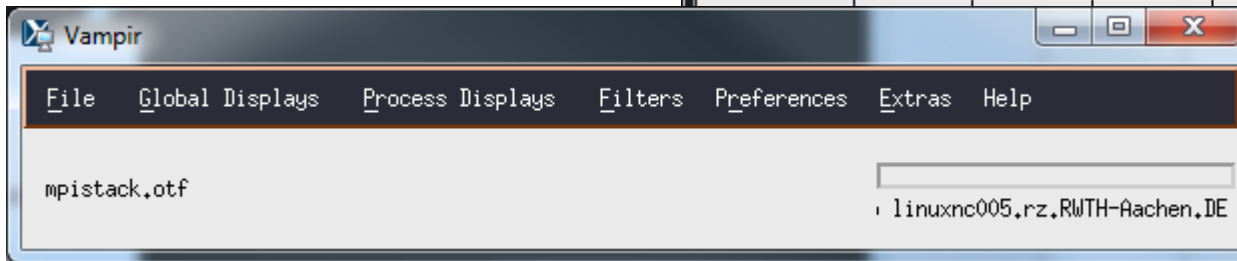
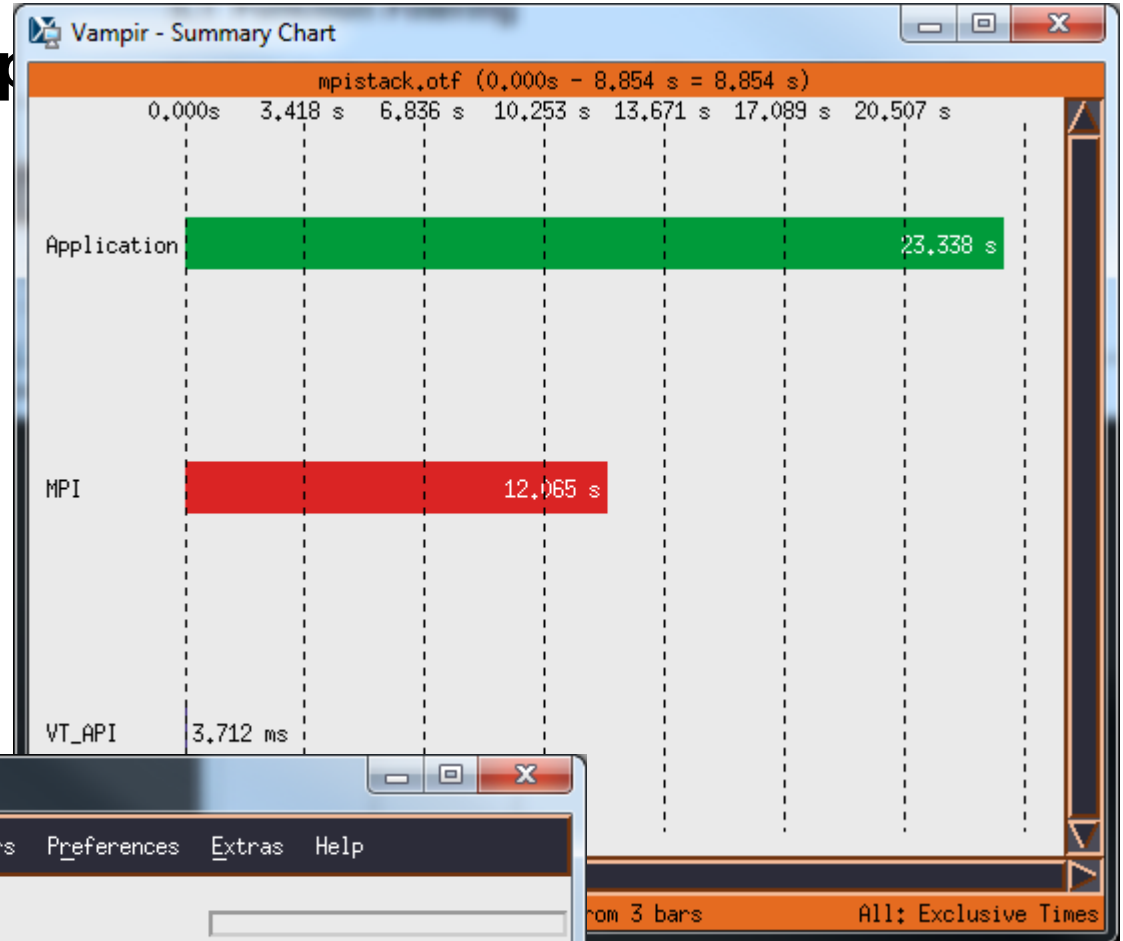


vampir next generation



How to use

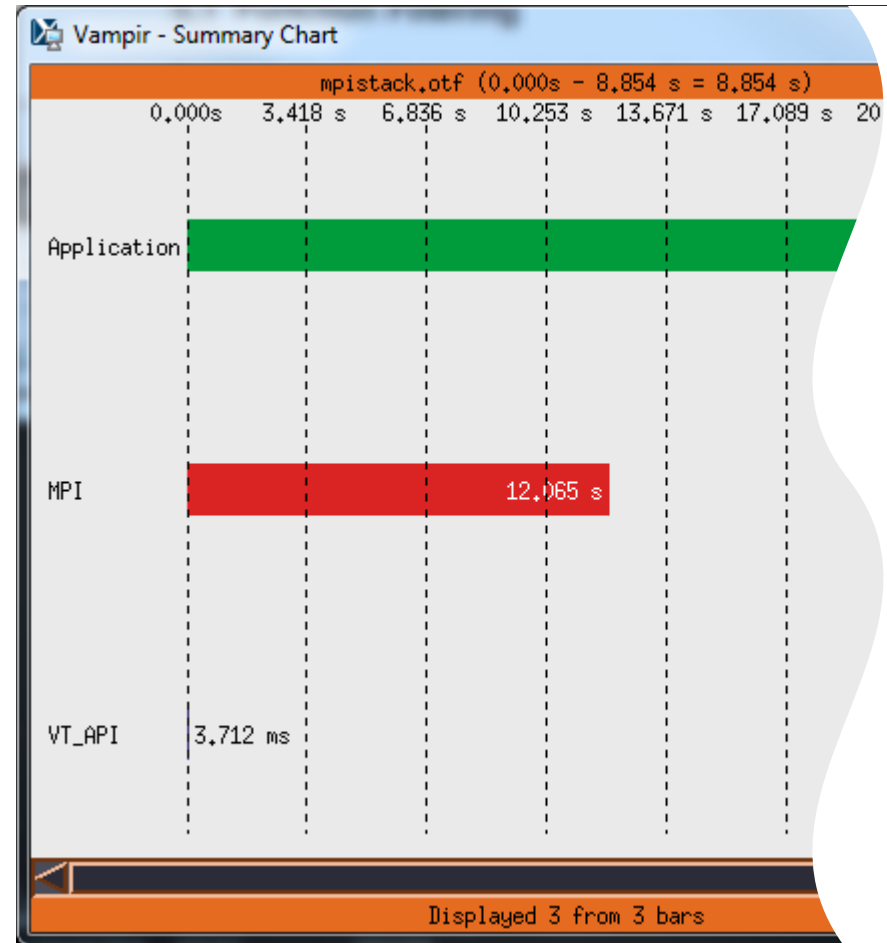
Windows after startup



**First window presented:
Summary chart**

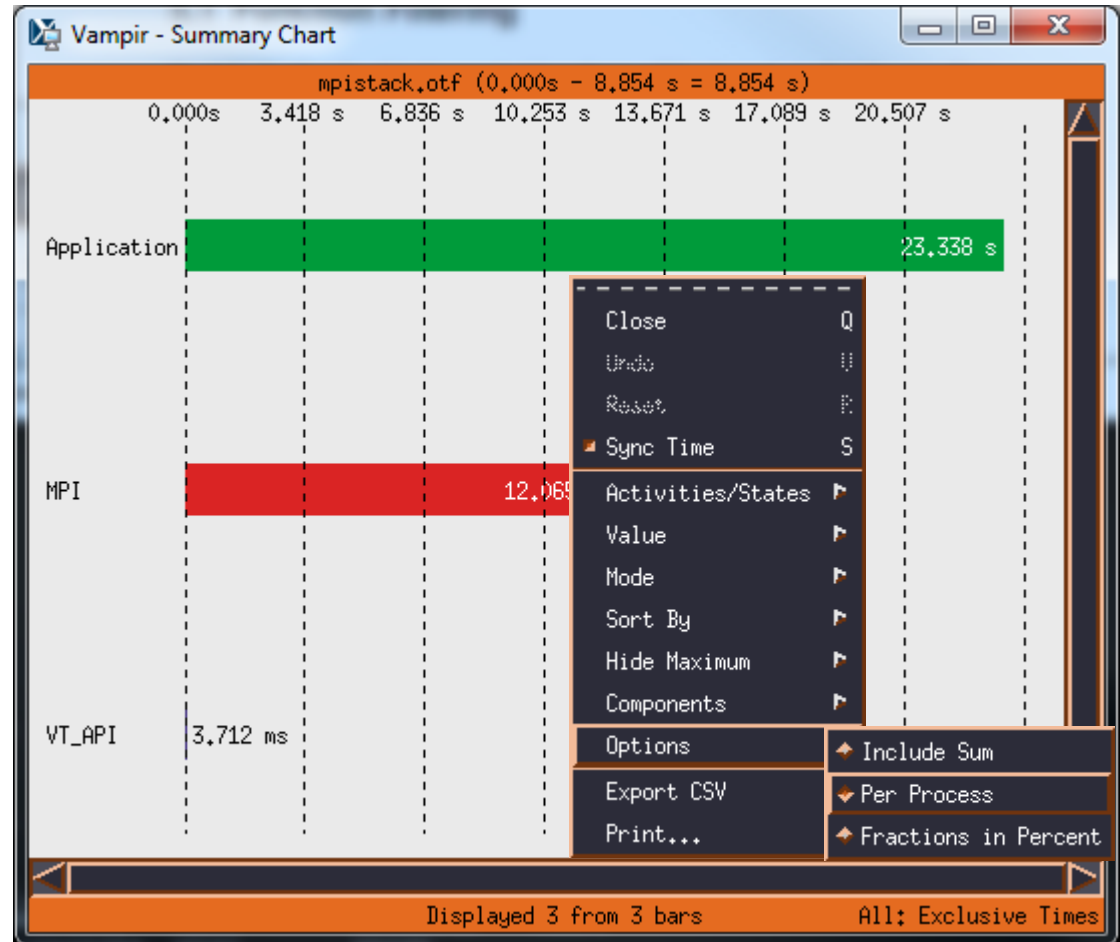
By default:

- all data is grouped
- aggregated for all processes



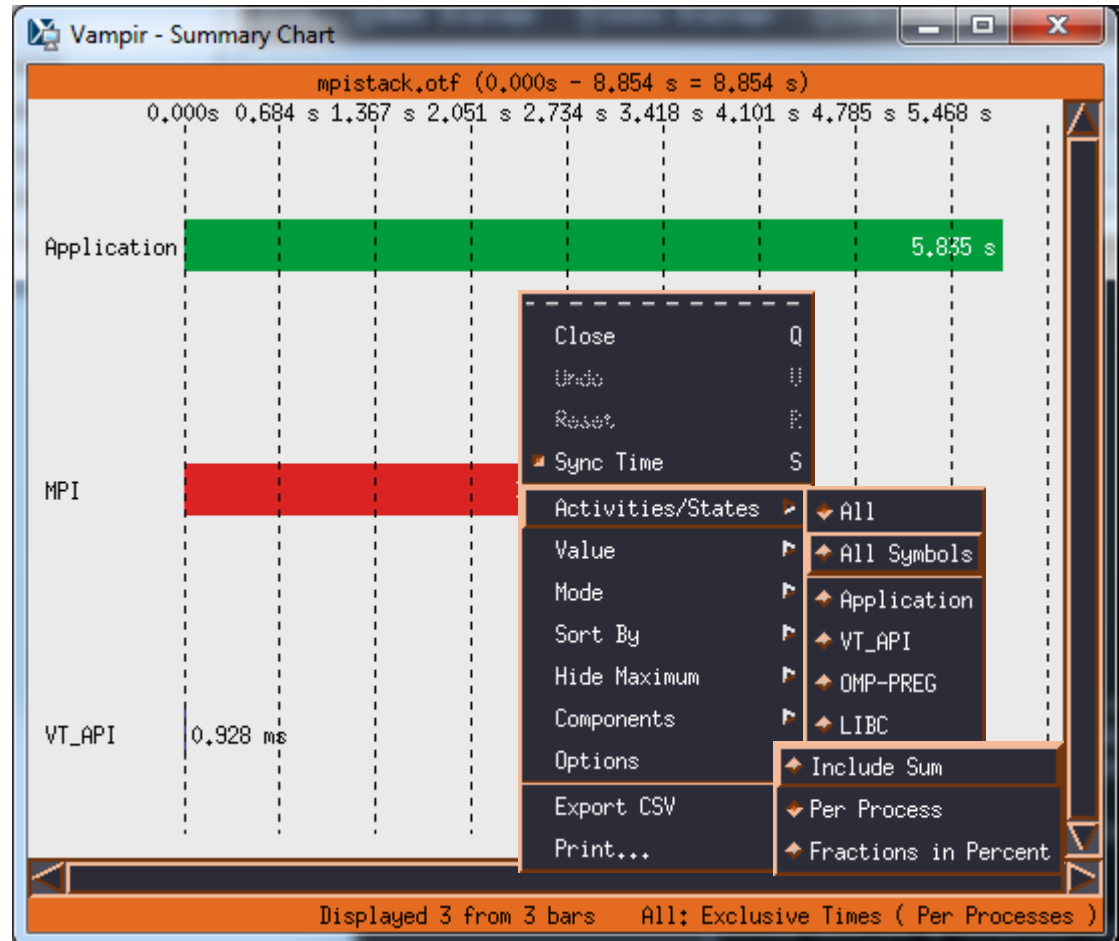
Un-aggregate

- Right-click
 - Options
 - Per Process



Un-group and add Sum

- Right-click
 - Activities/States
 - All symbols
- Right-click
 - Options
 - Include Sum

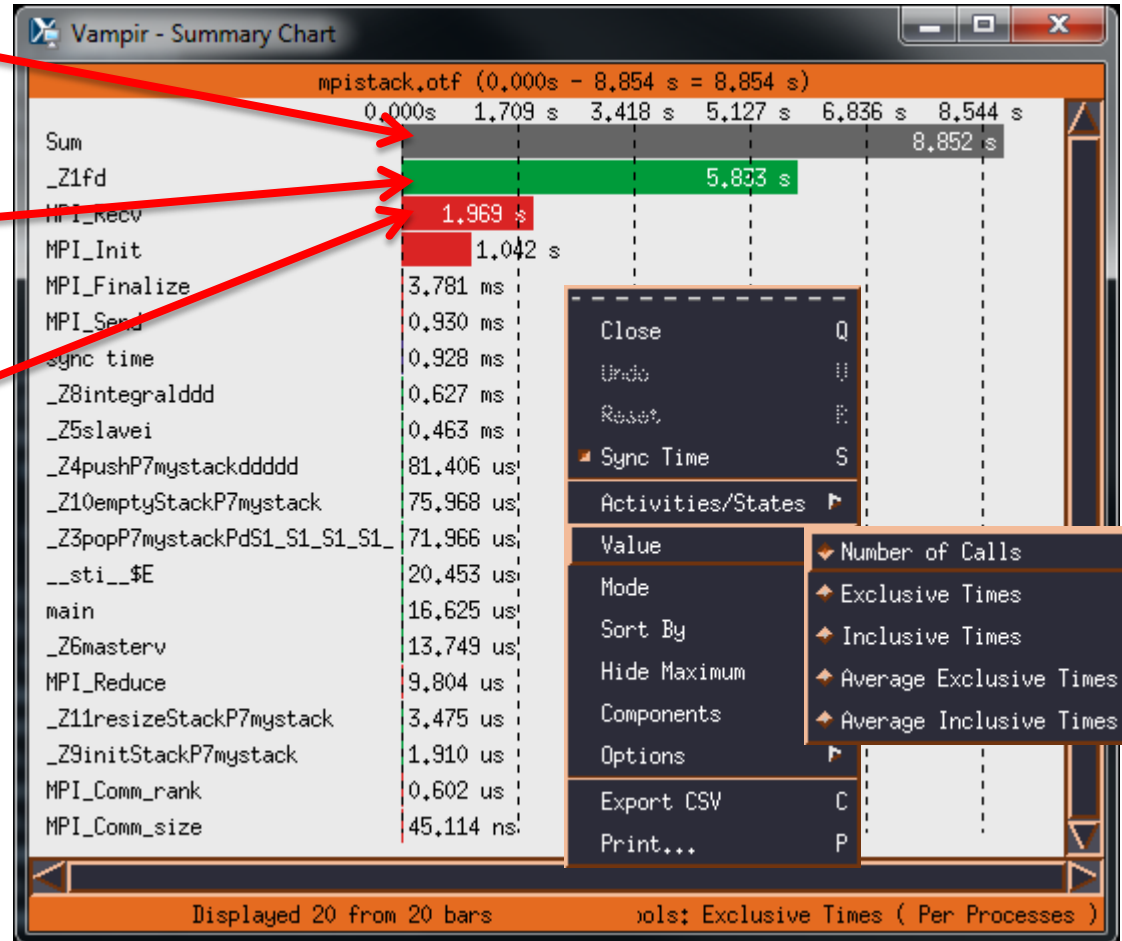


Breakdown of functions by time

Average total runtime of a single process

Average total runtime spent in function "f"

Average total runtime spent in MPI_Recv

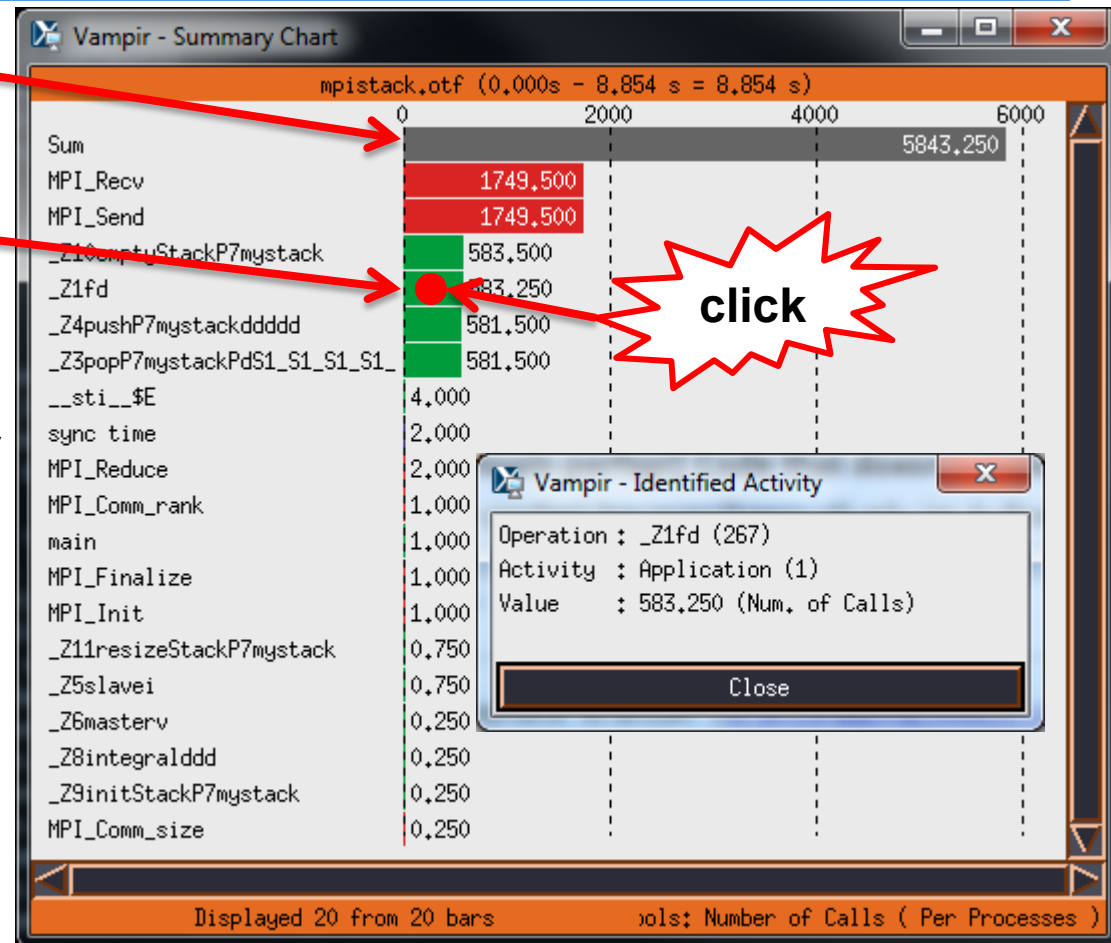


Breakdown of functions by time

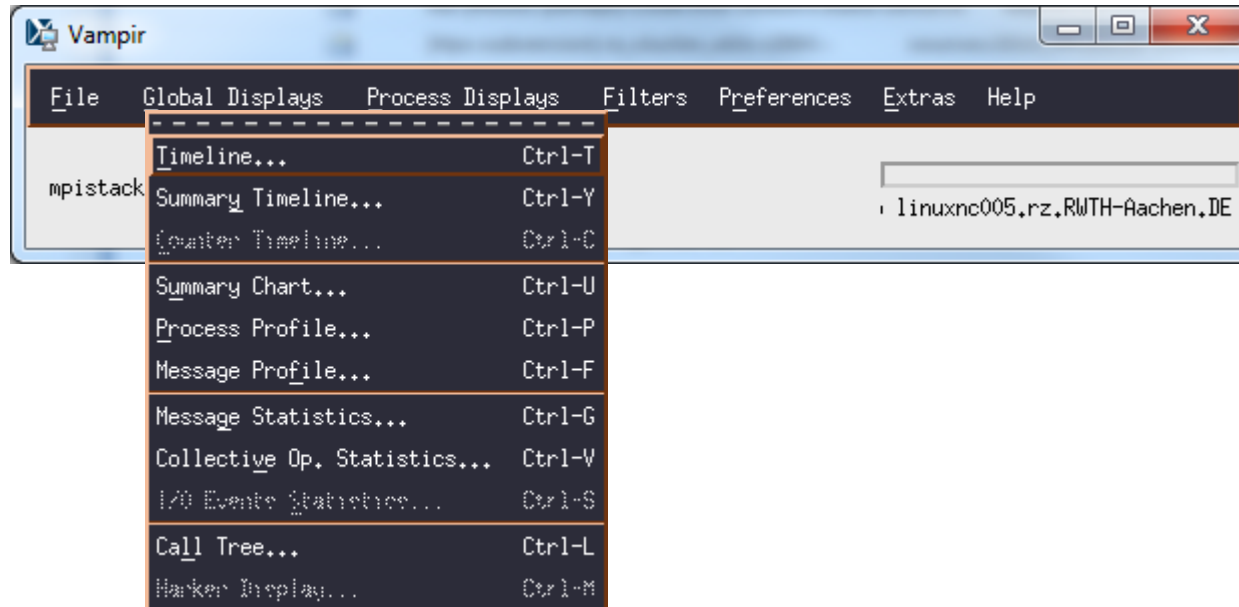
Total number of
function/method calls

Total number
calls to function “f”

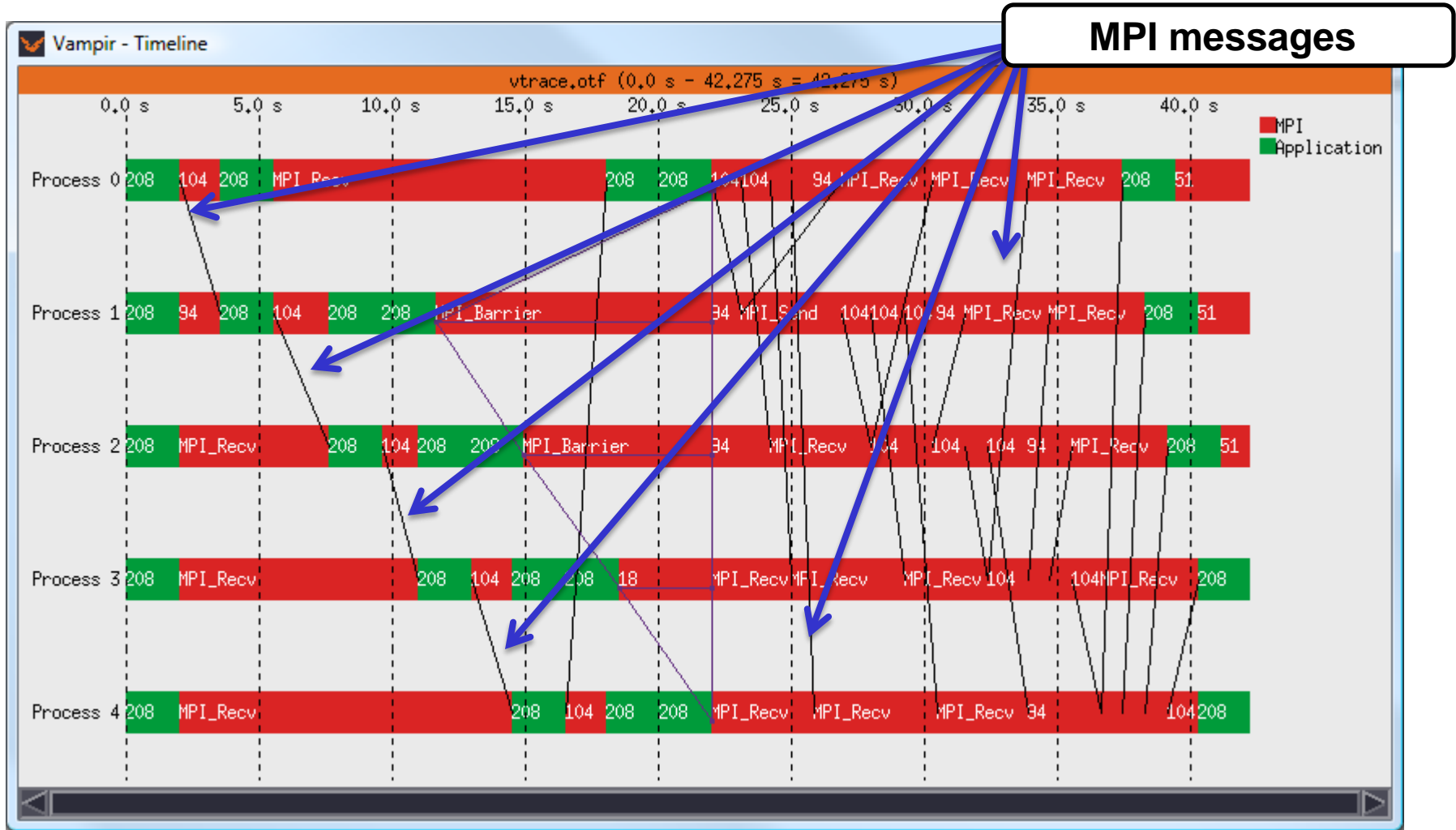
- Left-clicking on any
element in Vampir
will provide more
detail

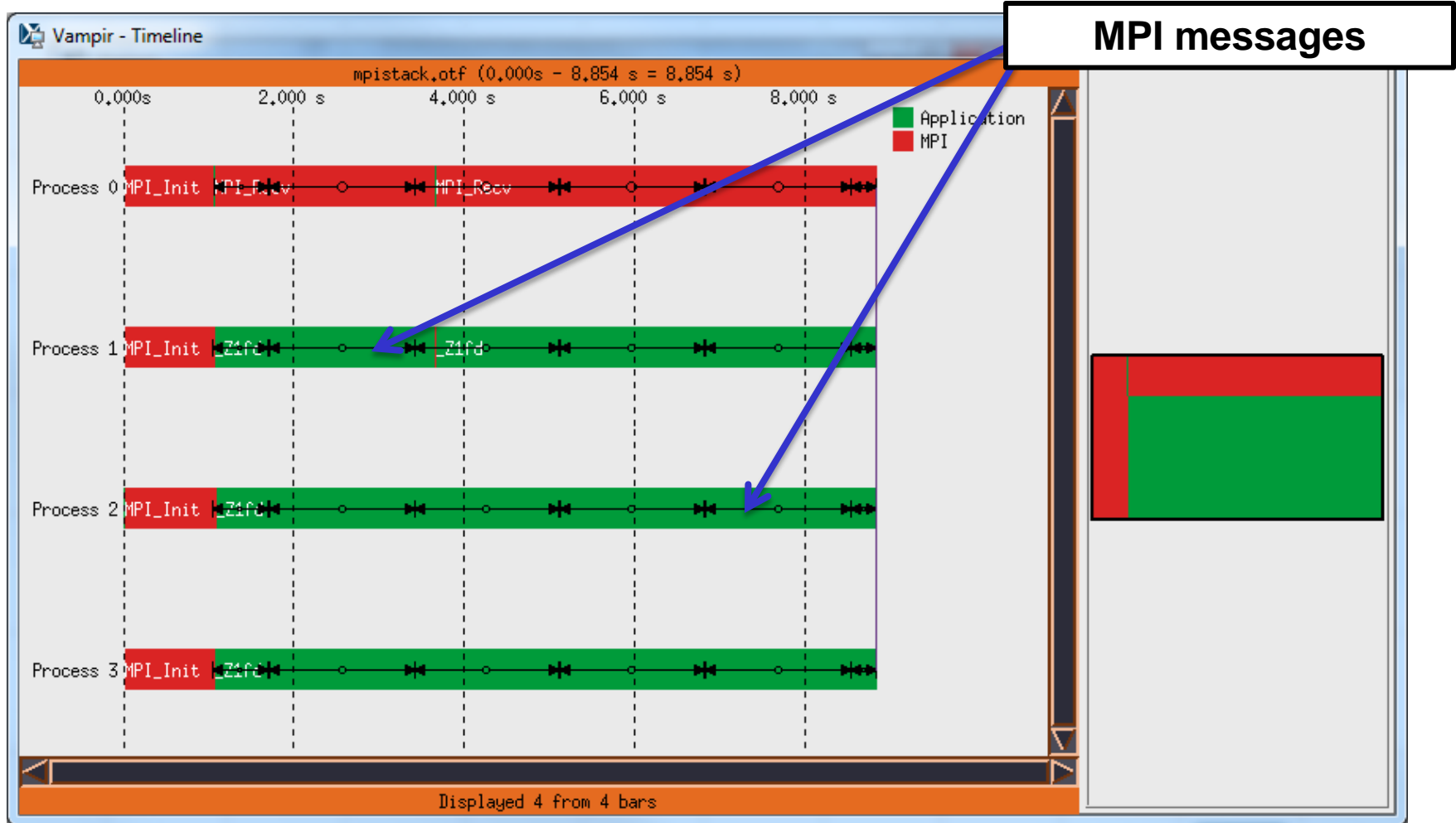


Start by looking at the timeline

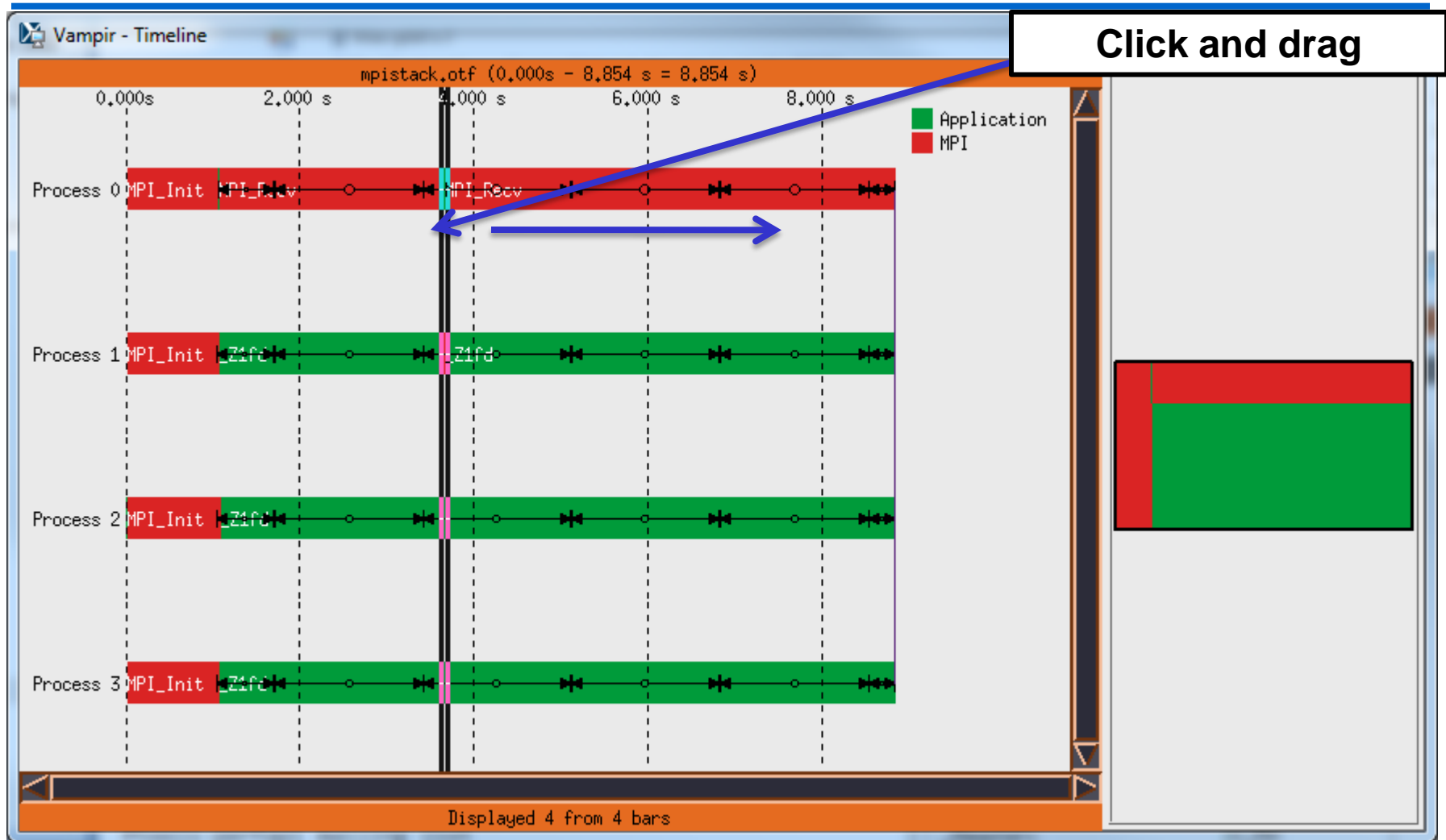


Vampir timeline

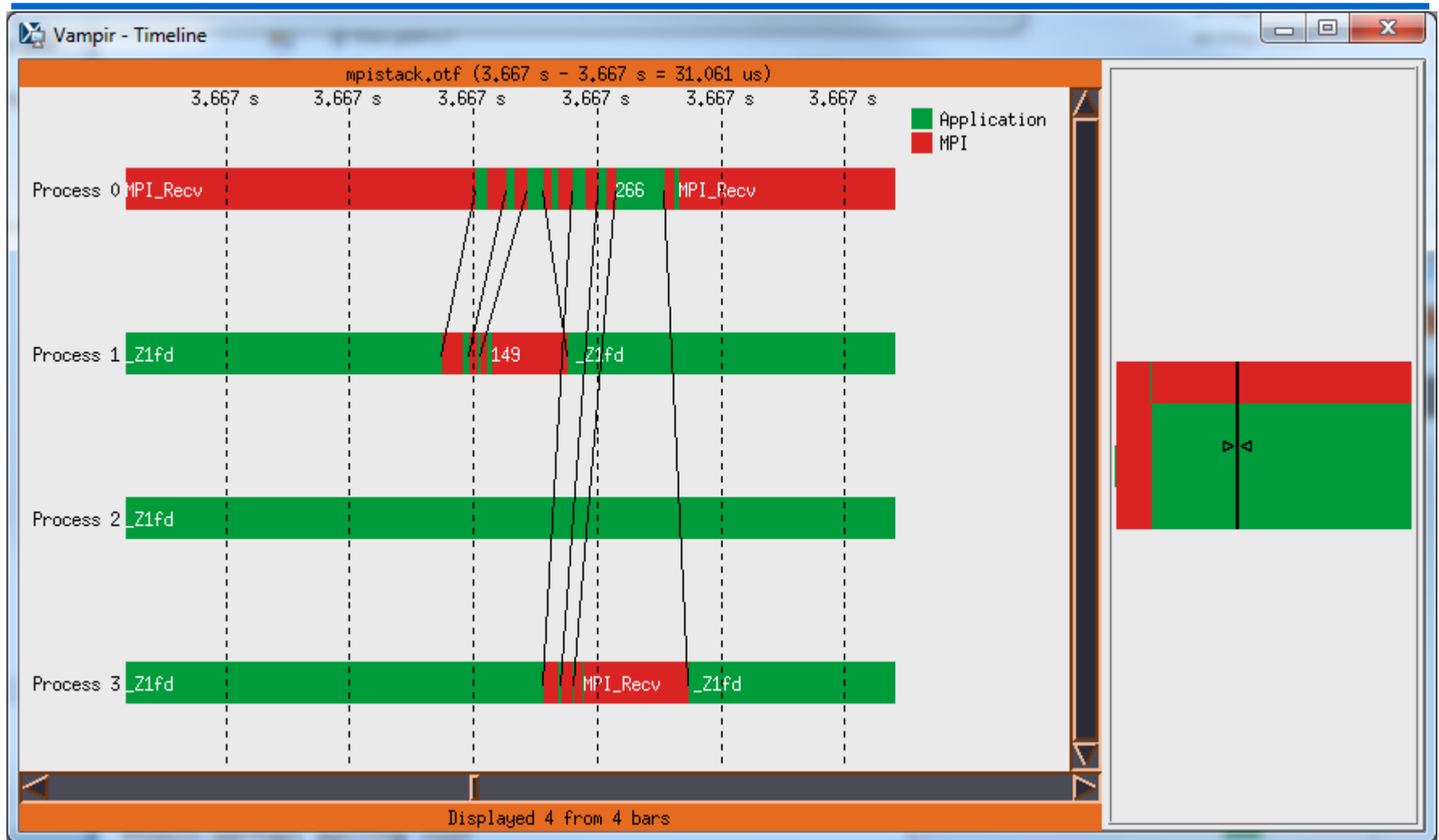




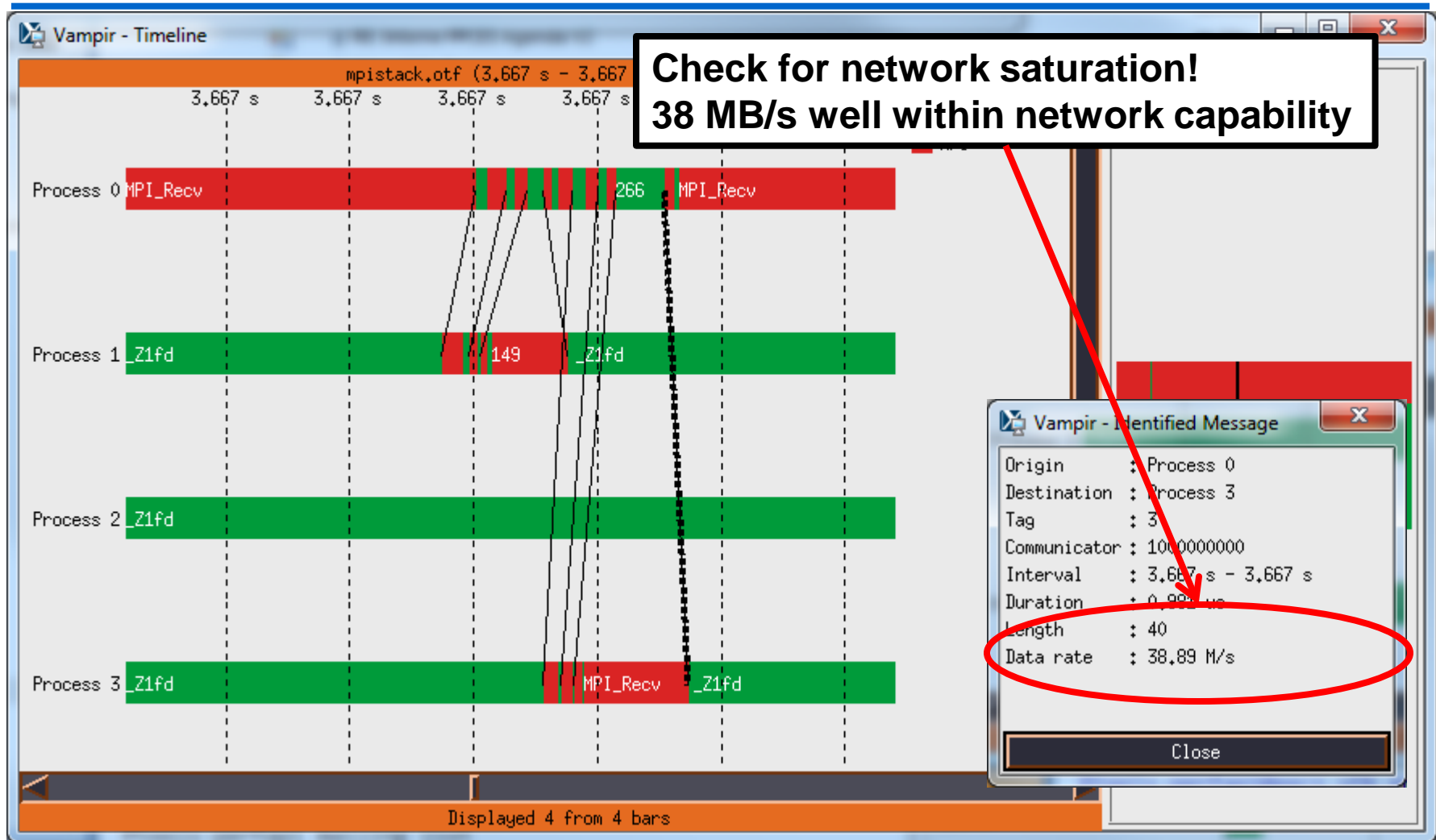
Zooming into the timeline



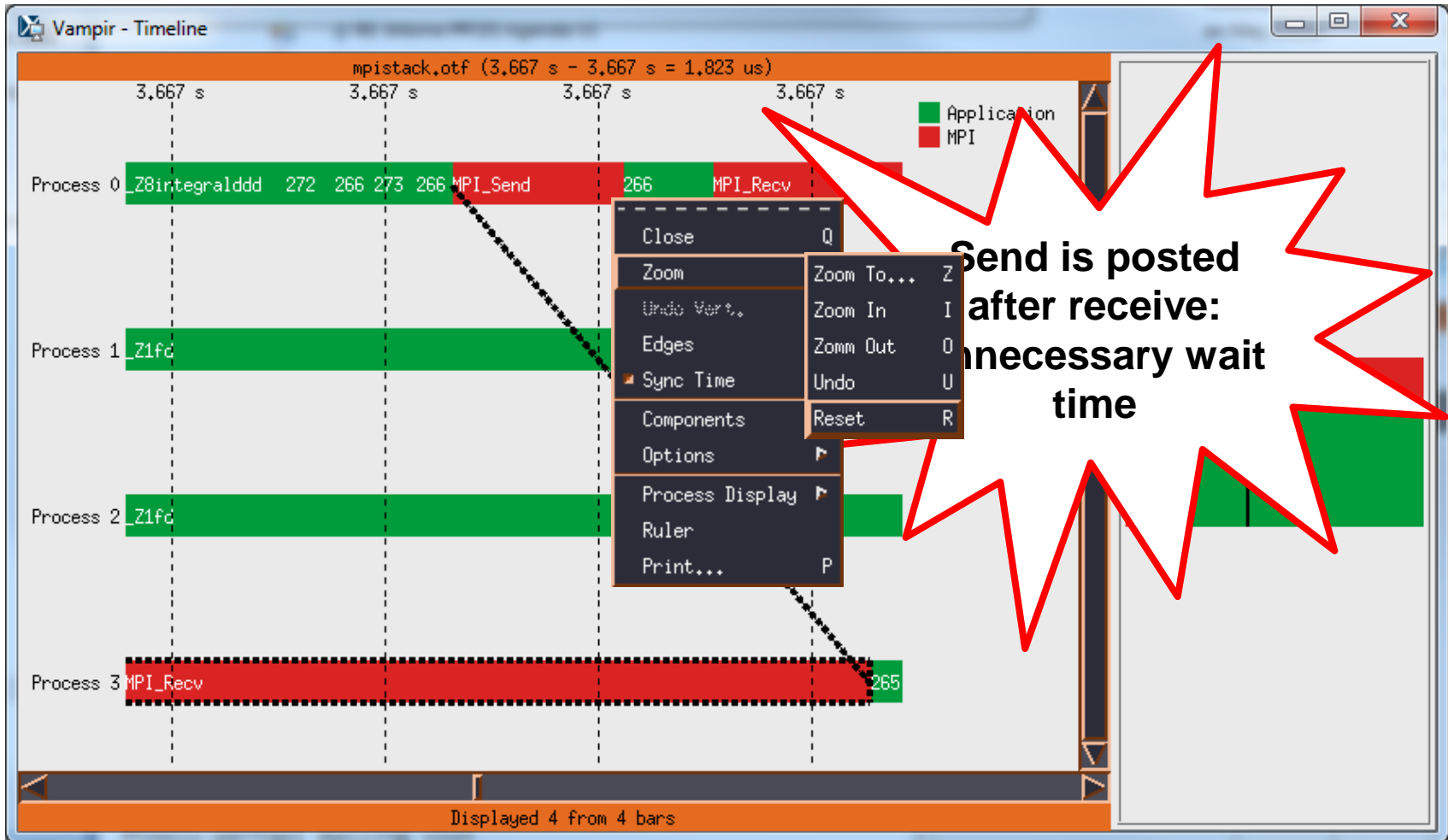
Zooming into the timeline



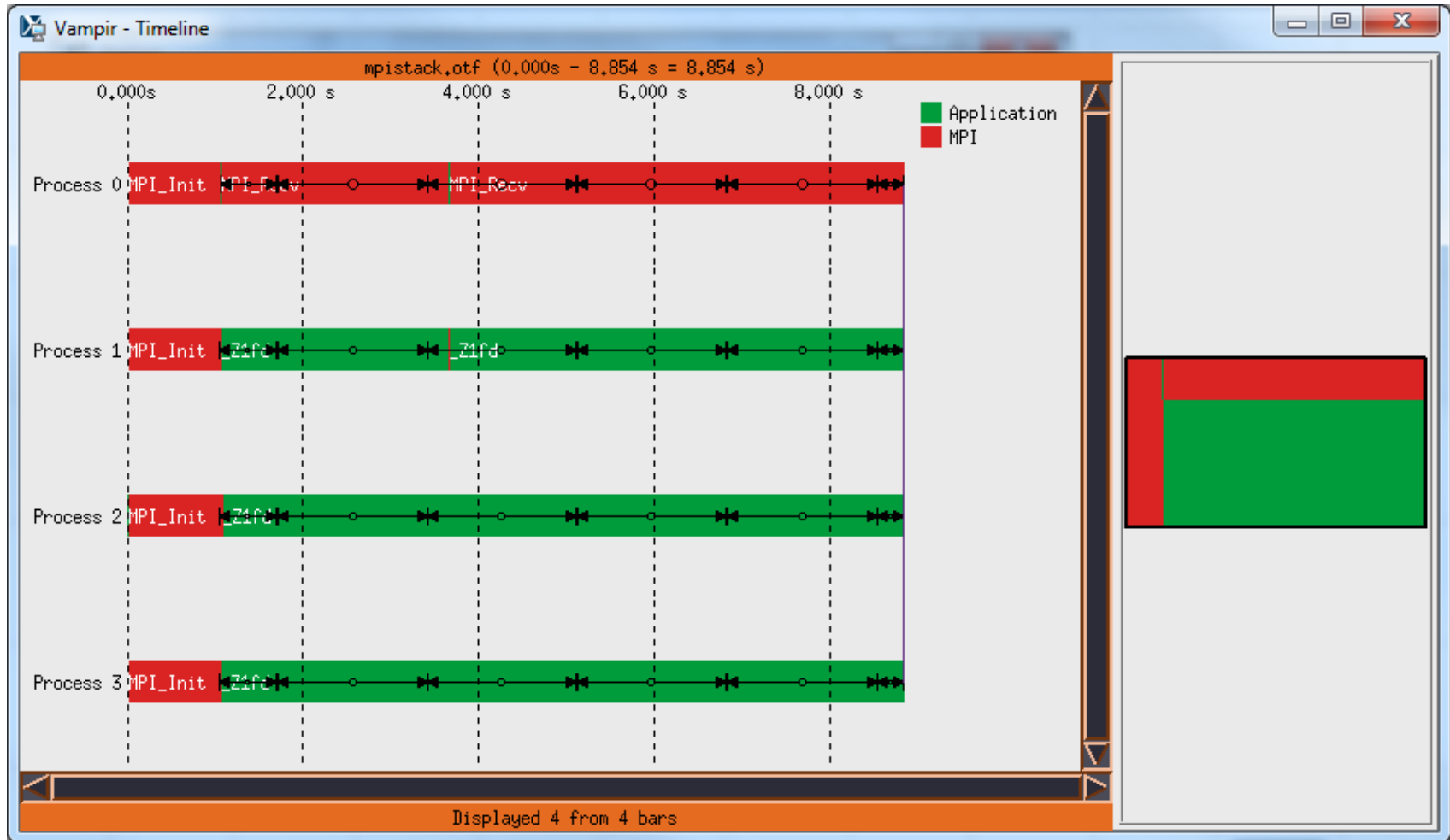
Example: Send-Receive



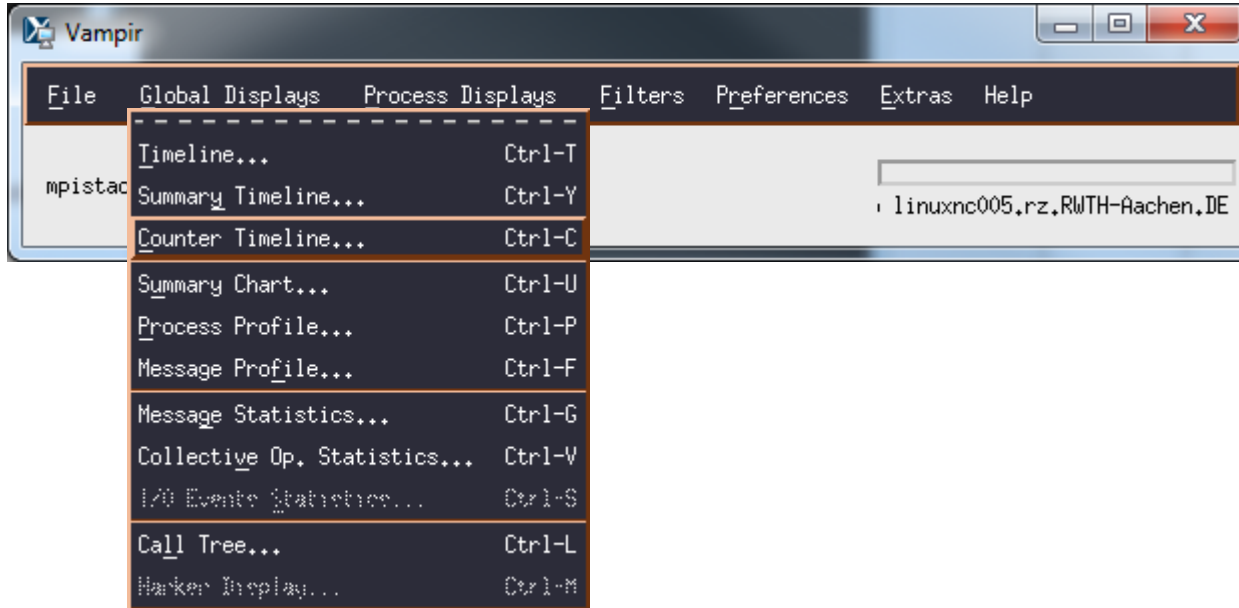
After zooming in



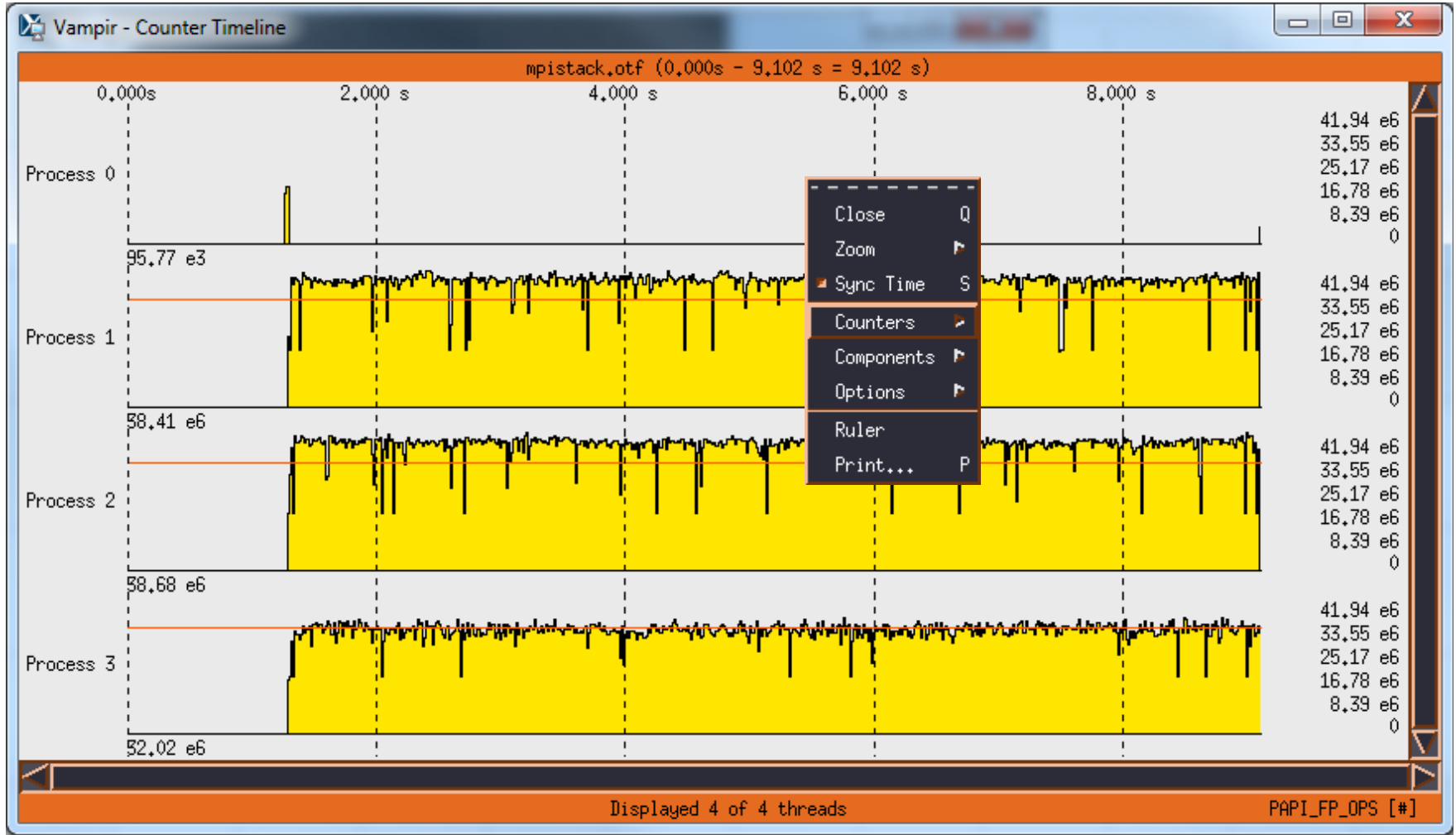
Vampir Next Generation timeline



Hardware counters



Counter timeline

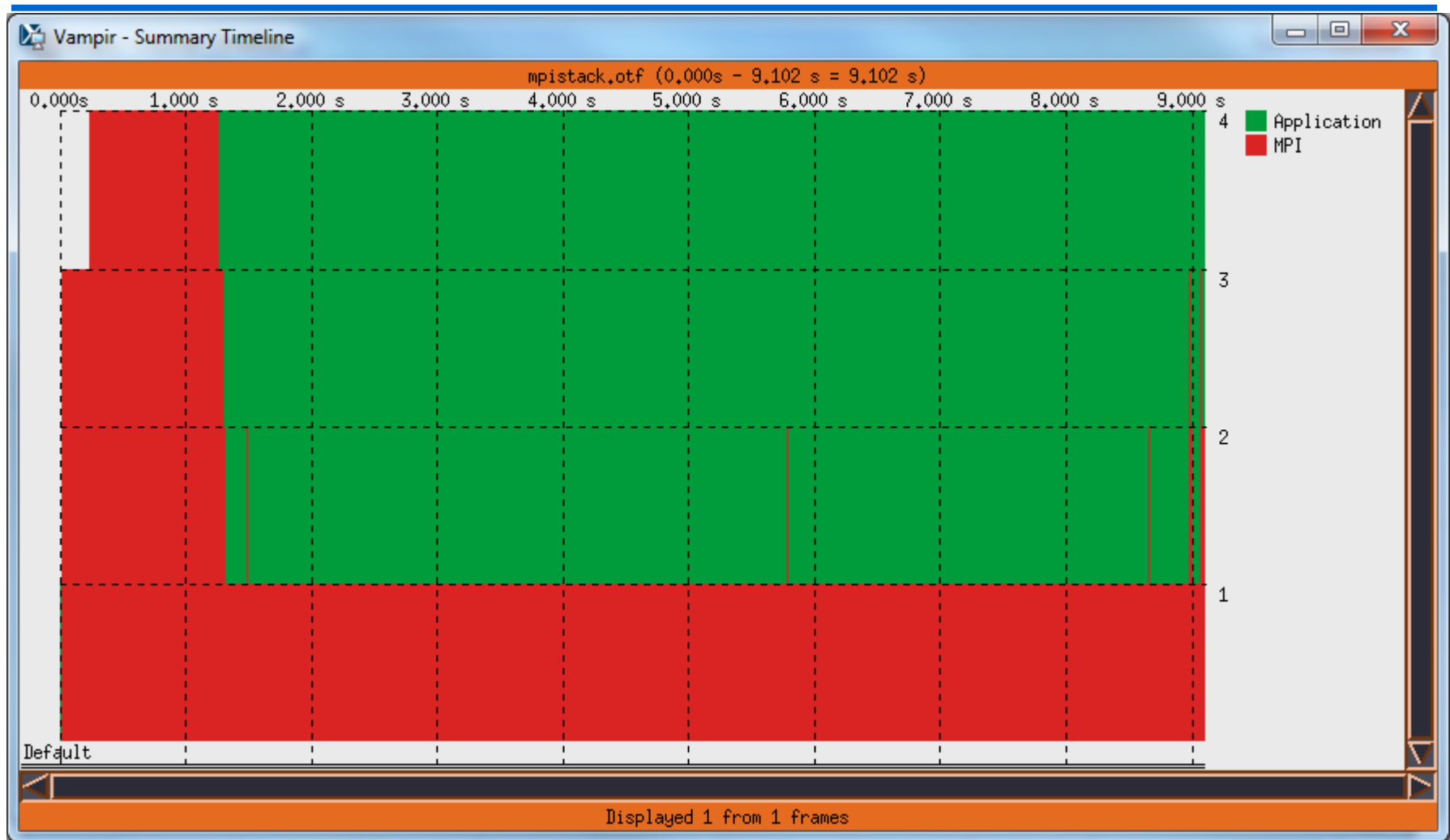


Vampir provides many different views of your application!

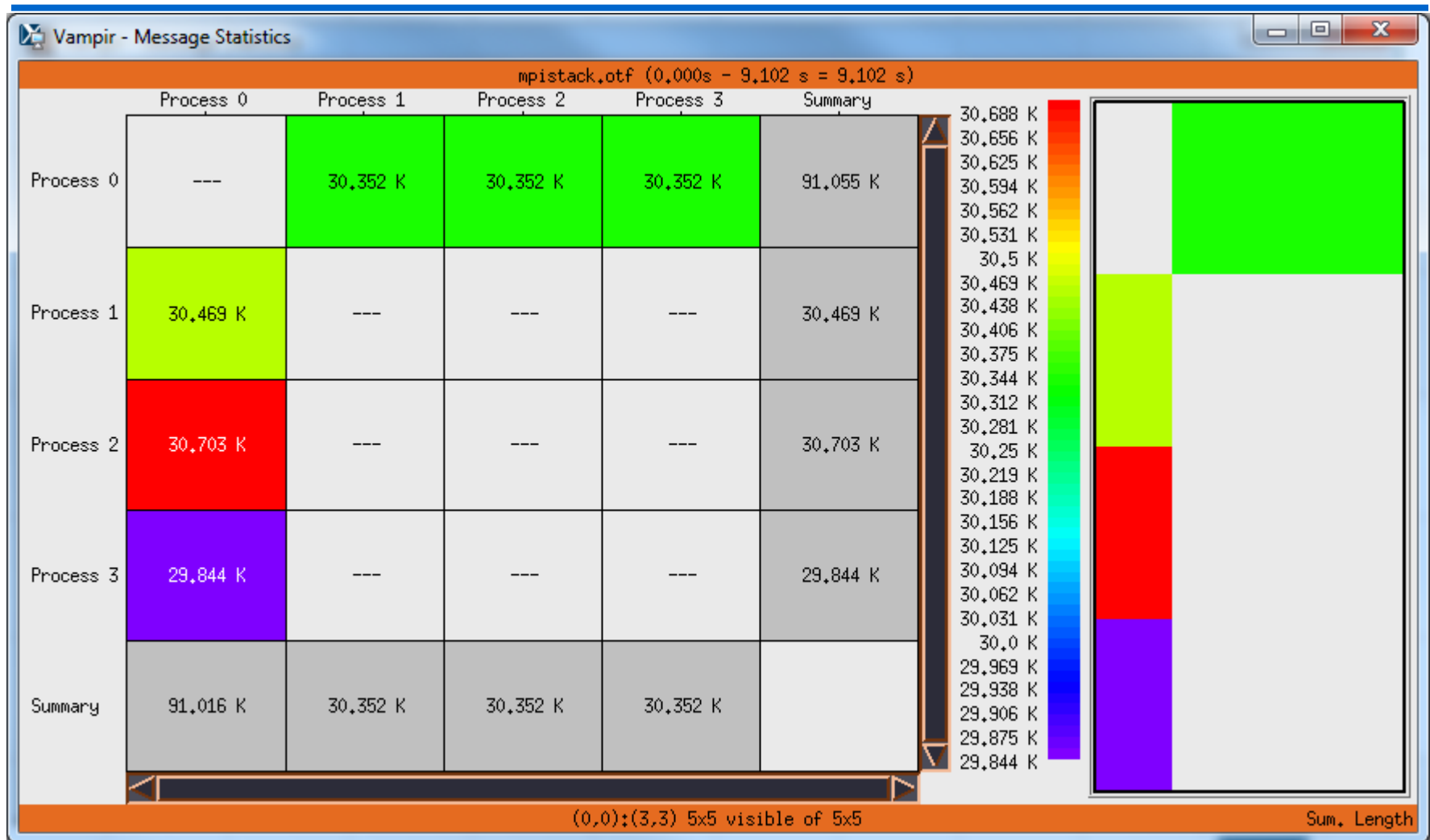
A selection:

- **Summary chart**
- **Time line**
- **Summary timeline**
- **Message profile**
- **Message statistics**
- **Call Tree**
- **Per process timeline**
- **...**

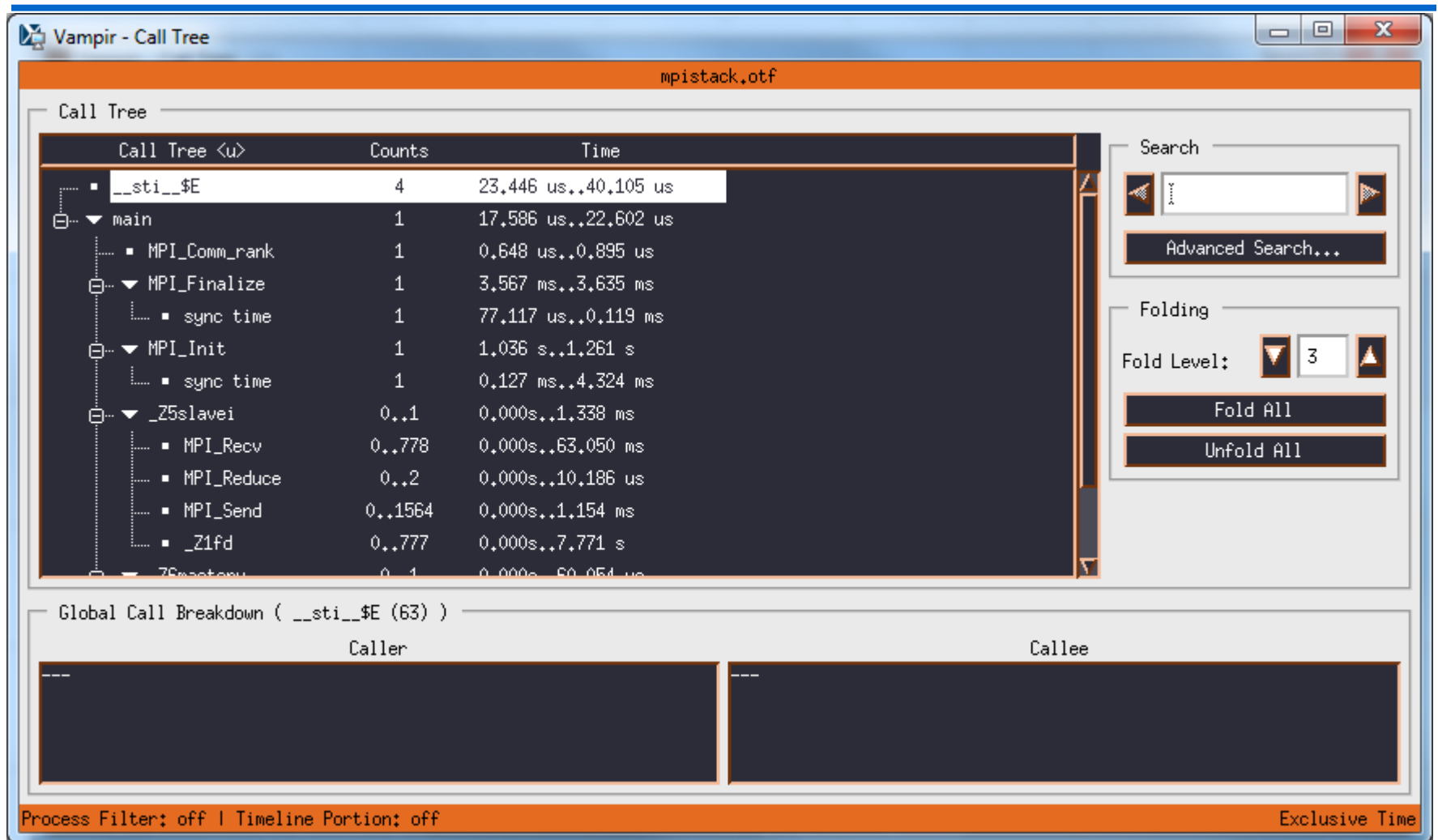
Summary timeline



Message statistics

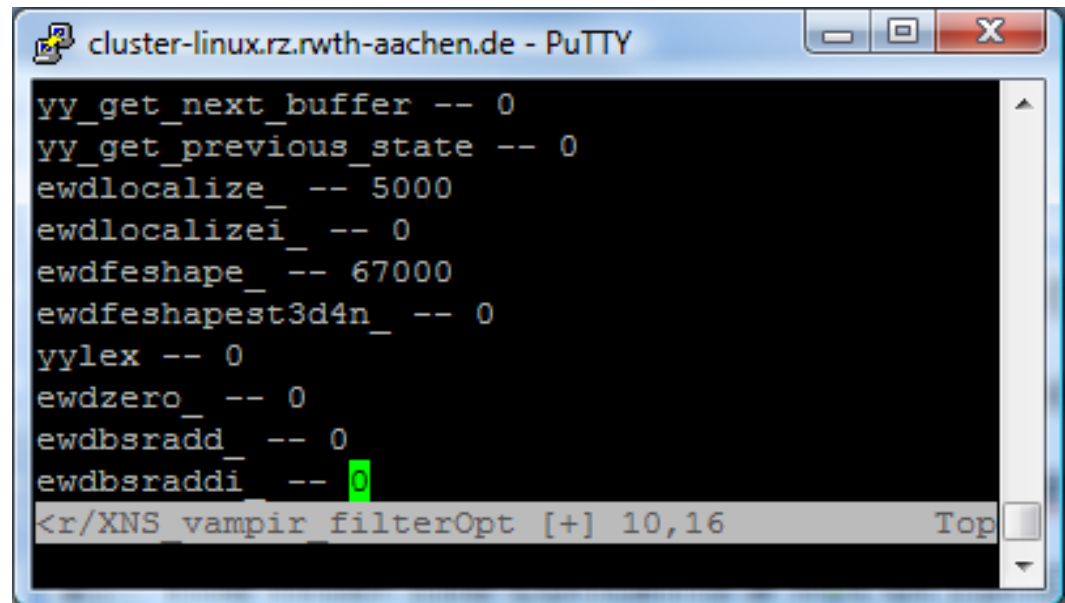


Call tree



Vampir also provides runtime filtering capabilities

1. Define a filter file in the environment variable
`export VT_FILTER_SPEC=~/.myFilter`
2. The filter file contains a list of functions names with a limit
3. Execute your application as normal



The screenshot shows a PuTTY terminal window titled "cluster-linux.rz.rwth-aachen.de - PuTTY". The terminal displays a list of Vampir filter options, each with a function name followed by "--" and a value. The values are: 0, 0, 5000, 0, 67000, 0, 0, 0, 0, and 0. The last line is highlighted in green. The prompt is "<r/XNS vampir filterOpt [+] 10,16".

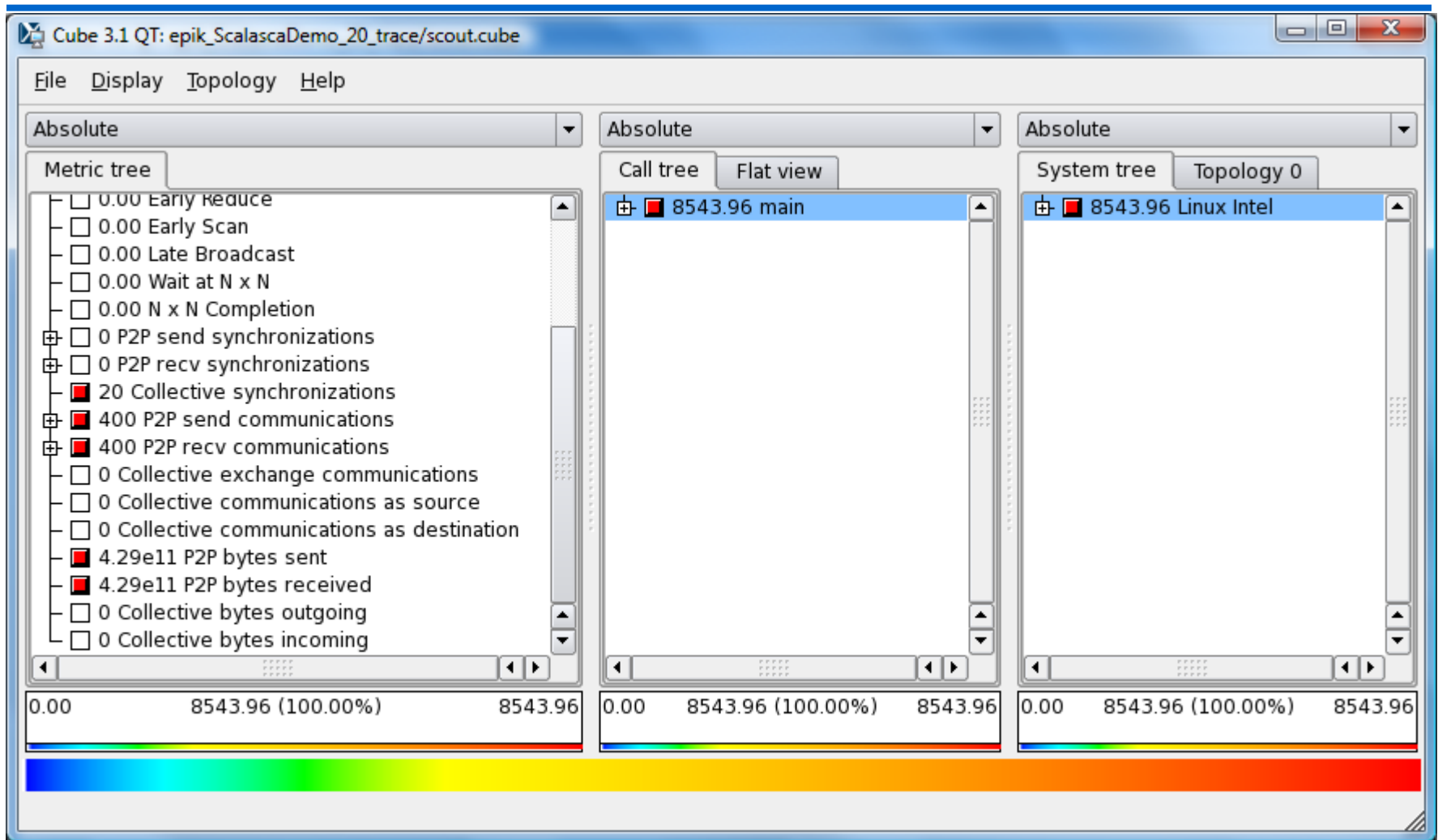
```
yy_get_next_buffer -- 0
yy_get_previous_state -- 0
ewdlocalize_ -- 5000
ewdlocalizei_ -- 0
ewdfeshape_ -- 67000
ewdfeshapest3d4n_ -- 0
yylex -- 0
ewdzero_ -- 0
ewdbsradd_ -- 0
ewdbsraddi_ -- 0
<r/XNS vampir filterOpt [+] 10,16
```

Brief gimps of Scalasca

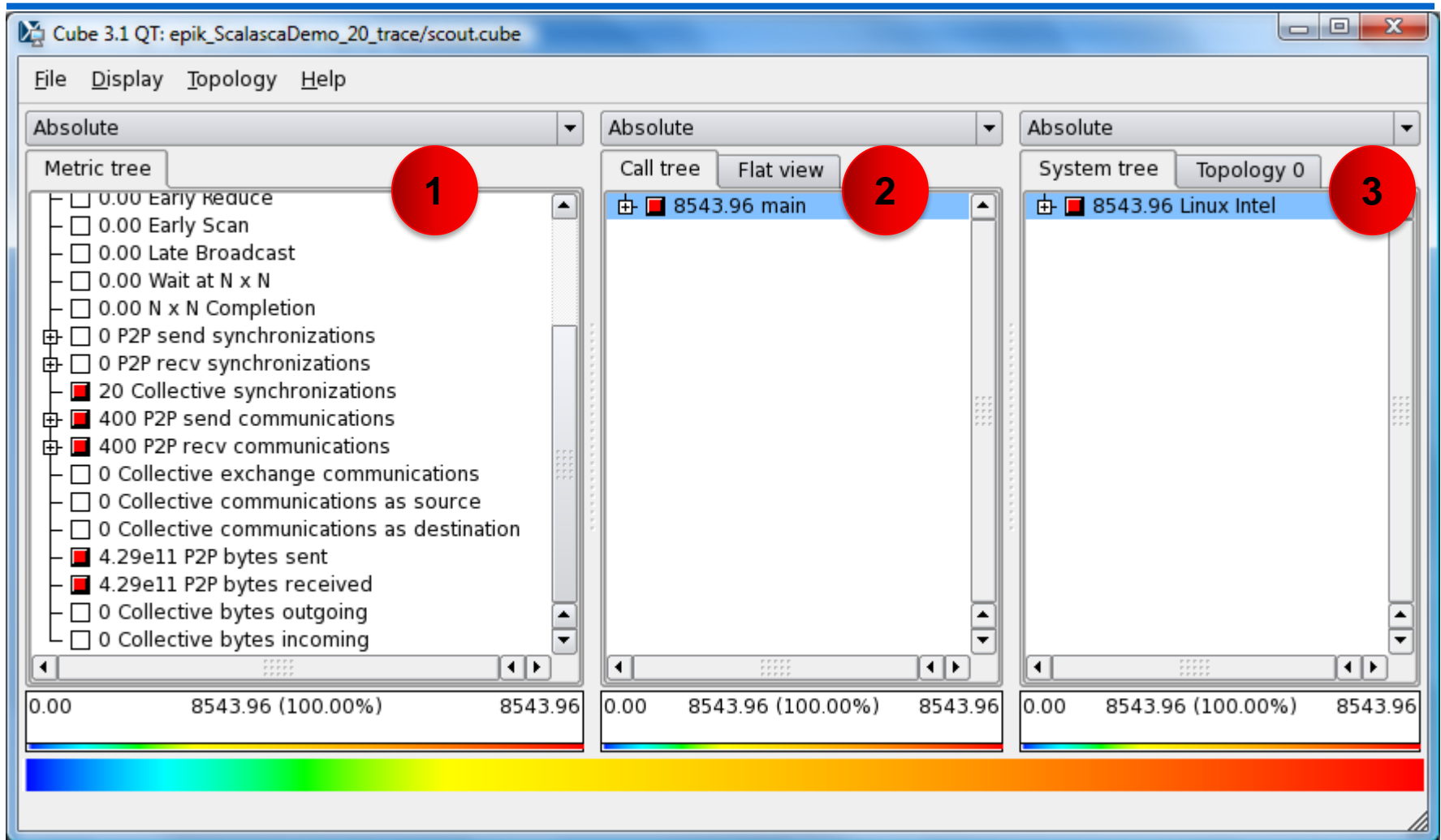
- Developed at the Jülich Supercomputing Center
- Performance toolset
 - scalasca: measurement driver
 - scout: trace analysis tool
 - scan: runtime measurement system
 - cube: trace and analysis inspection tool
- Designed to analyze large parallel applications
- Supports OpenMP and MPI
- **Automated analysis functionality to spot typical performance problems in large parallel runs**
- Performs combined measurement and analysis



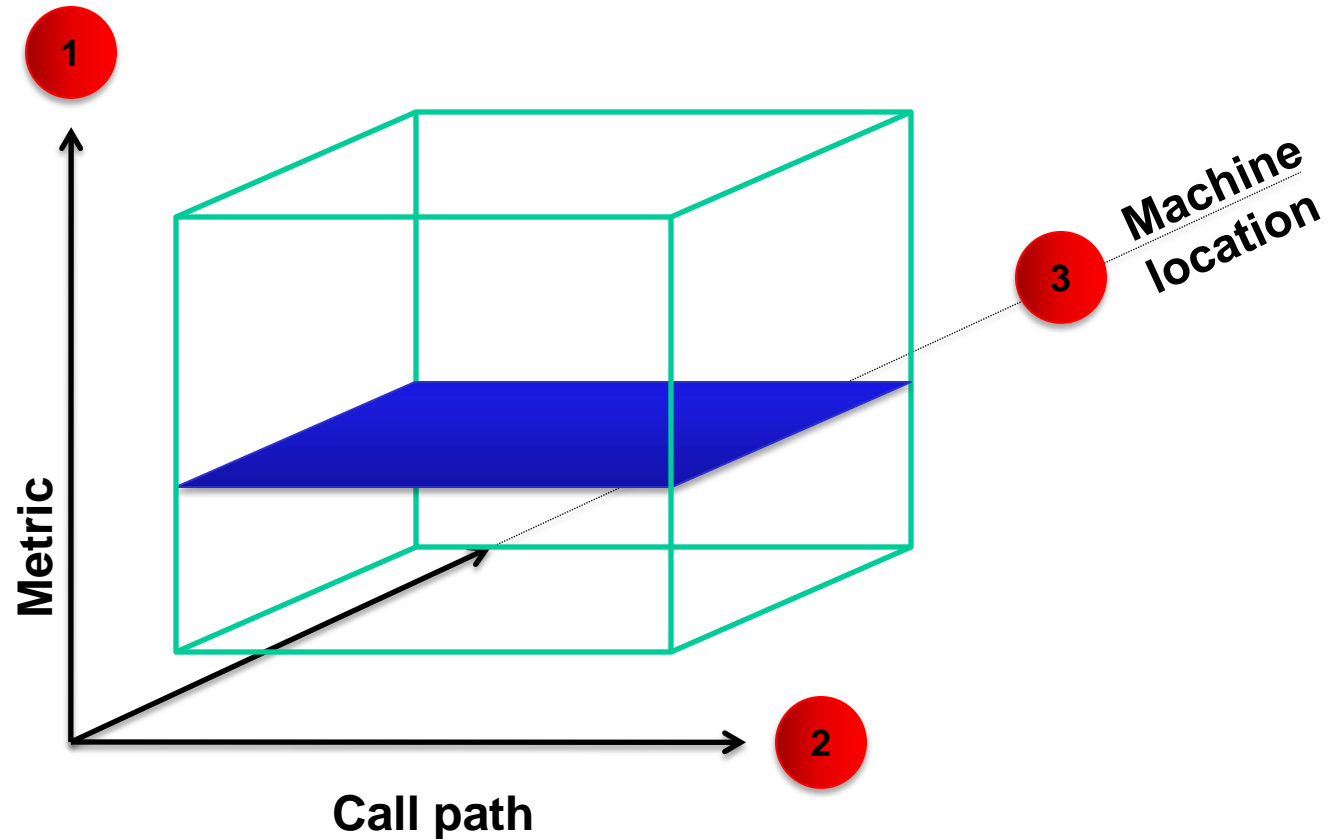
The main cube3 window



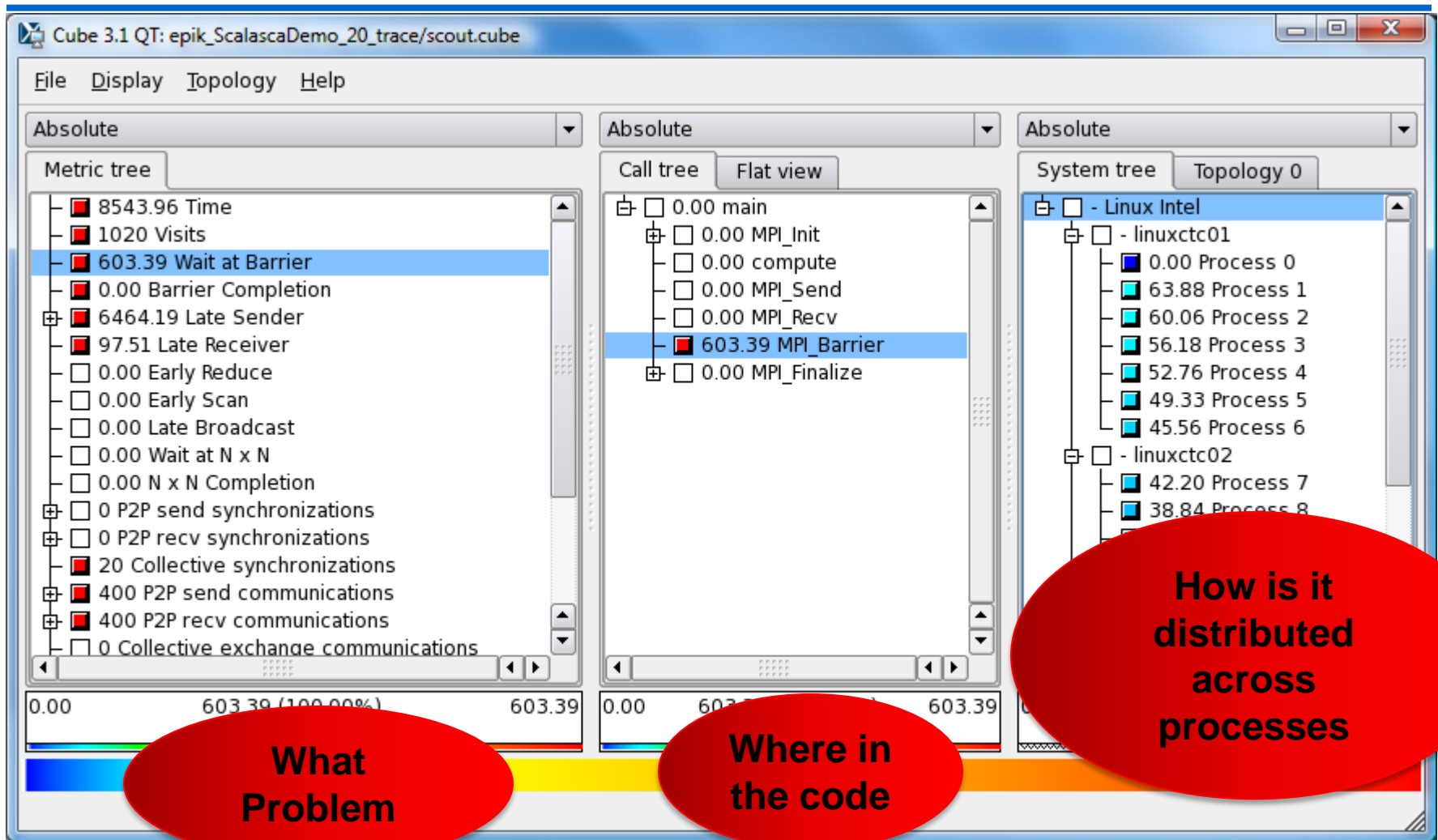
The main cube3 window



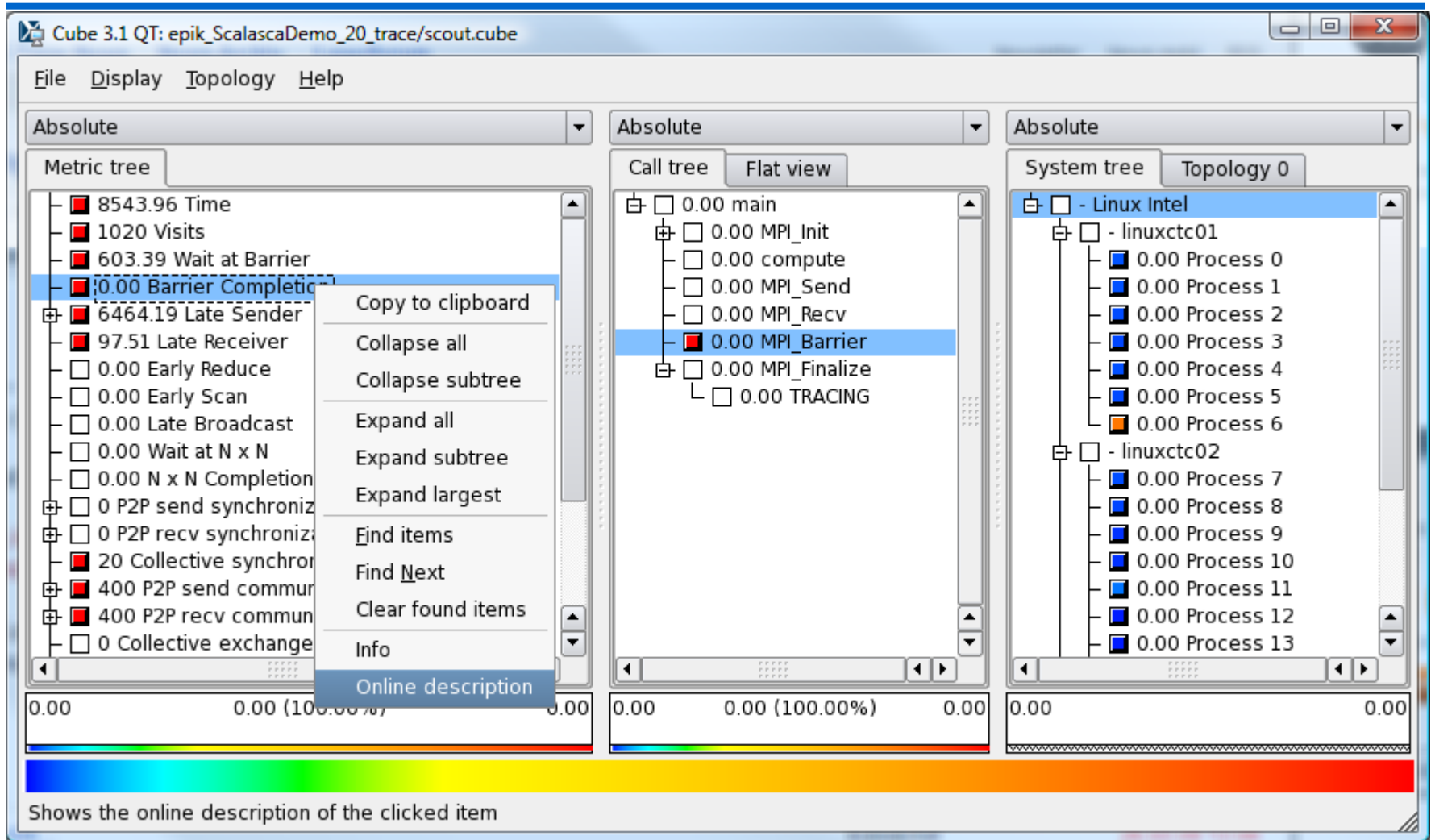
3D Data space



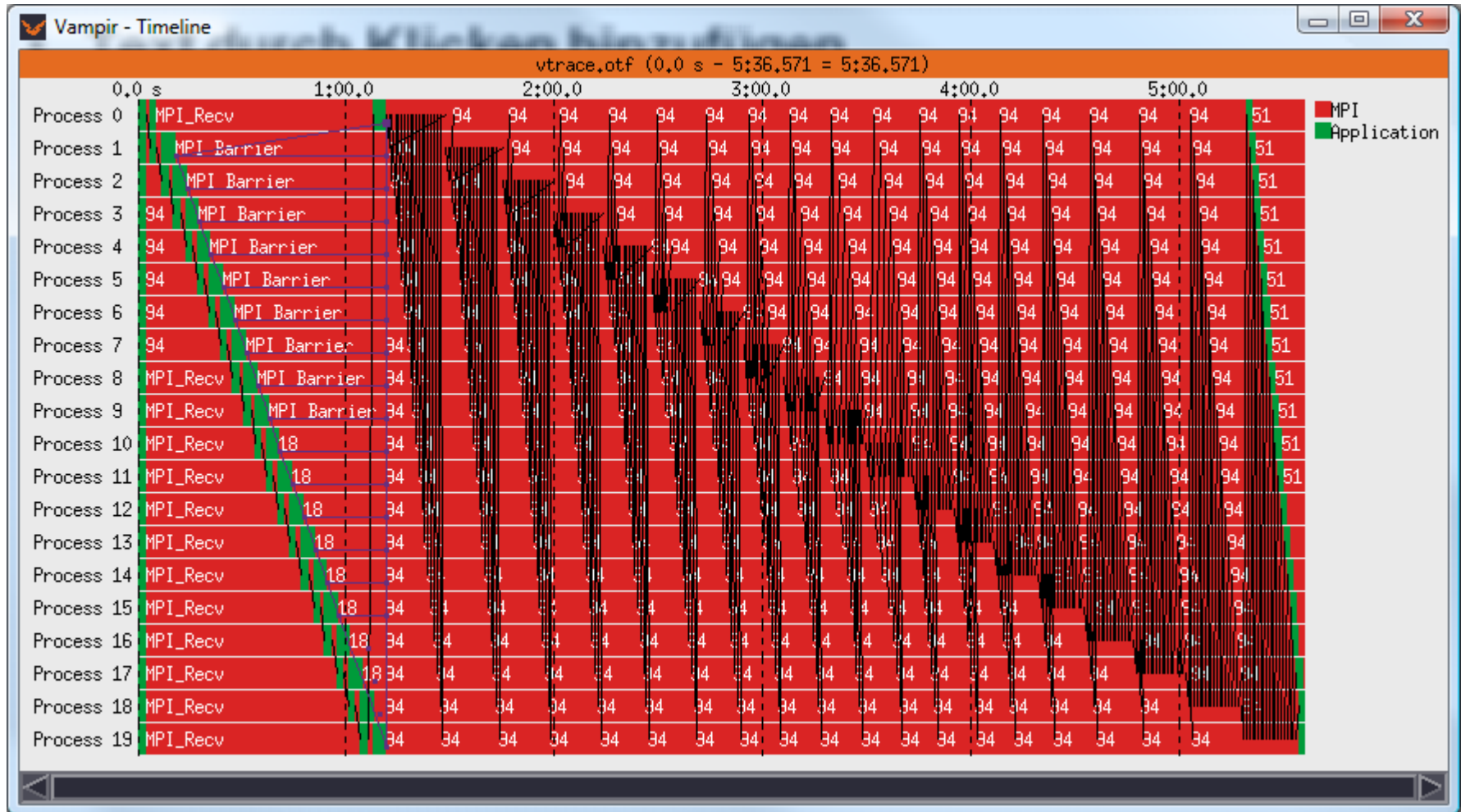
Breakup by topology



For every metric: online description



Why Scalasca? Vampir gets too crowded for large MPI jobs!



More information

More information about scalasca can be found on this seminars webpage!

<http://www.rz.rwth-aachen.de/ppces>

Go to course material!!

Oracle Performance Analyzer

Setup: `module load studio`

Important commands:

`collect`

`analyzer`

For function and usage details, refer to this mornings sessions!

Further Documentation:

<http://developers.sun.com/sunstudio/overview/topics/analyzing.jsp>

Where to get more information

HPC User Documentation (regularly updated):

<http://www.rz.rwth-aachen.de/hpc/primer>

User support:

Mail:

servicedesk@rz.rwth-aachen.de

Phone:

0241-80-24680

Questions