Analysis Guide

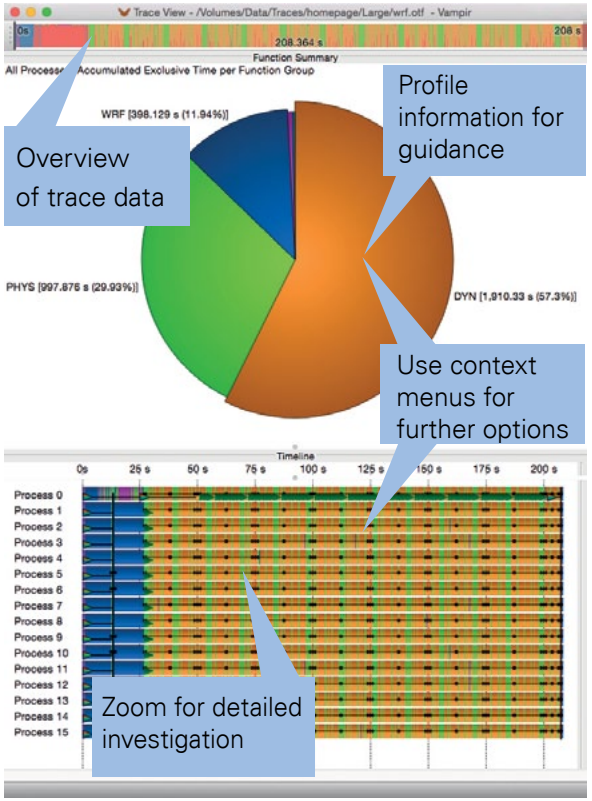
Vampir

- Install Vampir

```
./vampir-8.4.1-demo-linux-x86_64-\
   setup.bin --instdir=<inst_path>
```

- Following the Score-P example, you have a trace in the directory `scorep-jacobi`
- Open your trace file in Vampir for performance analysis

```
vampir scorep-jacobi/traces.otf2
```



Contact

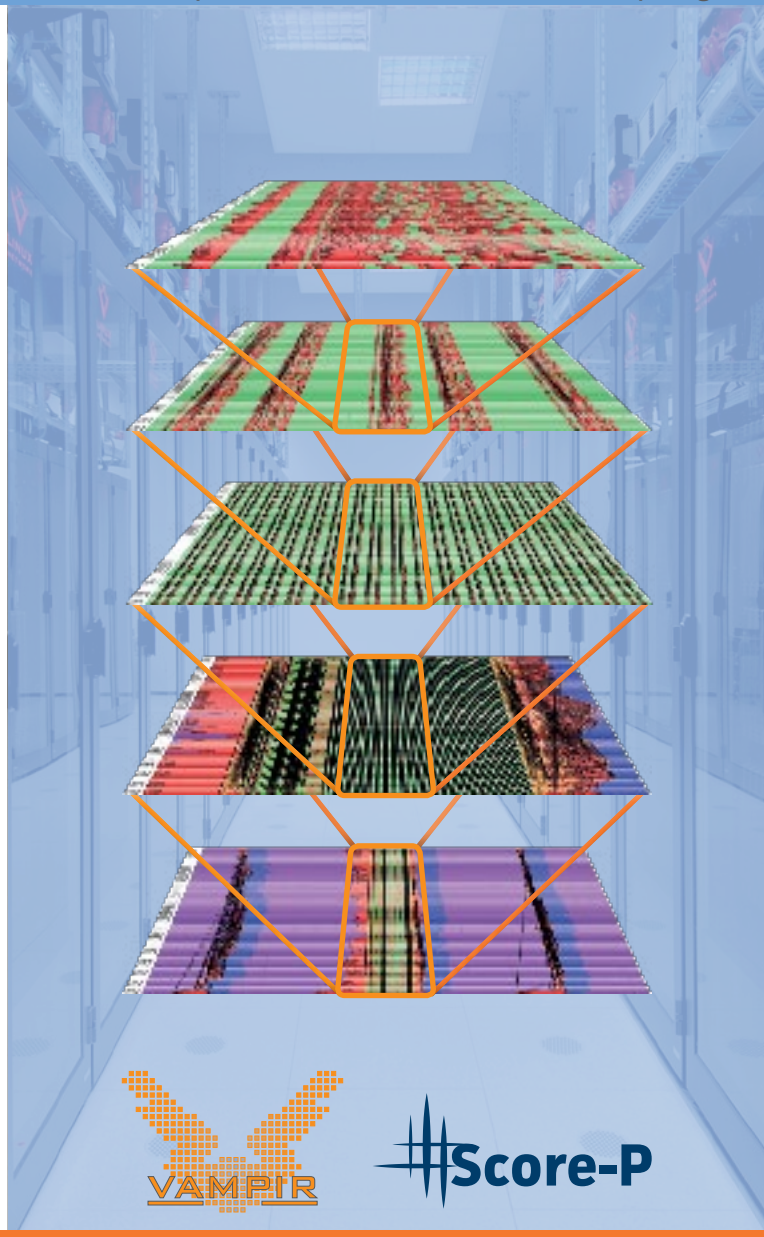
Score-P CONTACT
E-mail: support@score-p.org
Web: www.score-p.org

Vampir CONTACT/EU
GWT-TUD GmbH
E-mail: sales@vampir.eu
Web: www.vampir.eu

Vampir CONTACT/U.S.
ParaTools, Inc.
E-mail: info@paratools.com
Web: www.paratools.com

Performance Analysis

www.vampir.eu www.score-p.org

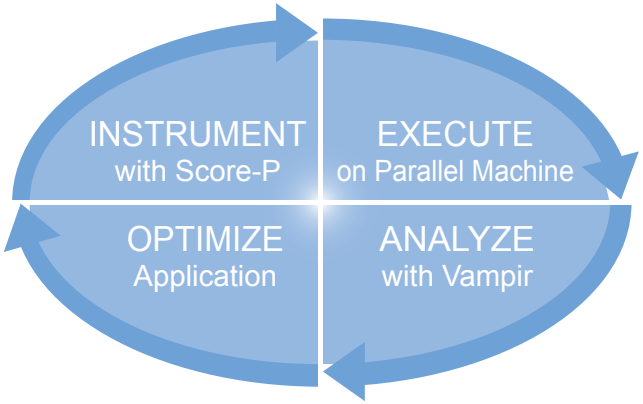


Goals

- Visualize application behavior
- Examine application execution during a given time in a given process or thread
- Investigate communication patterns of the application executed on a real system
- Identify
  - Bottlenecks
  - Load imbalances
  - Single core inefficiencies
  - I/O bottlenecks
- Understand performance impacts
- Optimize parallel applications

Workflow

- Prepare application (with symbols), insert extra code (probes/hooks)
- Collection of relevant data as input for the performance analysis
- Calculation of metrics, identification of performance metrics
- Presentation of results in an intuitive/ understandable form
- Modifications intended to eliminate/reduce performance problems



Score-P

- Monitor and record performance data of HPC applications
- Tool-suite for profiling, event tracing, and online analysis

Download: http://www.score-p.org

Vampir

- Scalable trace visualization
- In-depth analysis of parallel application runtime behavior, inter-process communication, synchronization, and I/O
- Statistics dynamically adapt to the selected time range

Download: https://www.vampir.eu/downloads/demo

Instrumentation Guide

Score-P

- Install Score-P

```
tar xvzf scorep-1.4.2.tar.gz
cd scorep-1.4.2
./configure –prefix=<inst_path>
make
make install
```

- Instrument your application

```
export PATH=\
   <inst_path>/bin:$PATH
cd test/jacobi/hybrid/C
scorep mpicc –fopenmp\
   jacobi.c main.c -o\
   jacobi.scorep
```

- Create a profile with full instrumentation

```
export SCOREP_EXPERIMENT_DIRECTORY=\
   scorep-jacobi
export OMP_NUM_THREADS=4
mpirun -np 4 ./jacobi.scorep
```

- Create initial filter file

```
echo "SCOREP_REGION_NAMES_BEGIN \
   EXCLUDE" >scorep.filt
export SCOREP_FILTERING_FILE=scorep.filt
```

- Incrementally refine the filter file using the output of the scorep-score tool and re-run the application until the overhead is acceptable

```
scorep-score -r -f ./scorep.filt \
   scorep-jacobi/profile.cubex
mpirun -np 4 ./jacobi.scorep
```

- For an in-depth analysis, generate a trace with the filter applied

```
export SCOREP_ENABLE_TRACING=true
export SCOREP_ENABLE_PROFILING=false
mpirun -np 4 ./jacobi.scorep
```