VAMPIR & VAMPIRTRACE INTRODUCTION AND OVERVIEW

Jens Doleschal, Andreas Knüpfer
ZIH, TU Dresden
jens.doleschal@tu-dresden.de

September 2009

Overview

- Introduction
- Event Trace Visualization
- Vampir & VampirServer
- The Vampir Displays
    - Timeline
    - Process Timeline with Performance Counters
    - Summary Display
    - Message Statistics
- VampirTrace
    - Instrumentation & Run-Time Measurement
- Conclusions

Why bother with performance analysis?

- well, why are you here after all?
- efficient usage of expensive and limited resources
- scalability to achieve next bigger simulation

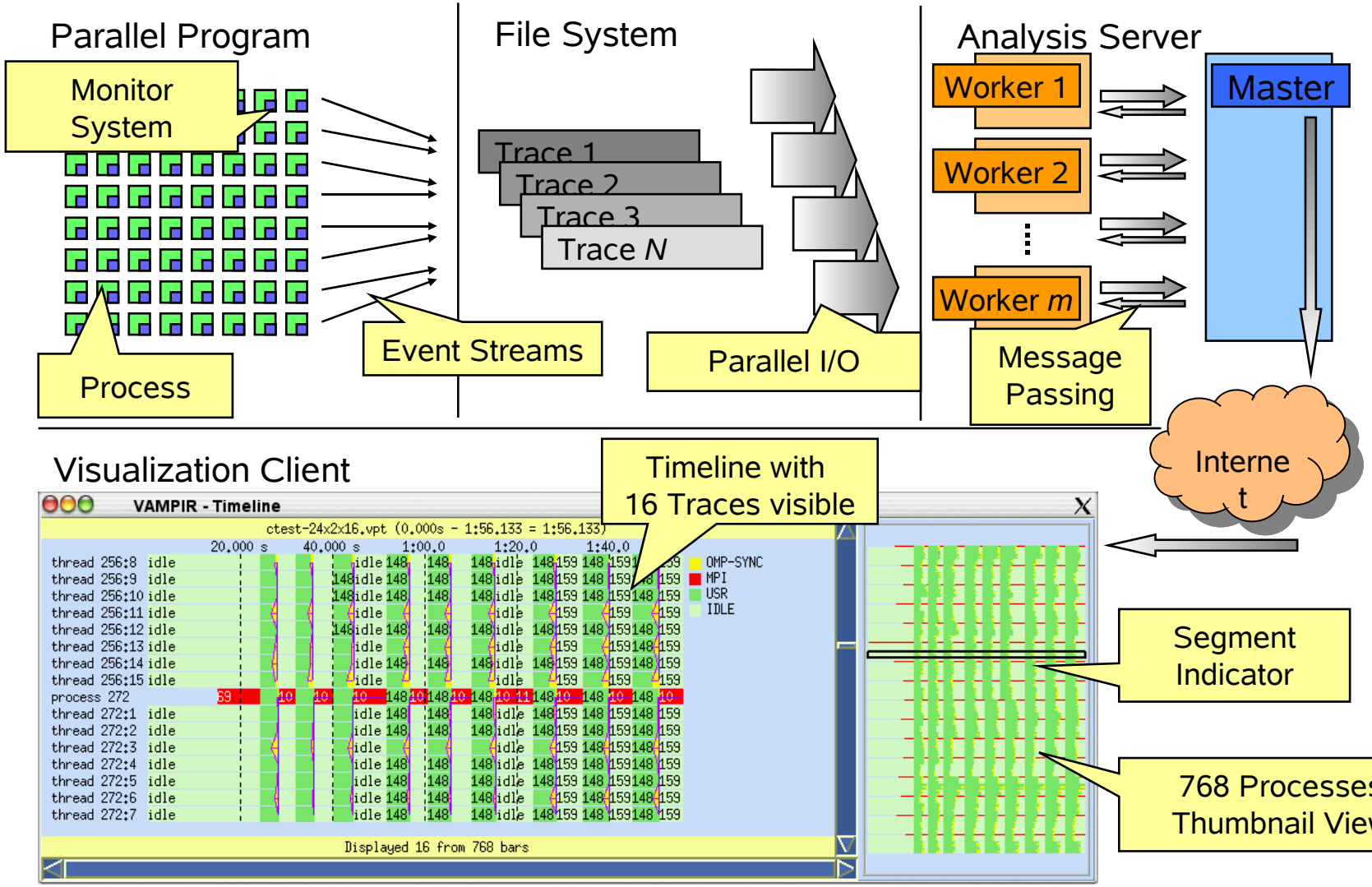
Profiling and Tracing

- have an optimization phase
  - just like testing and debugging phase
- use tools!
- avoid *do-it-yourself-with-printf* solutions, really!

Trace Visualization

- alternative and supplement to automatic analysis
- show dynamic run-time behavior graphically
- provide statistics and performance metrics
  - Global timeline for parallel processes/threads
  - Process timeline plus performance counters
  - Statistics summary display
  - Message statistics
  - more
- interactive browsing, zooming, selecting
  - adapt statistics to zoom level (time interval)
  - also for very large and highly parallel traces

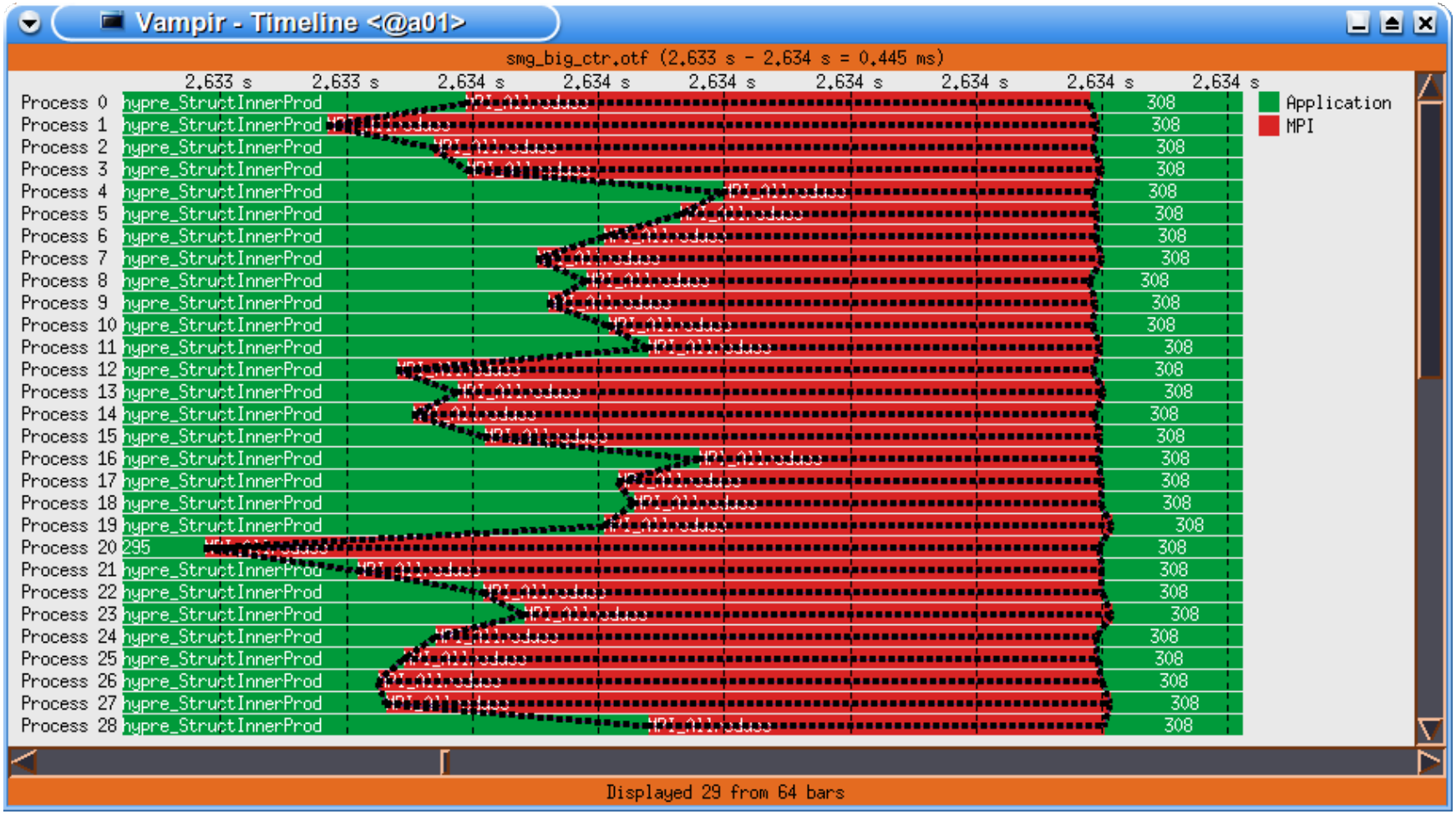
VampirServer Architecture



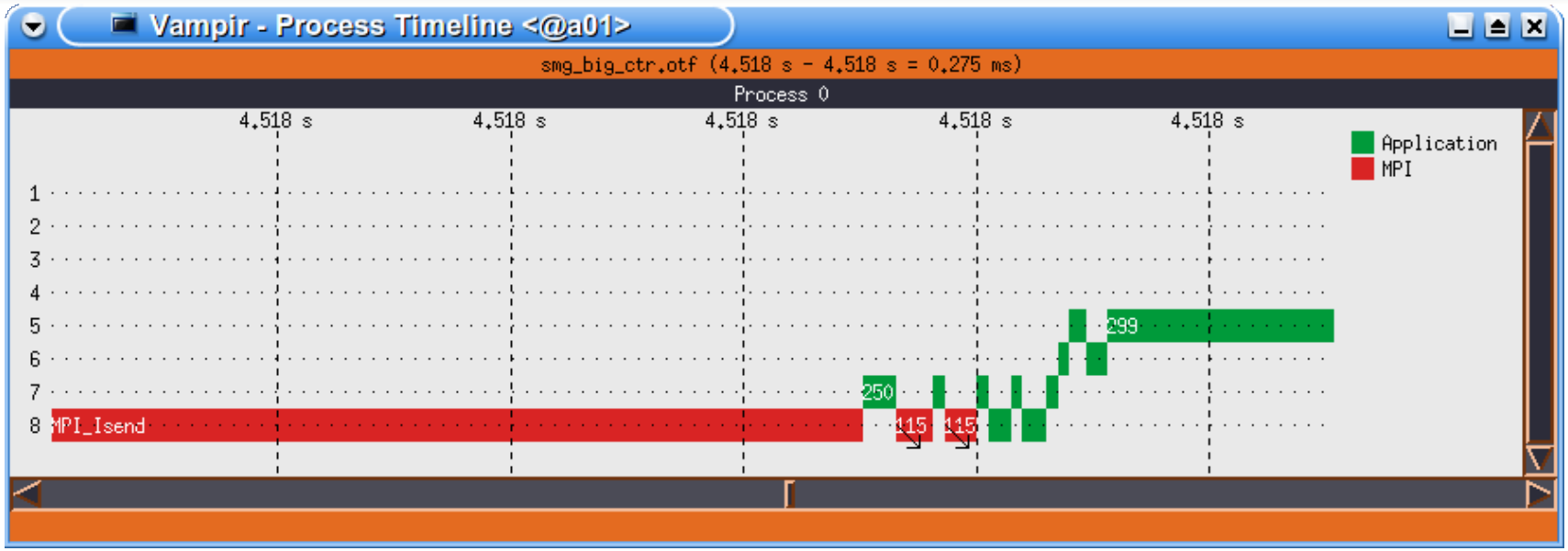
The main displays of Vampir:

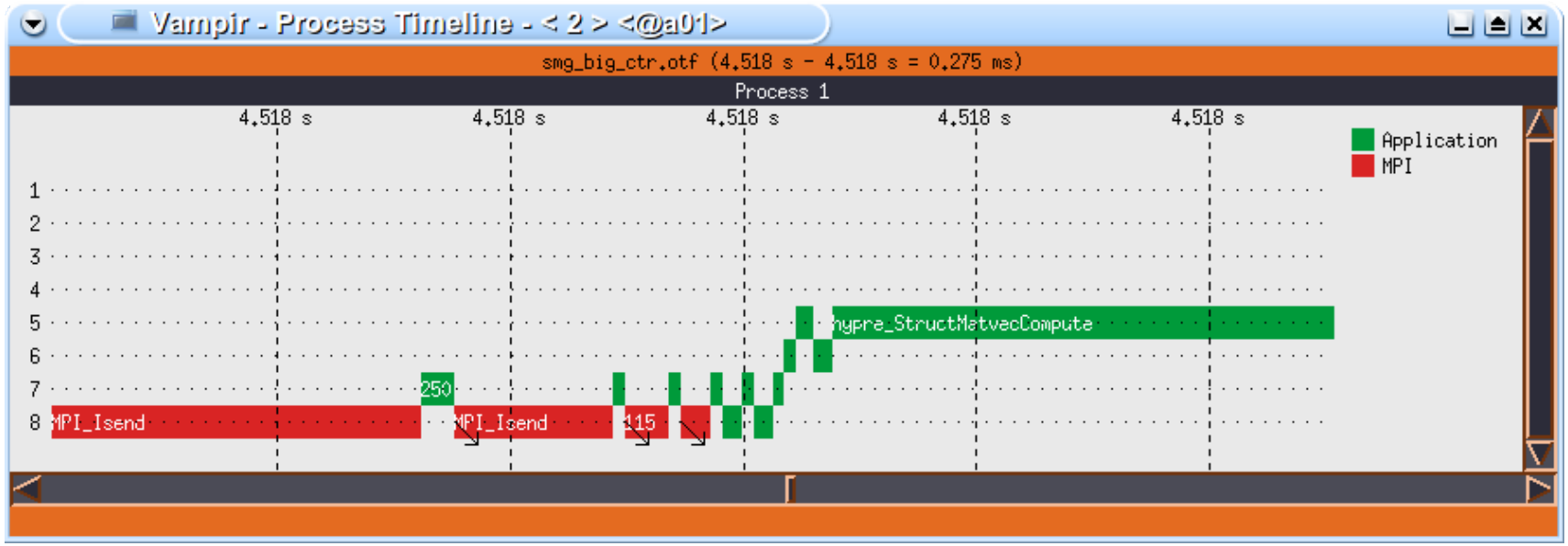
- Global Timeline
- Process Timeline w/o Counters
- Statistic Summary
- Summary Timeline
- Message Statistics
- Collective Operation Statistics
- Counter Timeline
- Call Tree

Vampir Global Timeline Display

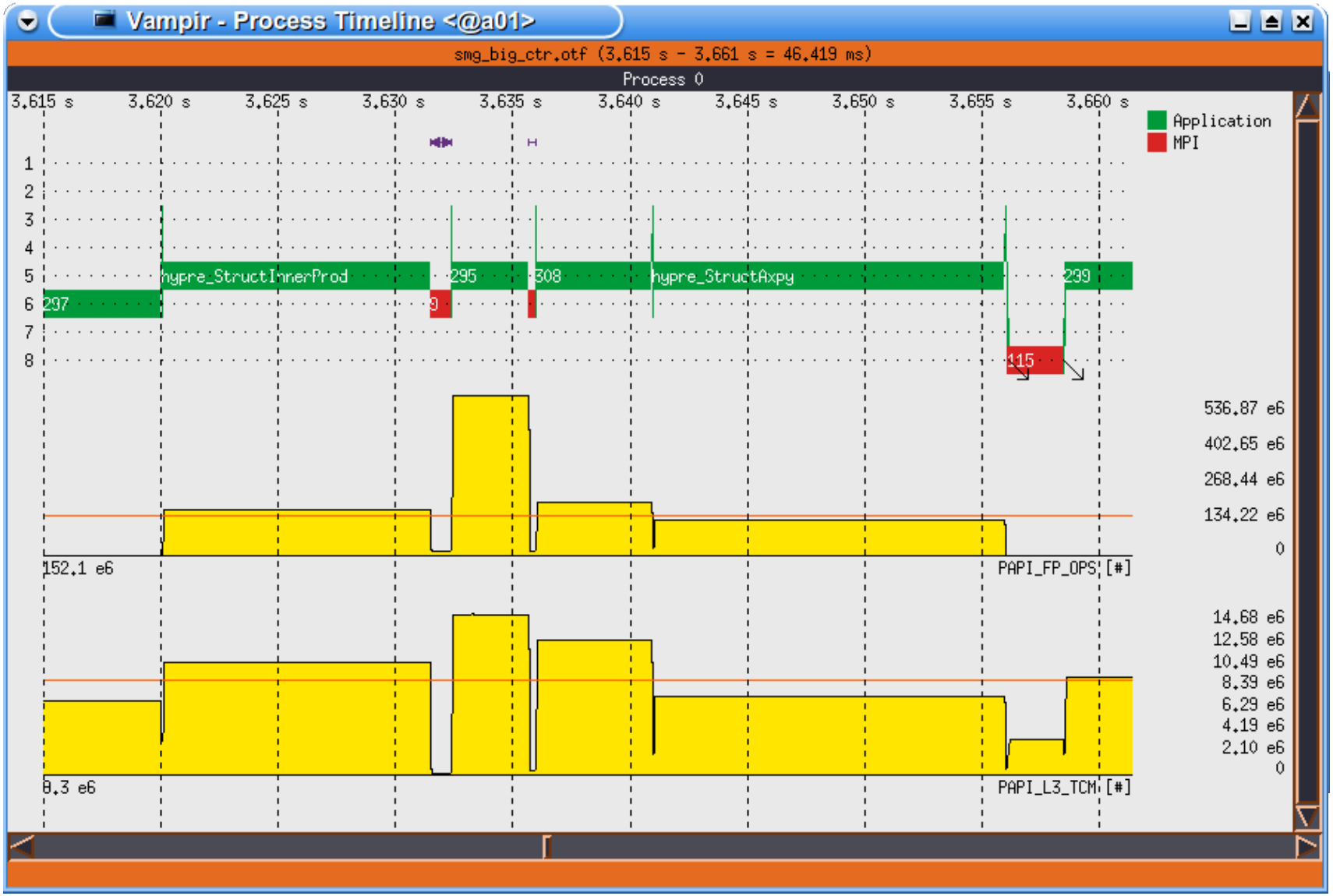


Process Timeline Display

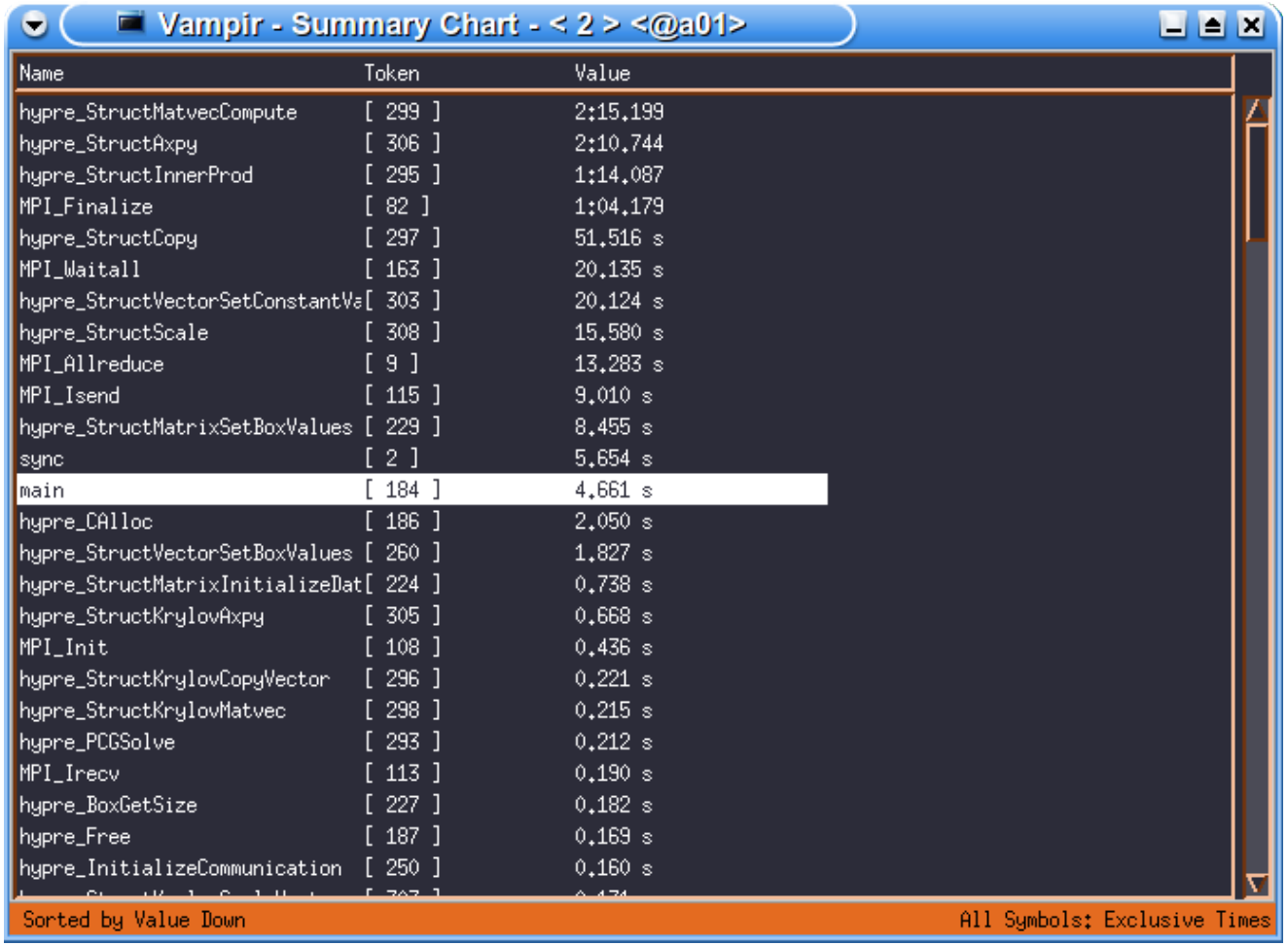




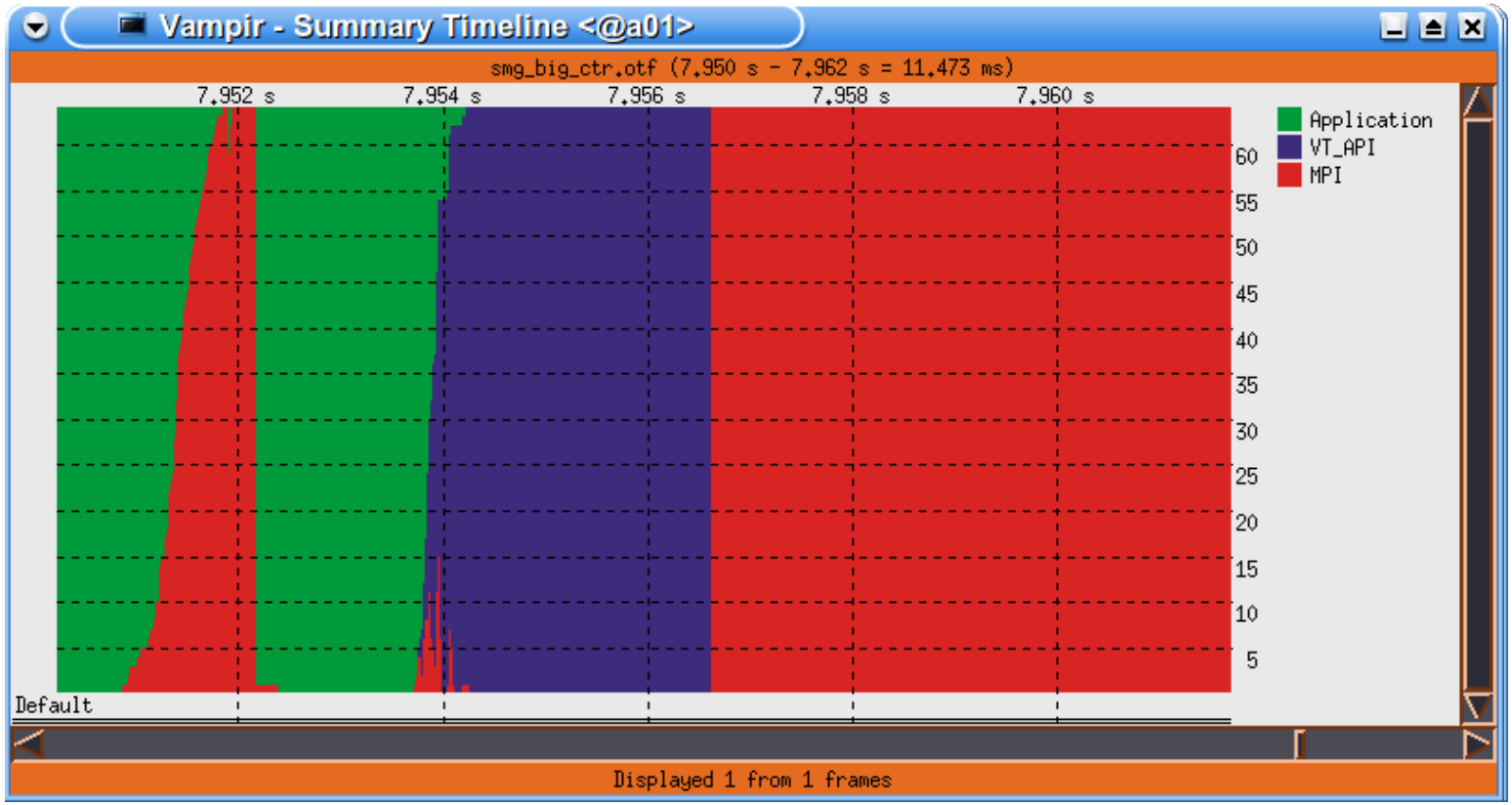
Process Timeline with Counters



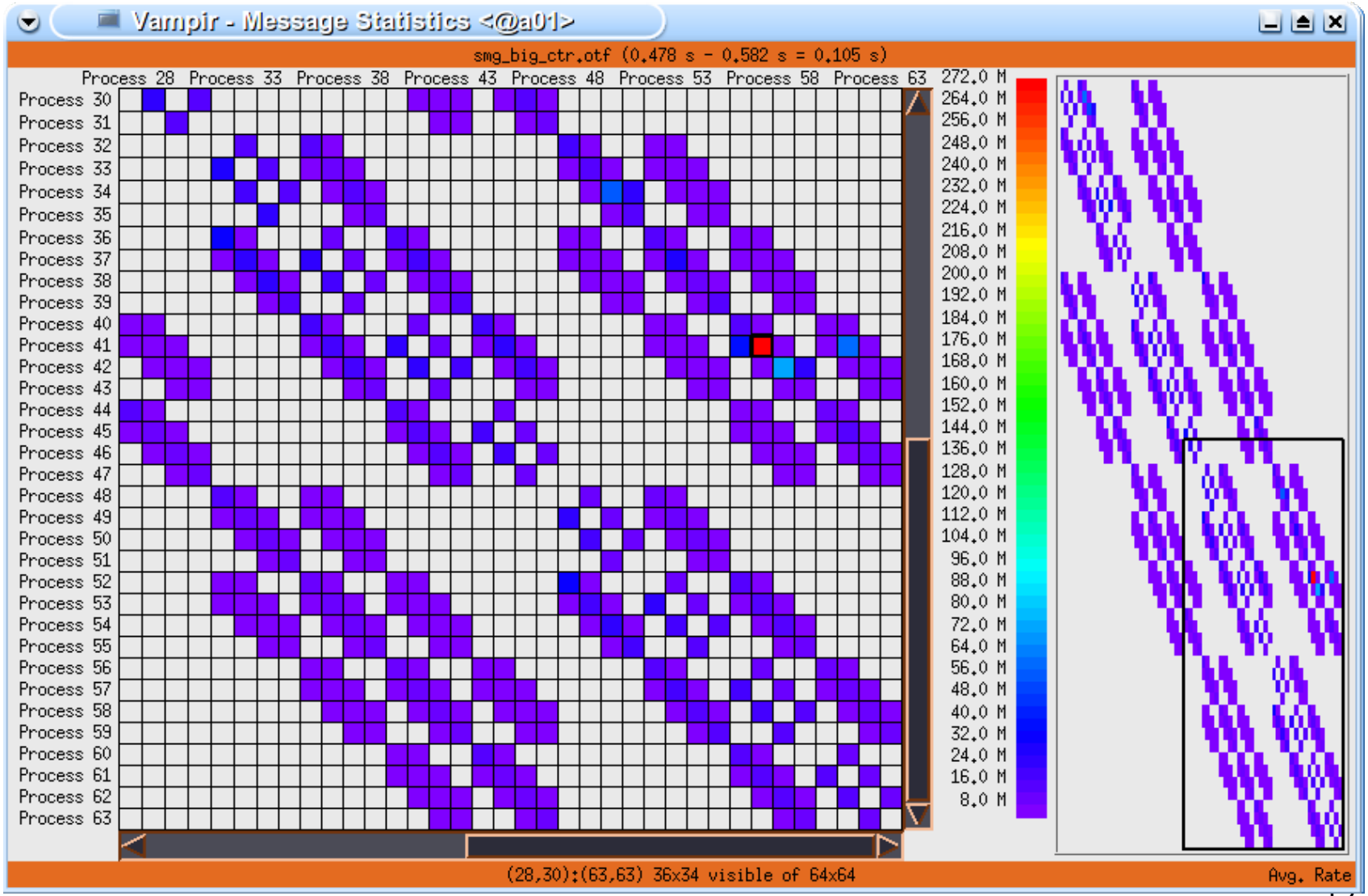
Statistics Summary Display



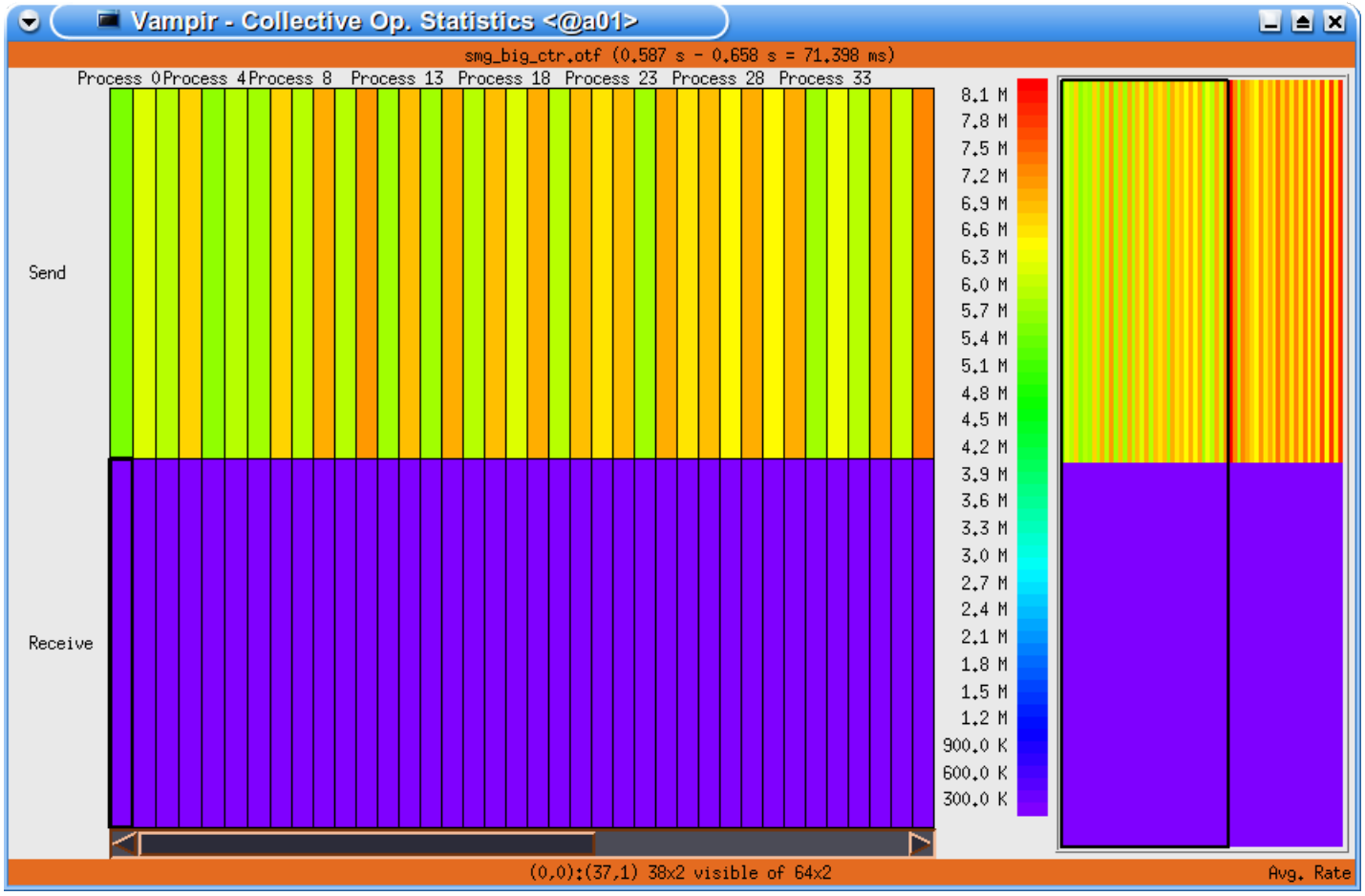
Summary Timeline Display



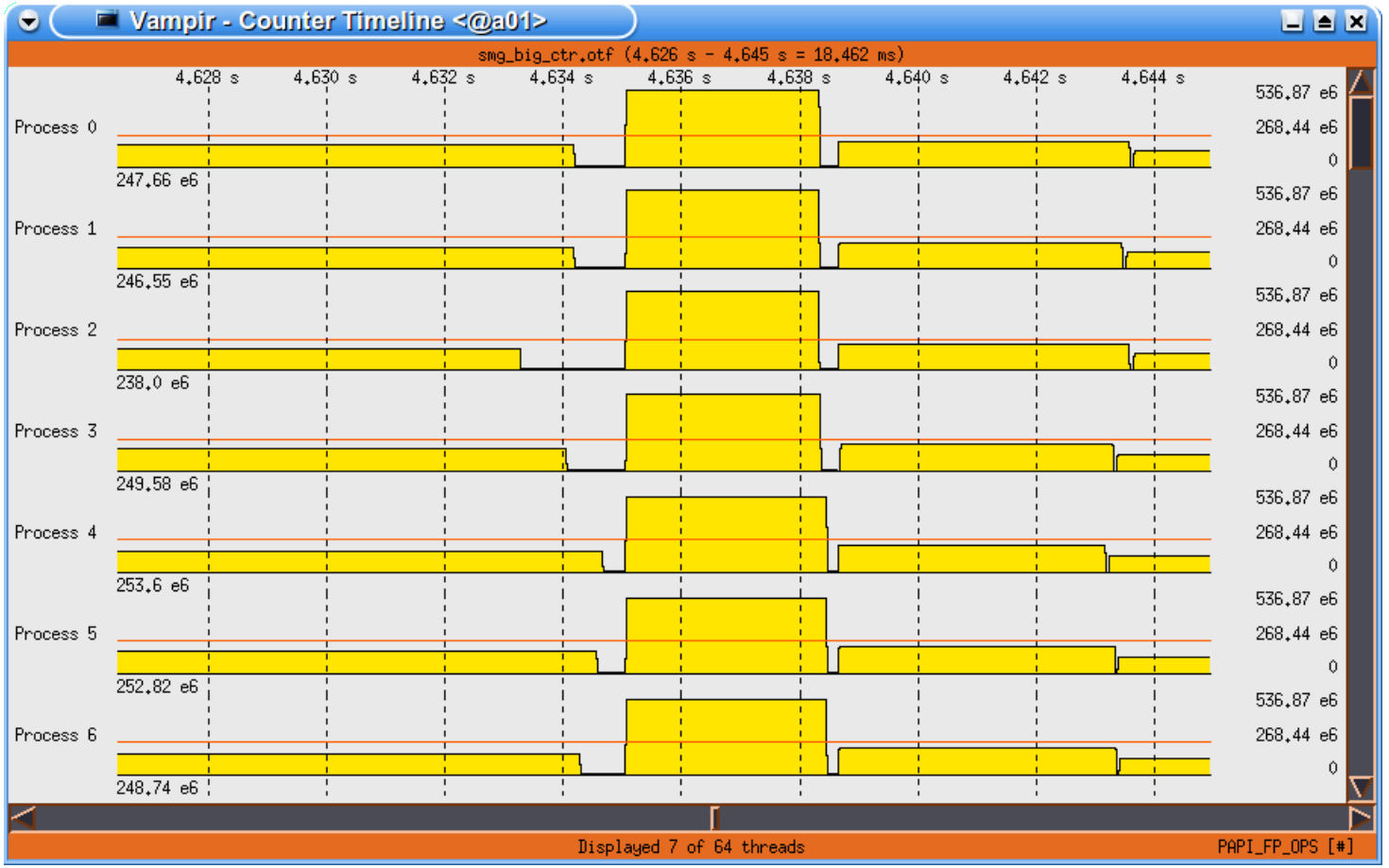
Message Statistics Display



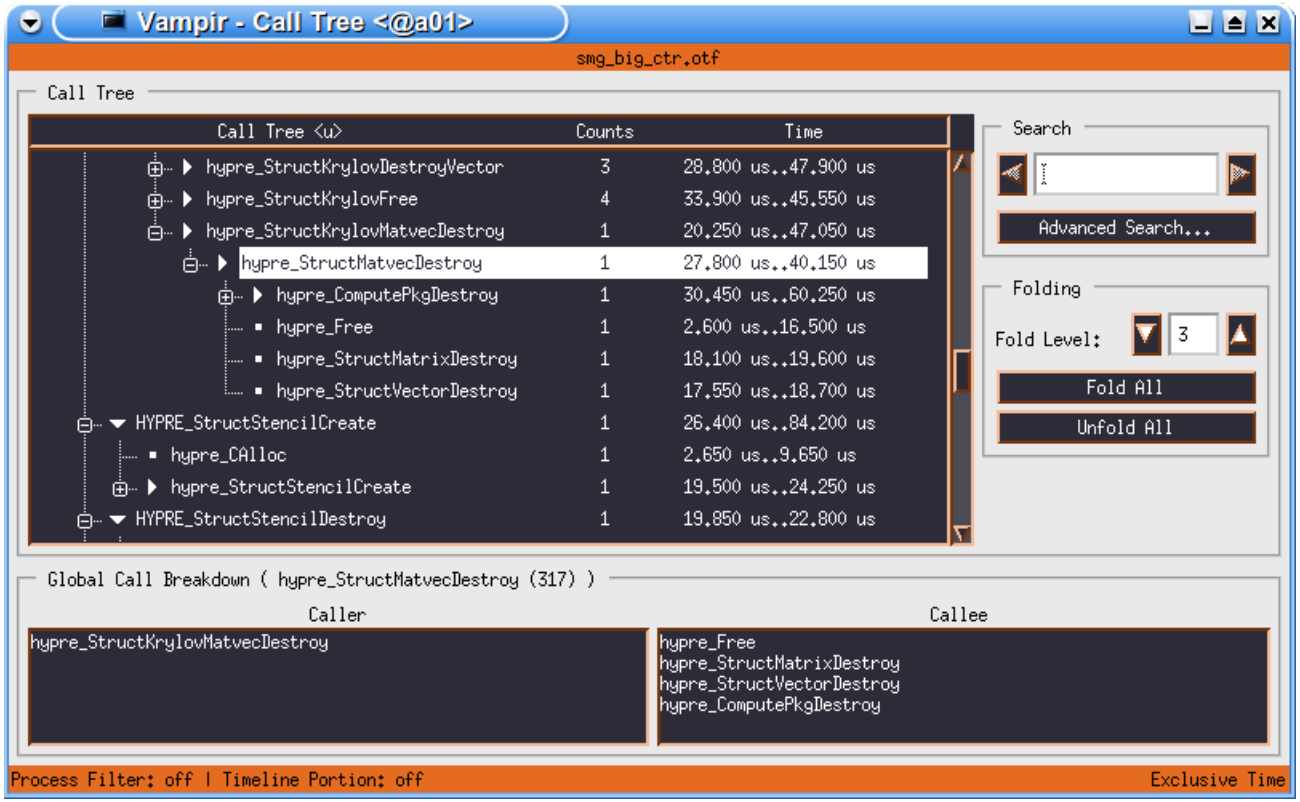
Collective Operation Statistics



Counter Timeline Display



Call Tree Display



Program Instrumentation

- detect run-time events (points of interest)
- pass information to run-time measurement library

Profile Recording

- collect aggregated information (Time, Counts, …)
- about program and system entities
  - functions, loops, basic blocks
  - application, processes, threads, …

Trace Recording

- save individual event records together with precise timestamp and process or thread ID
- plus event specific information

- What do you need to do for it?
  - Use VampirTrace
- Instrumentation (automatic with compiler wrappers)

```
CC=icc
CXX=icpc
F90=ifc
MPICC=mpicc
```

```
CC=vtcc
CXX=vtcxx
F90=vtf90
MPICC=vtcc
```

- Re-compile & re-link
- Trace Run (run with appropriate test data set)

- more details later

What does VampirTrace do in the background?

- Instrumentation:
  - via compiler wrappers
  - by underlying compiler with specific options
  - MPI instrumentation with replacement lib
  - OpenMP instrumentaion with Opari
  - also binary instrumentation with Dyninst
  - partial manual instrumentation

What does VampirTrace do in the background?

- Trace Run:
  - event data collection
  - precise time measurement
  - parallel timer synchronization
  - collecting parallel process/thread traces
  - collecting performance counters (from PAPI, memory usage, POSIX I/O calls and fork/system/exec calls, and more …)
  - filtering and grouping of function calls

Summary

- Vampir & VampirServer
  - interactive trace visualization and analysis
  - intuitive browsing and zooming
  - scalable to large trace data sizes (100GByte)
  - scalable to high parallelism (2000 processes)
- Vampir for Windows in progress, beta available

- VampirTrace
  - convenient instrumentation and measurement
  - hides away complicated details
  - provides many options and switches for experts
- VampirTrace is part of Open MPI 1.3

Partners with pictures

Forschungszentrum Jülich
- Jülich Supercomputing Centre

RWTH Aachen University
- Center for Computing and Communication

Technical University of Dresden
- Center for Information Services and High Performance Computing

University of Tennessee
- Innovative Computing Laboratory

Technical University of München
- Chair for Computer Architecture

University of Stuttgart
- High Performance Computing Centre

Partner logos