Vampir Performance Vis
Argonne Training Program
on Extreme-Scale Computing 2015

Guido Juckeland (guido.juckeland@tu-dresden.de)
Disclaimer

It is extremely easy to waste performance!

- Bad MPI (50-90%)
- No node-level parallelism (94%)
- No vectorization (75%)
- Bad memory access pattern (99%)
- In sum: 0.008% of the peak performance (785 GFLOPs of mira)
Performance tools will not automatically make your code run faster. They help you understand, what your code does and where to put in work.
Performance engineering workflow

- **Calculation of metrics**
- **Identification of performance problems**
- **Presentation of results**
- **Modifications intended to eliminate/reduce performance problem**
- **Collection of performance data**
- **Aggregation of performance data**
- **Prepare application with symbols**
- **Insert extra code (probes/hooks)**
- **Calculation of metrics**
- **Identification of performance problems**
- **Presentation of results**

Preparation → Measurement → Optimization → Analysis

ATPESC 2015 - Vampir Performance Vis
Agenda

Welcome to the Vampir Tool Suite

- Parallel Performance Analysis Approaches
- Mission
- Event Trace Visualization

The Vampir Workflow

- Score-P: Instrumentation & Run-Time Measurement
- Vampir & VampirServer

Vampir Performance Charts

Vampir Demo

- Tracing and Visualizing NPB-MZ-MPI / BT

Conclusions
Mission

Visualization of dynamics of concurrent processes

Two components / steps
- Monitor/Collector (Score-P)
- Charts/Browser (Vampir)

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?
Sampling vs. Tracing

Sampling

2011/06/30 10:15:12.672865 Enter foo
2011/06/30 10:15:12.894341 Leave foo

Tracing

Foo: Total Time 0.0815
Bar: Total Time 0.4711
Terms Used and How They Connect

Data Presentation

Data Recording

Data Acquisition

Analysis Layer

Analysis Technique

Profiling

Profiles

Summarization

Sampling

Tracing

Timelines

Logging

Event-based Instrumentation

Profiling and Tracing are connected through Profiles and Timelines.

Profiling and Summarization are connected through Profiles.

Summarization and Sampling are connected through Summarization.

Sampling and Event-based Instrumentation are connected through Sampling.

Profiling, Tracing, Summarization, and Sampling are part of the Analysis Layer.

Event-based Instrumentation, Profiling, Tracing, Summarization, and Sampling are part of the Analysis Technique.
So what is the right choice?

So, you have decided to understand what a program exactly does?

Congratulations!!!
You are ahead of 99% of your colleagues

Are you serious about this?

Yes!

What kind of professional are you?

Scientist

Engineer

Slacker

Go use gprof or the like...

Find interesting spots

Select magnification

Low

Med

High

Run in profiling mode

Use compiler, wrappers + filters

Instrument the critical parts

Technische Universität Dresden

ATPESC 2015 - Vampir Performance Vis
Event Trace Visualization with Vampir

- Show dynamic run-time behavior graphically at a fine level of detail
- Provide summaries (profiles) on performance metrics

Timeline charts
- Show application activities and communication along a time axis

Summary charts
- Provide quantitative results for the currently selected time interval
Agenda

Welcome to the Vampir Tool Suite
• Parallel Performance Analysis Approaches
• Mission
• Event Trace Visualization

The Vampir Workflow
• Score-P: Instrumentation & Run-Time Measurement
• Vampir & VampirServer

Vampir Performance Charts

Vampir Hands-on
• Tracing and Visualizing NPB-MZ-MPI / BT

Conclusions
Vampir Tool Suite Workflow

1. Instrument your application with Score-P

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

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

2. Perform a measurement run with **profiling enabled**

3. Use **scorep-score** to define an appropriate filter

4. Perform a measurement run with **tracing enabled** and the filter applied

5. Perform in-depth analysis on the trace data with **Vampir**
Score-P: Architecture

**Vampir**
- Event traces (OTF2)

**Scalasca**
- Call-path profiles (CUBE4, TAU)

**TAU**

**Periscope**
- Online interface

**Score-P measurement infrastructure**
- Hardware counter (PAPI, rusage)

**Application (MPI×OpenMP×CUDA)**

**Instrumentation wrapper**
- PMPI
- OPARI 2
- CUDA
- Compiler
- PDT
- User

**Compilers**
- OPARI 2
- POMP2
- CUDA
- Compiler
- TAU
- User

**Hardware计数器**
- PMPI
- OPARI 2
- CUDA
- Compiler
- PDT
- User

**Center for Information Services & High Performance Computing**
Score-P: Measurement Options

Measurements are configured via environment variables:

```bash
% scorep-info config-vars --full
```

- **SCOREP_ENABLE_PROFILING**
  - Description: Total memory in bytes for the measurement system
- **SCOREP_ENABLE_TRACING**
- **SCOREP_TOTAL_MEMORY**
  - Description: Name of the experiment directory
- **SCOREP_EXPERIMENT_DIRECTORY**
  - Description: A file name which contain the filter rules
- **SCOREP_FILTERING_FILE**
  - Description: PAPI metric names to measure
- **SCOREP_METRIC_PAPI**
  - Description: Resource usage metric names to measure

Profiles can be analyzed with **scorep-score**

- Helps to define appropriate filters for a tracing run
Vampir – Visualization Modes (1)

Directly on front end or local machine

```
% vampir
```
On local machine with remote VampirServer

```
% vampirserver start -n 12
```

```
% vampir
```

**VampirServer**

**Score-P**

**Many-Core Program**

**Large Trace File (OTF2)**

**LAN/WAN**

**MPI parallel application**

1.0 ms
2.0 ms
Welcome to the Vampir Tool Suite

- Parallel Performance Analysis Approaches
- Mission
- Event Trace Visualization

The Vampir Workflow

- Score-P: Instrumentation & Run-Time Measurement
- Vampir & VampirServer

Vampir Performance Charts

Vampir Demo

- Tracing and Visualizing the NPB-MZ-MPI / BT

Conclusions
Main Charts of Vampir

Timeline Charts:

- **Master Timeline**: all threads’ activities over time per thread
- **Summary Timeline**: all threads’ activities over time per act.
- **Performance Radar**: all threads’ perf-metric over time
- **Process Timeline**: single thread’s activities over time
- **Counter Data Timeline**: single threads perf-metric over time
Summary Charts:

- Function Summary
- Message Summary
- Process Summary
- Communication Matrix View
- I/O Summary
Master Timeline
Process and Counter Timeline

**Vampir**

**Timeline**

**Function Legend**
- Application
- DYN
- I/O
- IO
- MEM
- MPI
- PHYS
- VT_API
- WRF

**Process 0**
- Values of Counter "MEM_APP_ALLOC" over Time
- Values: 50 M, 0 M

**Process 63**
- Values of Counter "ru_utime" over Time
- Values: 750 k, 0 k

**Display**
- Process Timeline

**Type**
- Function

**Function Name**
- MPI_Wait

**Function Group**
- MPI

**Interval Begin**
- 18.121 s

**Interval End**
- 18.121 s

**Duration**
- 0 s

**Source File**
- Source Line
Where Do the Metrics Come From?

Built-In Editor
Function Summary

ATPESC 2015 - Vampir Performance Vis

Slide 26
Agenda

Welcome to the Vampir Tool Suite
- Parallel Performance Analysis Approaches
- Mission
- Event Trace Visualization

The Vampir Workflow
- Score-P: Instrumentation & Run-Time Measurement
- Vampir & VampirServer

Vampir Performance Charts

Vampir Demo
- Tracing and Visualizing NPB-MZ-MPI / BT → tonight

Conclusions
Agenda

Welcome to the Vampir Tool Suite
- Parallel Performance Analysis Approaches
- Mission
- Event Trace Visualization

The Vampir Workflow
- Score-P: Instrumentation & Run-Time Measurement
- Vampir & VampirServer

Vampir Performance Charts

Vampir Demo
- Tracing and Visualizing NPB-MZ-MPI / BT

Conclusions
Conclusions

**Vampir & VampirServer**
- Interactive trace visualization and analysis
- Intuitive browsing and zooming
- Scalable to large trace data sizes (20 TByte)
- Scalable to high parallelism (200000 processes)
- Vampir for Linux, Windows and Mac OS

**Score-P**
- Common instrumentation and measurement infrastructure for various analysis tools
- Hides away complicated details
- Provides many options and switches for experts
The people behind Vampir, Score-P, and OTF2:

Active
- Dr. Holger Brunst
- Jens Doleschal
- Ronald Geisler
- Tobias Hilbrich
- Matthias Jurenz
- Dr. Andreas Knüpfer
- Dr. Hartmut Mix
- Prof. Wolfgang E. Nagel
- Ronny Tschüter
- Michael Wagner
- Matthias Weber
- Bert Wesarg
- Thomas William
- Johannes Ziegenbalg

Retired
- Alfred Arnold
- Laszlo Barabas
- Ronny Brendel
- Heike McCraw/Jagode
- Shino Mathukutty George
- Daniel Hackenberg
- Robert Henschel
- Dr. Matthias Müller
- Reinhard Neumann
- Frank Noack
- Michael Peter
- Heide Rohling
- Johannes Spazier
- Frank Winkler
- Manuela Winkler
Vampir is available at http://www.vampir.eu
Vampir at Argonne NL: https://www.alcf.anl.gov/vampir
Get support via vampirsupport@zih.tu-dresden.de
Score-P: http://www.vi-hps.org/projects/score-p
Agenda

- Welcome to the Vampir Tool Suite
  - Parallel Performance Analysis Approaches
  - Mission
  - Event Trace Visualization

The Vampir Workflow
- Score-P: Instrumentation & Run-Time Measurement
- Vampir & VampirServer

Vampir Performance Charts

Vampir Demo
- Tracing and Visualizing NPB-MZ-MPI / BT → tonight

Conclusions
The NAS Parallel Benchmark suite (MPI+OpenMP version)

- Available from: [http://www.nas.nasa.gov/Software/NPB](http://www.nas.nasa.gov/Software/NPB)
- 3 benchmarks in Fortran77 (bt-mz, lu-mz, sp-mz)
- Configurable for various sizes & classes (S, W, A, B, C, D, E)

Benchmark configuration for demo:

- Benchmark name: **bt-mz**
- Number of MPI processes: **NPROCS=4**
- Benchmark class: **CLASS=W**
- What does it do?
  - Solves a discretized version of unsteady, compressible Navier-Stokes equations in three spatial dimensions
  - Performs 200 time-steps on a regular 3-dimensional grid
Connect to Mira and add Vampir to the SoftEnv system

```
% vi .soft
+vampir
% resoft
```

Copy sources to working directory

```
% cp /projects/Tools/vampir/tutorial/NPB3.3-MZ-MPI.tar.gz .
% tar xzvf NPB3.3-MZ-MPI.tar.gz
% cd NPB3.3-MZ-MPI
```

Compile the benchmark:

```
% make bt-mz CLASS=W NPROCS=4
cd BT-MZ; make CLASS=W NPROCS=4 VERSION=
make: Entering directory 'BT-MZ'
cd ../sys; cc -o setparams setparams.c
../sys/setparams bt-mz 4 W
mpixlf77_r -c -O3 -qsmp=omp -qextname=flush bt.f
[...]
Built executable ../bin/bt-mz_W.4
make: Leaving directory 'BT-MZ'
```
Copy jobscript and launch as a hybrid MPI+OpenMP application

```
cd bin
% cp ../jobscript/mira/run.sh .
% less run.sh
export OMP_NUM_THREADS=4
runjob -n 4 -p 4 --block $COBALT_PARTNAME --env-all : bt-mz_W.4
% qsub -A <projid> -t 10 -n 1 --mode script run.sh
% cat <jobid>.output
NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
Number of zones: 4 x 4
Iterations: 200 dt: 0.000800
Number of active processes: 4
Total number of threads: 16 (4.0 threads/process)

Time step 1
Time step 20
[...]
Time step 200
Verification Successful

BT-MZ Benchmark Completed.
Time in seconds = 2.27
```

Hint: save the benchmark output (or note the run time) to be able to refer to it later
Edit `config/make.def` to adjust build configuration

- Modify specification of compiler/linker: **MPIF77**

```plaintext
# SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS
#----------------------------------------------------------------------------------
# Items in this file may need to be changed for each platform.
#----------------------------------------------------------------------------------
...
#----------------------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#----------------------------------------------------------------------------------
#MPIF77 = mpixlf77_r

# Alternative variants to perform instrumentation
...
MPIF77 = scorep mpixlf77_r

# This links MPI Fortran programs; usually the same as ${MPIF77}
FLINK   = $(MPIF77)
...
```

Uncomment the Score-P compiler wrapper specification
Return to root directory and clean-up

```bash
% make clean
```

Re-build executable using Score-P compiler wrapper

```bash
% make bt-mz CLASS=W NPROCS=4
cd BT-MZ; make CLASS=W NPROCS=4 VERSION=
made: Entering directory 'BT-MZ'
cd ../sys; cc -o setparams setparams.c
../sys/setparams bt-mz 4 W
scorep mpixlf77_r -c -O3 -qsmp=omp -qextname=flush bt.f

[...]

cd ../common; scorep mpixlf77_r -c -O3 -qsmp=omp -qextname=flush timers.f
scorep mpixlf77_r -O3 -qsmp=omp -qextname=flush -o ../bin.scorep/bt-mz_W.4
bt.o initialize.o exact_solution.o exact_rhs.o set_constants.o \
adi.o rhs.o zone_setup.o x_solve.o y_solve.o exch_qbc.o \
solve_subs.o z_solve.o add.o error.o verify.o mpi_setup.o \
../common/print_results.o ../common/timers.o
Built executable ../bin.scorep/bt-mz_W.4
make: Leaving directory 'BT-MZ'
```
Change to the directory containing the new executable before running it and adjust configuration

```bash
% cd bin.scorep
% cp ../jobscript/mira/* .
% less run_profile.sh
export SCOREP_ENABLE_TRACING=false
export SCOREP_ENABLE_PROFLING=true
export SCOREP_TOTAL_MEMORY=100M
export SCOREP_EXPERIMENT_DIRECTORY=scorep_bt-mz_W_4x4_sum
export OMP_NUM_THREADS=4
runjob -n 4 -p 4 --block $COBALT_PARTNAME --env-all : bt-mz_W.4
% qsub -A <projid> -t 10 -n 1 --mode script run_profile.sh
% cat <jobid>.outpout
NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
Number of zones:  4 x  4
[...]
Time step  200
Verification Successful

BT-MZ Benchmark Completed.
Time in seconds = 12.74
```
Creates experiment directory ./scorep_bt-mz_W_4x4_sum containing

- A record of the measurement configuration (scorep.cfg)
- The analysis report that was collated after measurement (profile.cubex)

```
% ls
... scorep_bt-mz_W_4x4_sum
% ls scorep_bt-mz_W_4x4_sum
profile.cubex scorep.cfg
```
Report scoring as textual output

```bash
% scorep-score scorep_bt-mz_W_4x4_sum/profile.cubex
```

Estimated aggregate size of event trace:
Estimated requirements for largest trace buffer (max_tbc):
(hint: When tracing set SCOREP_TOTAL_MEMORY > max_tbc to avoid intermediate flushes or reduce requirements using file listing names of USR regions to be filtered.)

<table>
<thead>
<tr>
<th>flt type</th>
<th>max_tbc</th>
<th>time</th>
<th>% region</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>235123428</td>
<td>419.92</td>
<td>100.0</td>
</tr>
<tr>
<td>USR</td>
<td>232516724</td>
<td>78.19</td>
<td>18.6</td>
</tr>
<tr>
<td>OMP</td>
<td>5973040</td>
<td>121.45</td>
<td>28.9</td>
</tr>
<tr>
<td>COM</td>
<td>314710</td>
<td>1.38</td>
<td>0.3</td>
</tr>
<tr>
<td>MPI</td>
<td>88898</td>
<td>218.90</td>
<td>52.1</td>
</tr>
</tbody>
</table>

868 MB total memory
224 MB per rank!

Region/callpath classification

- MPI (pure MPI library functions)
- OMP (pure OpenMP functions/regions)
- USR (user-level source local computation)
- COM ("combined" USR + OpenMP/MPI)
- ANY/ALL (aggregate of all region types)
Score report breakdown by region

```
% scorep-score -r scorep_bt-mz_W_4x4_sum/profile.cubex

<table>
<thead>
<tr>
<th>flt_type</th>
<th>max_tbc</th>
<th>time</th>
<th>% region</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>235123428</td>
<td>419.92</td>
<td>100.0</td>
</tr>
<tr>
<td>USR</td>
<td>232516724</td>
<td>78.19</td>
<td>18.6</td>
</tr>
<tr>
<td>OMP</td>
<td>5973040</td>
<td>121.45</td>
<td>28.9</td>
</tr>
<tr>
<td>COM</td>
<td>314710</td>
<td>1.38</td>
<td>0.3</td>
</tr>
<tr>
<td>MPI</td>
<td>88898</td>
<td>218.90</td>
<td>52.1</td>
</tr>
</tbody>
</table>
```

More than 223 MB just for these 6 regions

USR
USR
USR
USR
COM
OMP
OMPI
USR

More than 223 MB just for these 6 regions
Report scoring with prospective filter listing 6 USR regions

```bash
% cat ..//config/scorep.filt
SCOREP_REGION_NAMES_BEGIN EXCLUDE
binvcrhs*
matmul_sub*
matvec_sub*
exact_solution*
binvrhs*
lhs*init*
timer_*

% scorep-score -f ..//config/scorep.filt scorep_bt-mz_W_4x4_sum/profile.cubex
```

Estimated aggregate size of event trace: 20482398 bytes
Estimated requirements for largest trace buffer (max_tbc): 6377242 bytes
(hint: When tracing set SCOREP_TOTAL_MEMORY > max_tbc to avoid intermediate flushes or reduce requirements using file listing names of USR regions to be filtered.)

20 MB of memory in total, 6 MB per rank!
Available PAPI metrics

- Preset events: common set of events deemed relevant and useful for application performance tuning
  - Abstraction from specific hardware performance counters, mapping onto available events done by PAPI internally

```bash
% qsub -A <projid> -n 1 --mode c1 --proccount 1 -t 10 \ /soft/perftools/papi/bin/papi_avail
% cat <jobid>.output
```

- Native events: set of all events that are available on the CPU (platform dependent)

```bash
% qsub -A <projid> -n 1 --mode c1 --proccount 1 -t 10 \ /soft/perftools/papi/bin/papi_native_avail
% cat <jobid>.output
```

Note:
Due to hardware restrictions
- number of concurrently recorded events is limited
- there may be invalid combinations of concurrently recorded events
Re-run the application using the tracing mode of Score-P

```bash
% cd bin.scorep
% less run_trace.sh
  export SCOREP_ENABLE_TRACING=true
  export SCOREP_ENABLE_PROFILING=false
  export SCOREP_FILTERING_FILE=../config/scorep.filt
  export SCOREP_TOTAL_MEMORY=100M
  export SCOREP_EXPERIMENT_DIRECTORY=scorep bt-mz_W 4x4_trace
  export SCOREP_METRIC_PAPI=PAPI_FP_OPS, PAPI_L1_DCM
  export OMP_NUM_THREADS=4
  runjob -n 4 -p 4 --block $COBALT_PARTNAME --env-all: bt-mz_W.4
% qsub -A <projid> -t 10 -n 1 --mode script run_trace.sh
% cat <jobid>.output
NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
Number of zones:  4 x  4
    Time step  200
 Verification Successful

BT-MZ Benchmark Completed.
Time in seconds = 3.49
```
Download and install VampirClient for target platform

```
# Linux 64bit
$ scp <user>@mira.alcf.anl.gov:/projects/Tools/vampir/vampir-gui/vampir-*-x86_64.bin .
$ scp <user>@mira.alcf.anl.gov:/projects/Tools/vampir/vampir-gui/vampir-remote.license .
$ bash ./vampir-*
```

Start VampirServer and follow output instructions

```
$ vampirserver start -a <projid> -n 16
Launching VampirServer...
Submitting PBS batch job (this might take a while)...
** Project 'tools'; job rerouted to queue 'prod-short'
VampirServer 8.2.1  (r8876)
Licensed to Mira Performance Boot Camp 2014
Running 15 analysis processes... (abort with vampirserver stop 28448)
VampirServer <28448> listens on: Q2G-I5-J01.mira.i2b:30066
Please run:
    ssh -L 30001:Q2G-I5-J01.mira.i2b:30066 <user>@mira.alcf.anl.gov
on your desktop to create ssh tunnel to VampirServer.
Start vampir on your desktop and choose 'Open Other -> Remote File'
    Description: mira,  Server: localhost,  Port: 30001
    Authentication: None
    Connection type: Socket
    Ignore "More Options"
```
NPB-MZ-MPI / BT Trace Analysis with Vampir

ATPESC 2015 - Vampir Performance Vis