TECHNISCHE UNIVERSITÄT DRESDEN

Center for Information Services and High Performance Computing (ZIH)

Extending the Functionality of Score-P through Plugins: Interfaces and Examples

10th International Parallel Tools Workshop, Oct 5th 2016

Robert Schöne

Ronny Tschüter

Thomas Ilsche

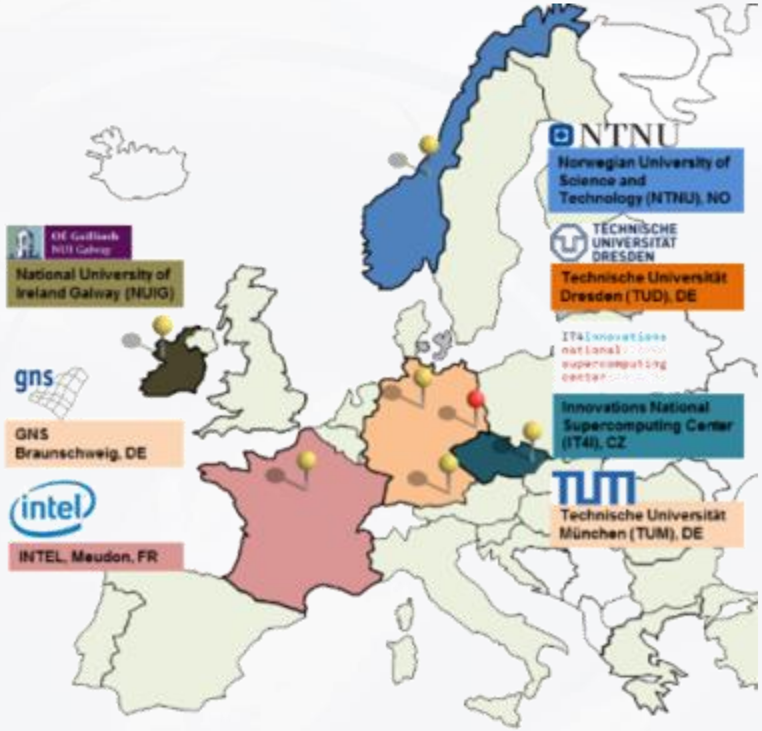
Joseph Schuchart

Daniel Hackenberg

ZIH
Center for Information Services &
High Performance Computing

READEX

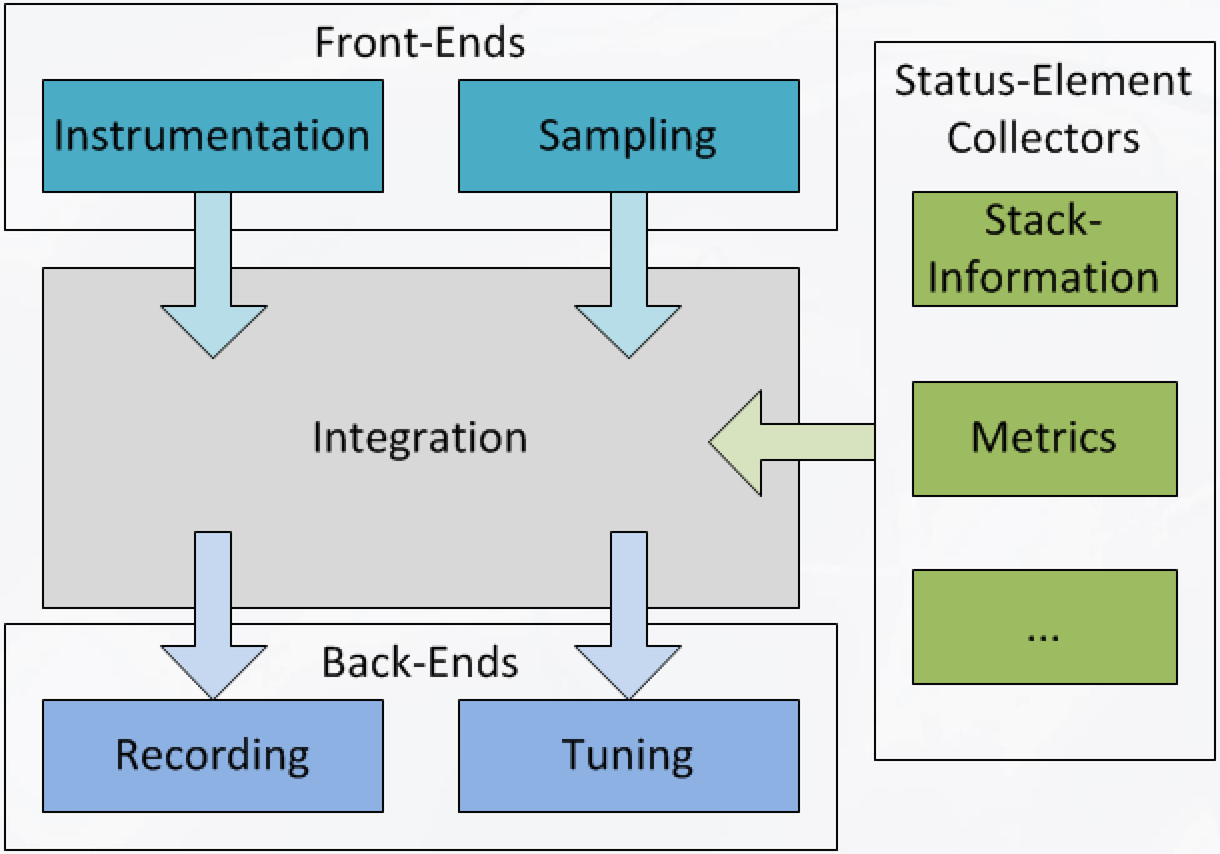
- Technische Universität Dresden (Coordinator), Germany
- Norwegian University of Science and Technology, Norway
- IT4Innovations, Czech Republic
- Technische Universität München, Germany
- Intel European Exascale Labs, France
- GNS Braunschweig, Germany
- National University of Ireland Galway, Ireland


NTNU
Norwegian University of Science and Technology (NTNU), NO
TECHNISCHE UNIVERSITÄT DRESDEN
Technische Universität Dresden (TUD), DE
IT4Innovations national supercomputing center
Innovations National Supercomputing Center (IT4I), CZ
TUM
Technische Universität München (TUM), DE
OÉ Gaillimh NUI Galway
National University of Ireland Galway (NUIG)
gns
GNS Braunschweig, DE
intel
INTEL, Meudon, FR

General Overview Measurement/Tuning Infrastructure

Interrupt workload, create initial status

Collect current status of workload



Provide status information

Act depending on status

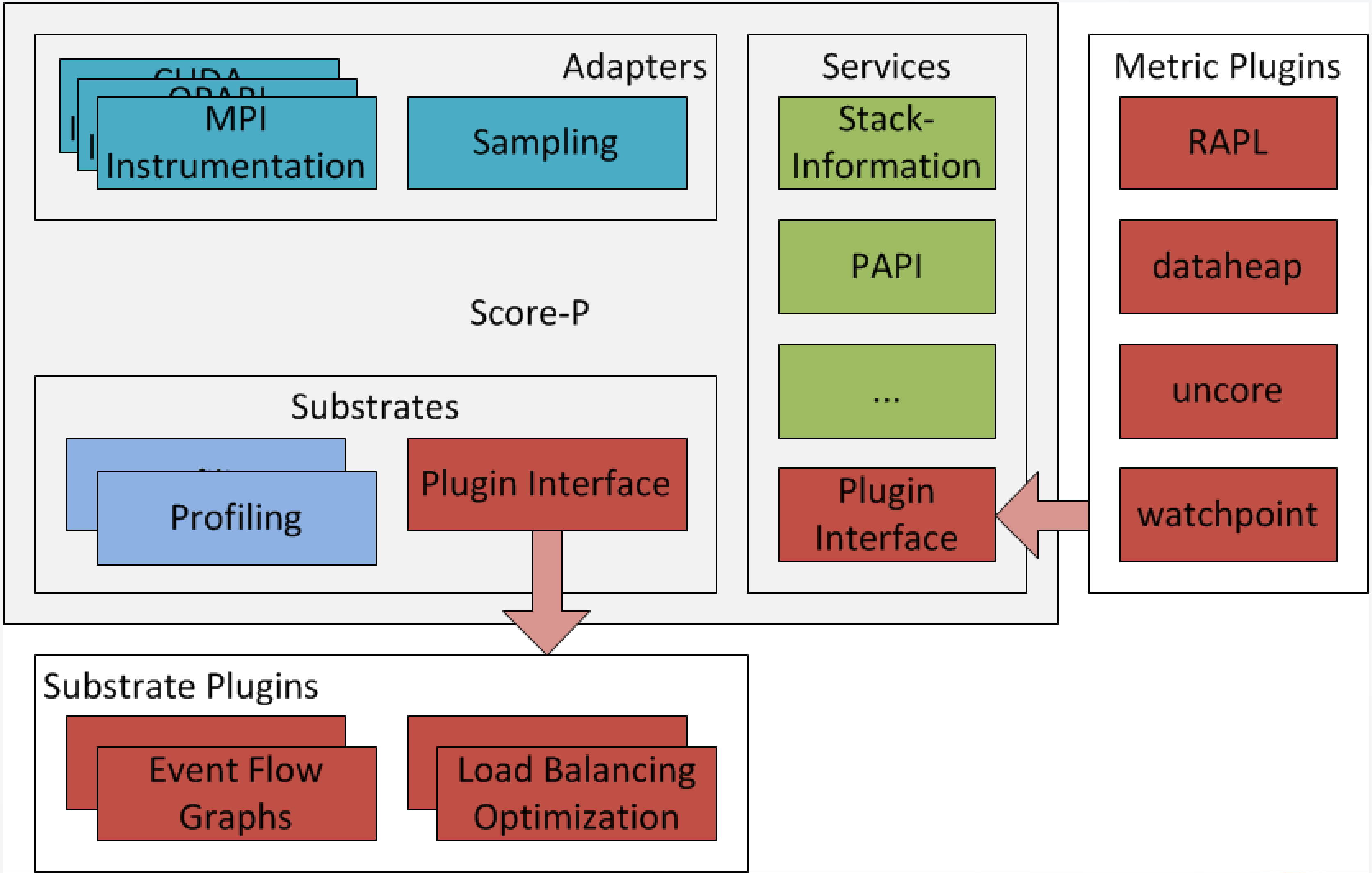
Demands (Metrics)

- Performance counter metrics
  - Get information on performance bottlenecks and dynamism
  - Part of Score-P
  - PAPI
  - perf
- Non traditional information
  - Open interface for different metrics
  - Power consumption information
  - Different scopes
  - Score-P Metric Plugins interface already available
  - Comparable to VampirTrace plugins [STHI11]

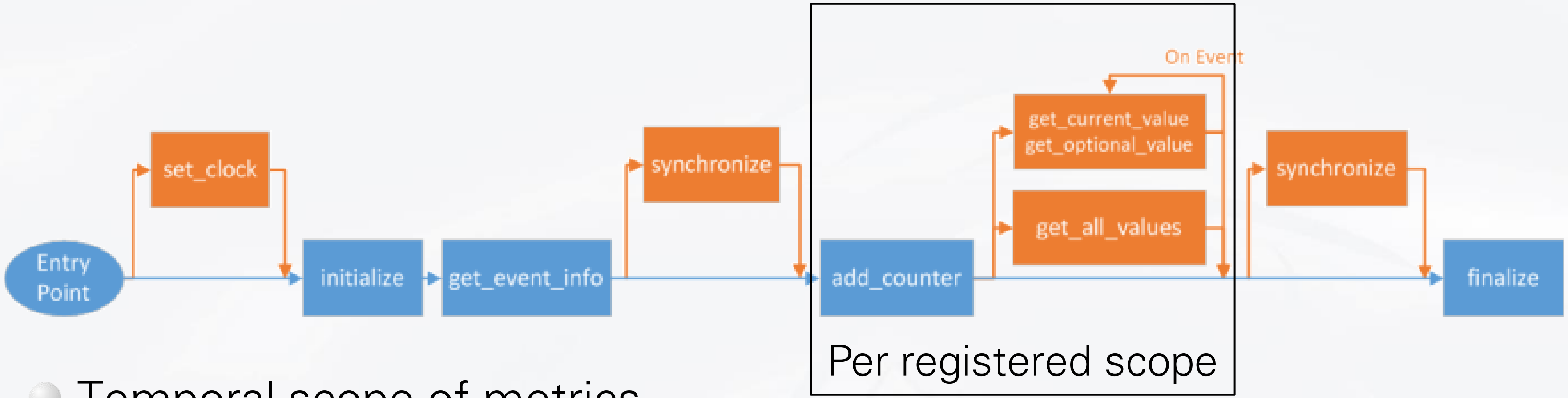
Demands (Back-Ends)

- Consume program events
  - Act according to specific **regions**
  - Act according to **metrics**
  - Act depending on **location**
  - Act on instructions from **Online-Access** server
  - READEX Runtime Library
- Goal: Adapt HW/SW environment to increase energy efficiency
  - Independent of READEX
  - Different for different architectures → exchangeable

Score-P overview



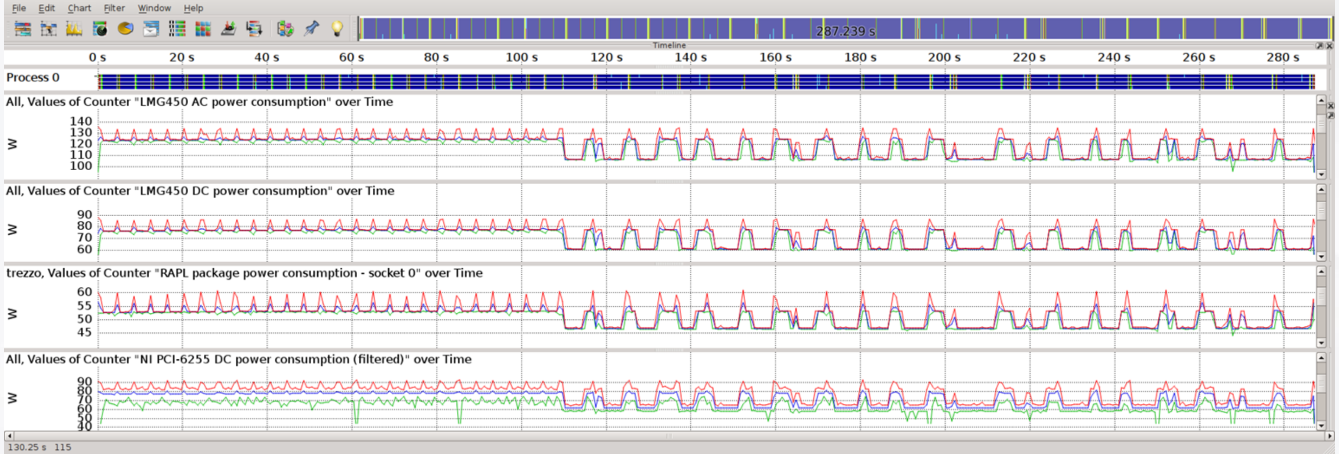
Metric Plugins



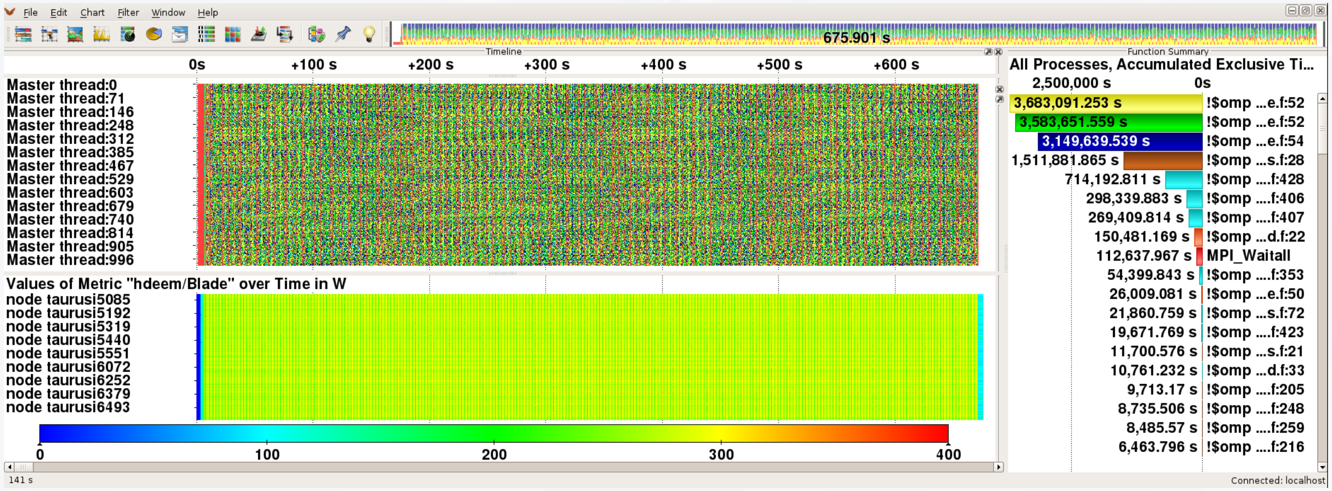
Per registered scope

- Temporal scope of metrics
  - synchronous/strictly synchronous/asynchronous
  - Instantaneous/backward-related/forward-related
- Spatial scope of metrics
  - Per thread/process/computing node/system
- `SCOREP_METRIC_PLUGINS=foo # libfoo.so`
- `SCOREP_METRIC_FOO=bar # event bar in libfoo`
- Plugins on https://github.com/score-p/

Examples Power Consumption


File Edit Chart Filter Window Help
287.239 s
Timeline
0 s
20 s
40 s
60 s
80 s
100 s
120 s
140 s
160 s
180 s
200 s
220 s
240 s
260 s
280 s
Process 0
All, Values of Counter "LMG450 AC power consumption" over Time
W
140
130
120
110
100
All, Values of Counter "LMG450 DC power consumption" over Time
90
80
70
60
trezzo, Values of Counter "RAPL package power consumption - socket 0" over Time
60
55
50
45
All, Values of Counter "NI PCI-6255 DC power consumption (filtered)" over Time
90
80
70
60
50
40
130.25 s 115

SPEC OMP applu, measurements with ZES-ZIMMER LMG 450/dataheap, RAPL, and HAEC infrastructure [HIS+13]



All Processes, Accumulated Exclusive Ti...

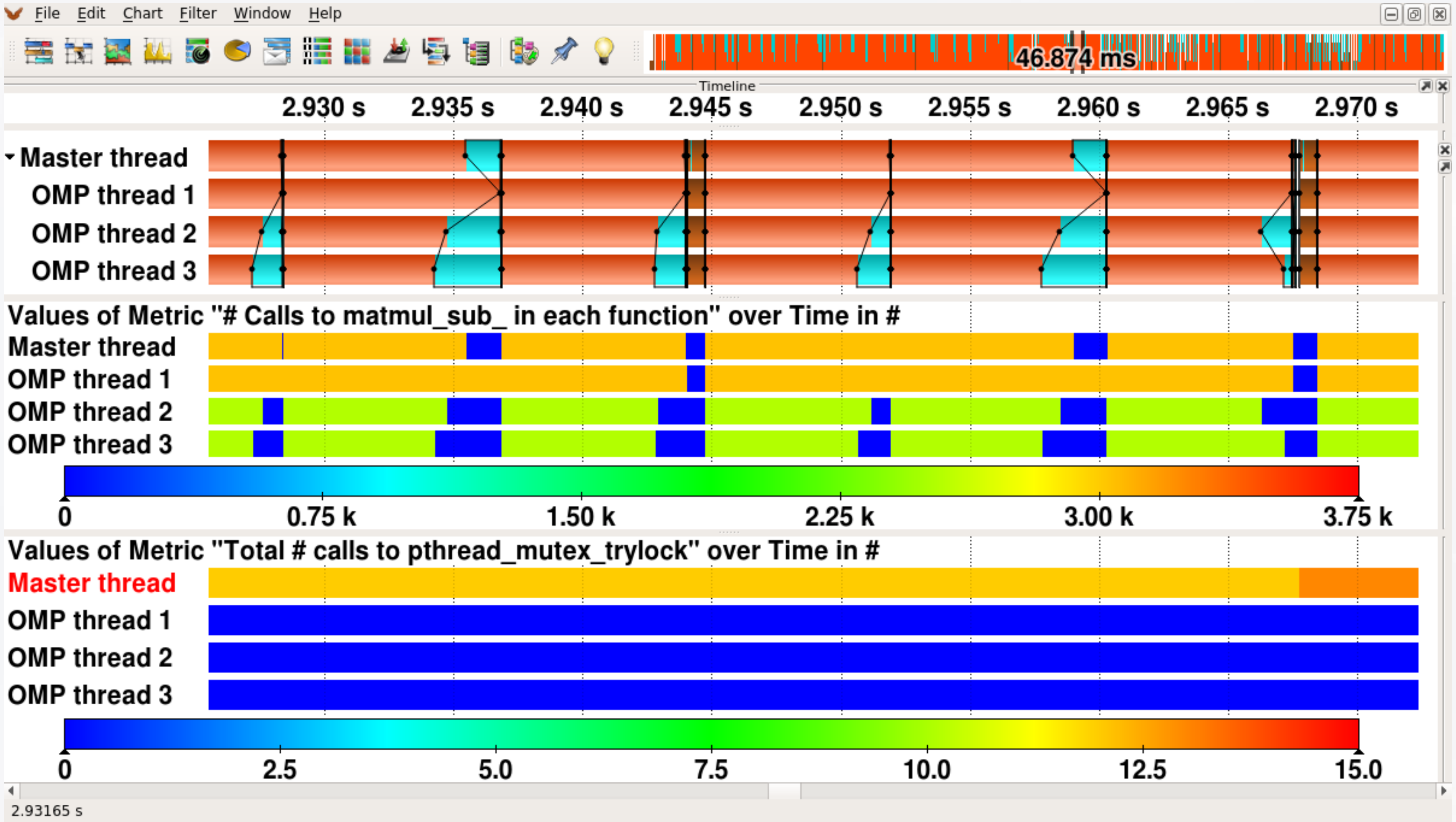
2,500,000 s	0s
3,683,091.253 s	!$omp ...e.f:52
3,583,651.559 s	!$omp ...e.f:52
3,149,639.539 s	!$omp ...e.f:54
1,511,881.865 s	!$omp ...s.f:28
714,192.811 s	!$omp f:428
298,339.883 s	!$omp f:406
269,409.814 s	!$omp f:407
150,481.169 s	!$omp ...d.f:22
112,637.967 s	MPI_Waitall
54,399.843 s	!$omp f:353
26,009.081 s	!$omp ...e.f:50
21,860.759 s	!$omp ...s.f:72
19,671.769 s	!$omp f:423
11,700.576 s	!$omp ...s.f:21
10,761.232 s	!$omp ...d.f:33
9,713.17 s	!$omp f:205
8,735.506 s	!$omp f:248
8,485.57 s	!$omp f:259
6,463.796 s	!$omp f:216

NPB-MZ benchmark BT, Class F, 1024 nodes, measurement with HDEEM [HIS+14]

Example Watchpoints

- Synchronous
  - Register variables or functions that shall be watched
  - Watch for number of read/write/execute access to variable
  - E.g., `SCOREP_METRIC_PERFBREAKPOINT_PLUGIN=x_pthread_mutex_trylock` to watch the number of calls of `pthread_mutex_trylock`
  - Statistic information on accesses to global variables
- Asynchronous
  - Register variables whose content will be watched
  - Parse binary with libbfd, find local & global variables
  - E.g., `SCOREP_METRIC_WATCHPOINT_PLUGIN=foo:uint64_t:bar` registers the content of the local variable `bar`, defined in function `foo`
  - High overhead (from kernel infrastructure), for debugging purposes

Example Watchpoints (Synchronous)



NPB-OpenMP benchmark BT, CLASS W, 4 Threads

Example Watchpoints (Asynchronous)

- OpenMP parallel program, 2 Threads
- Threads concurrently write their loop iteration to a global variable
- Depending on scheduling strategy, the content of the global variable changes over time

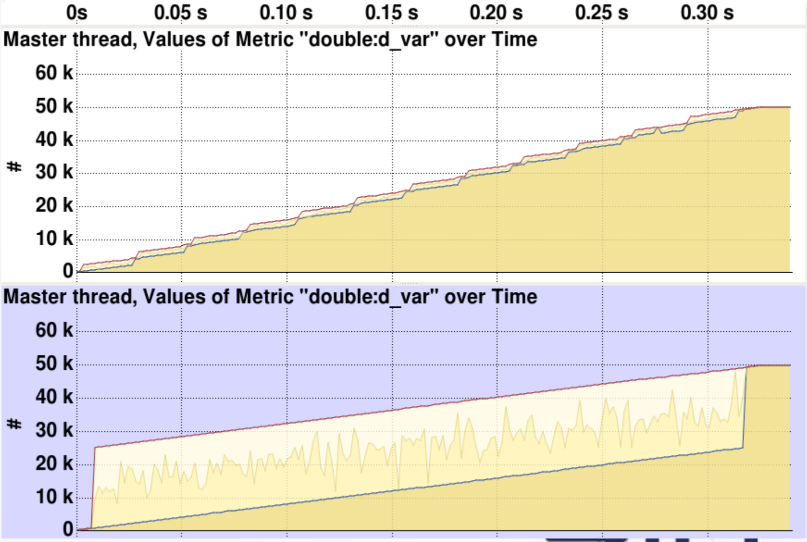
```
static double d_var=0;
void func( int i ){
    d_var=0.5*i;
}
int main( int argc, char** argv ){
    int i=0;
#pragma omp parallel for schedule(runtime)
    for(i=0;i<100000;i++){
        func(i);
    }
    return 0;
}
```

- White:

  `OMP_SCHEDULE=dynamic,4096`

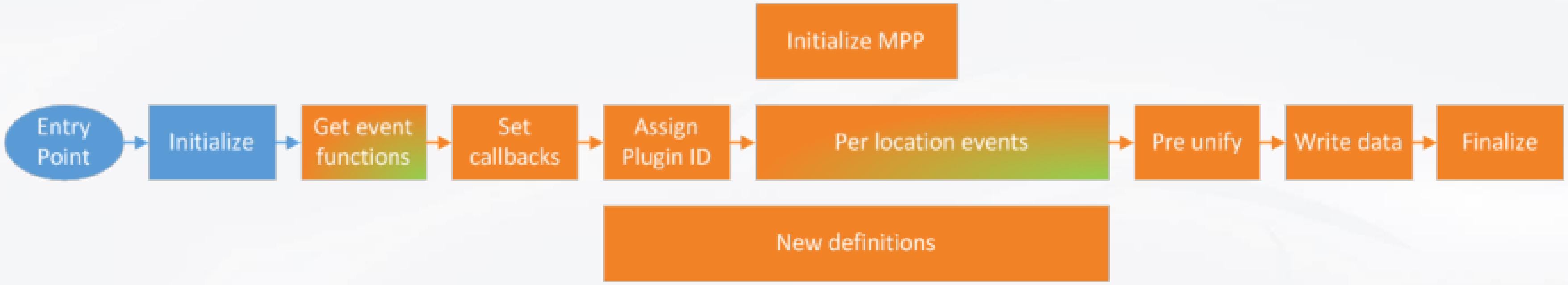
- Purple:

  `OMP_SCHEDULE=static`



European Commission
Horizon 2020
European Union funding
for Research & Innovation

Substrate Plugins

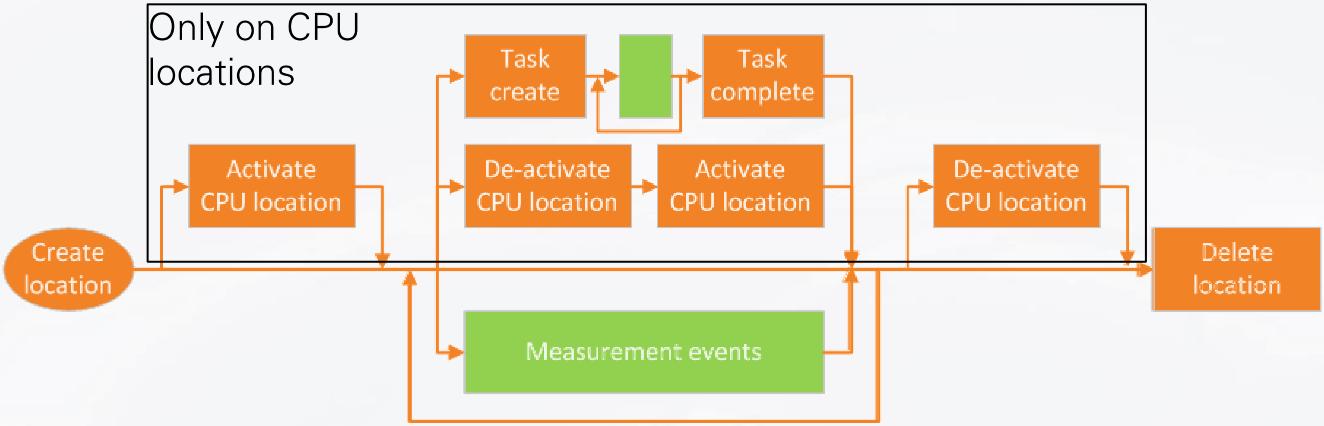


Per process calls to Substrate plugins

- Based on internal Substrate infrastructure
  - Management functions – related to Score-P internal processing
  - Event functions – related to program internals
- Only two non-optional functions
- Callbacks for plugins to get metadata for events from Score-P
- `export SCOREP_SUBSTRATE_PLUGINS=foo`

```
# load libscorep_substrate_foo
```

Substrate Plugins



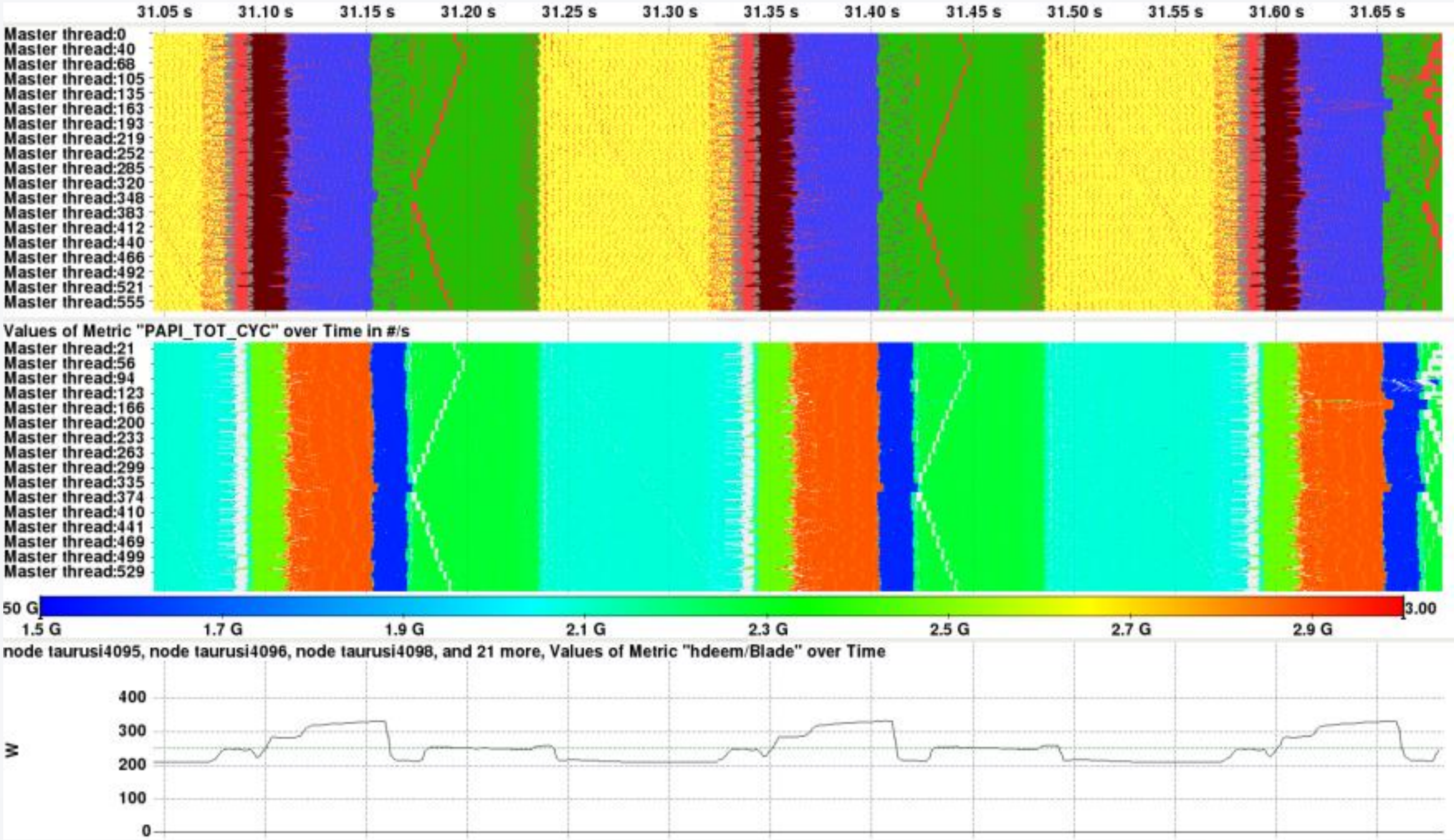
Per location calls to Substrate plugins

- Calls for CPU locations
- All measurement events get location and timestamp
- Plugins can use callbacks to get metadata from handles (e.g., name of entered region, type of location, …)
- Support for storage of location specific data in Score-P (thread local storage)

Examples

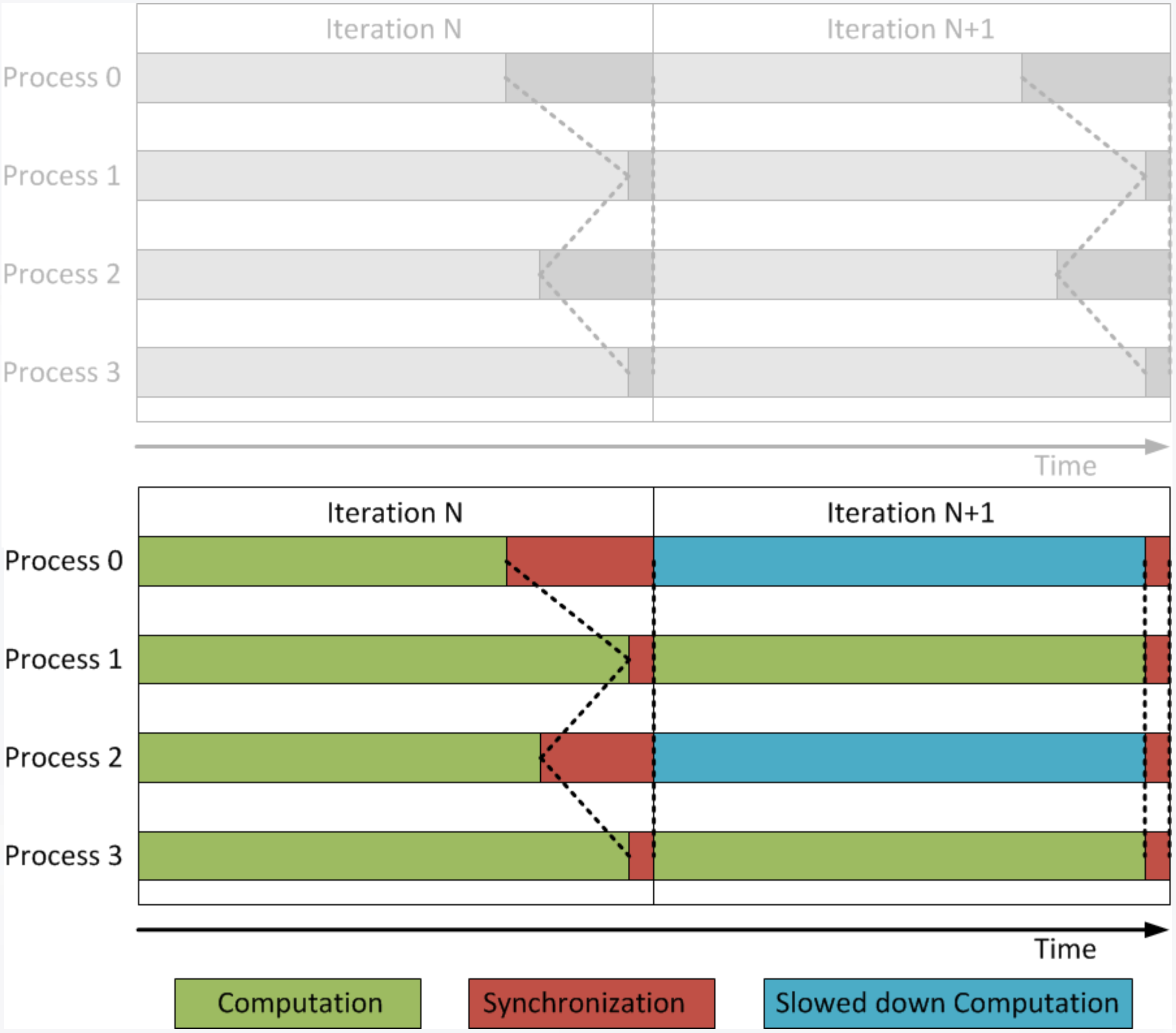
- Per region optimization using libadapt
  - As presented in [SM13]
  - Plugin reads config file and registers for enter/exit events to change HW/SW environment
  - DVFS/DCT/low-level hardware options
- Load balancing using DVFS
  - Like [RLdS+09], plus OpenMP
  - Successive execution of (compute region, synchronization region) pairs
  - Slow down computation to arrive just in time for synchronization
  - Plugin registers for enter/exit events
- Event Flow Graphs
  - Based on [AFL14]
  - Write event flow graph of exclusive regions as graphviz diagram

Region Based Optimization

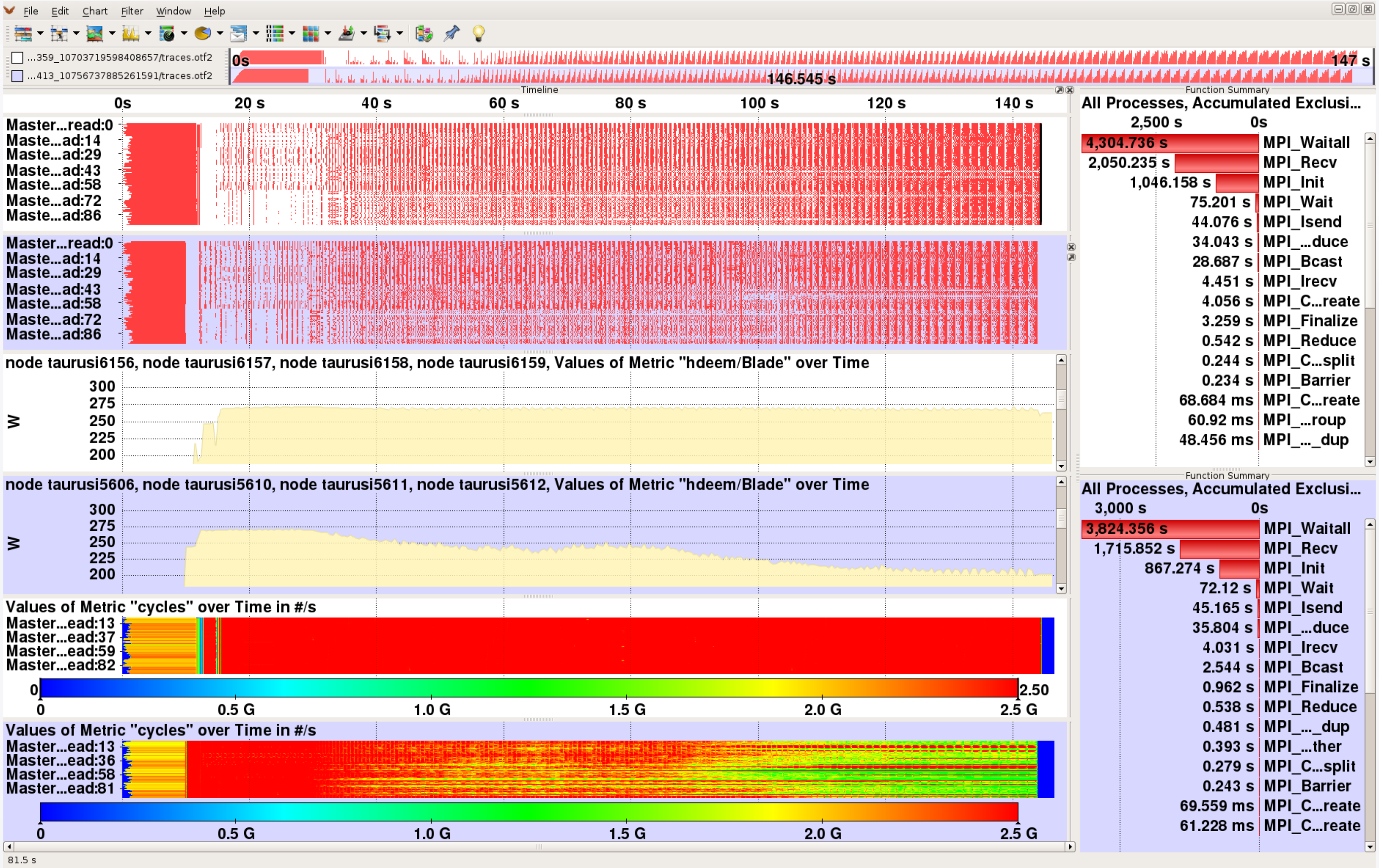


NPB-MPI benchmark BT, Class D, 576 ranks, optimized version: varying frequency and power consumption (HDEEM)

Load Balancing Substrate



Load Balancing SPEC COSMO FD4 via DVFS



All Processes, Accumulated Exclusi...

2,500 s	0s
4,304.736 s	MPI_Waitall
2,050.235 s	MPI_Recv
1,046.158 s	MPI_Init
75.201 s	MPI_Wait
44.076 s	MPI_Isend
34.043 s	MPI_...duce
28.687 s	MPI_Bcast
4.451 s	MPI_Irecv
4.056 s	MPI_C...reate
3.259 s	MPI_Finalize
0.542 s	MPI_Reduce
0.244 s	MPI_C...split
0.234 s	MPI_Barrier
68.684 ms	MPI_C...reate
60.92 ms	MPI_...roup
48.456 ms	MPI_..._dup

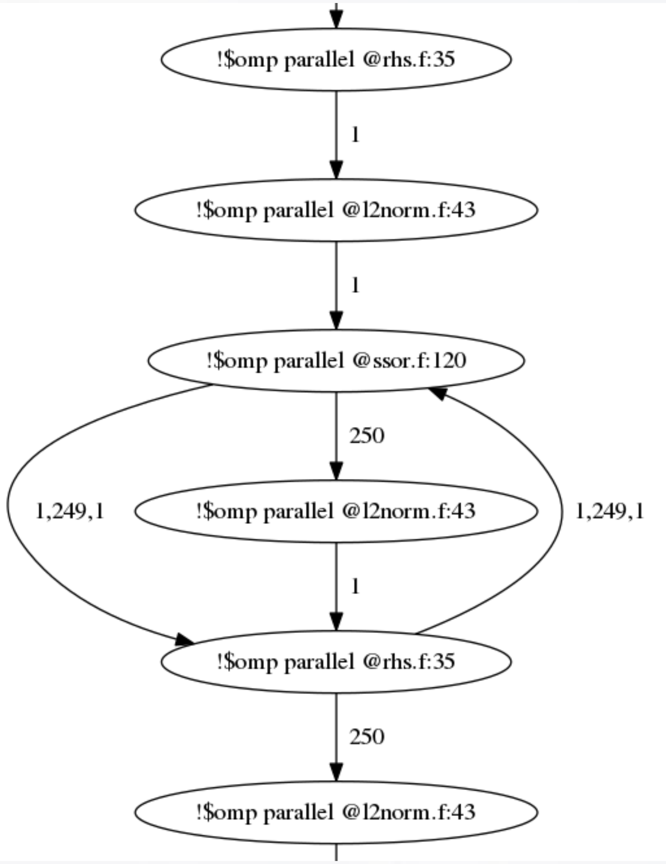
All Processes, Accumulated Exclusi...

3,000 s	0s
3,824.356 s	MPI_Waitall
1,715.852 s	MPI_Recv
867.274 s	MPI_Init
72.12 s	MPI_Wait
45.165 s	MPI_Isend
35.804 s	MPI_...duce
4.031 s	MPI_Irecv
2.544 s	MPI_Bcast
0.962 s	MPI_Finalize
0.538 s	MPI_Reduce
0.481 s	MPI_..._dup
0.393 s	MPI_...ther
0.279 s	MPI_C...split
0.243 s	MPI_Barrier
69.559 ms	MPI_C...reate
61.228 ms	MPI_C...reate

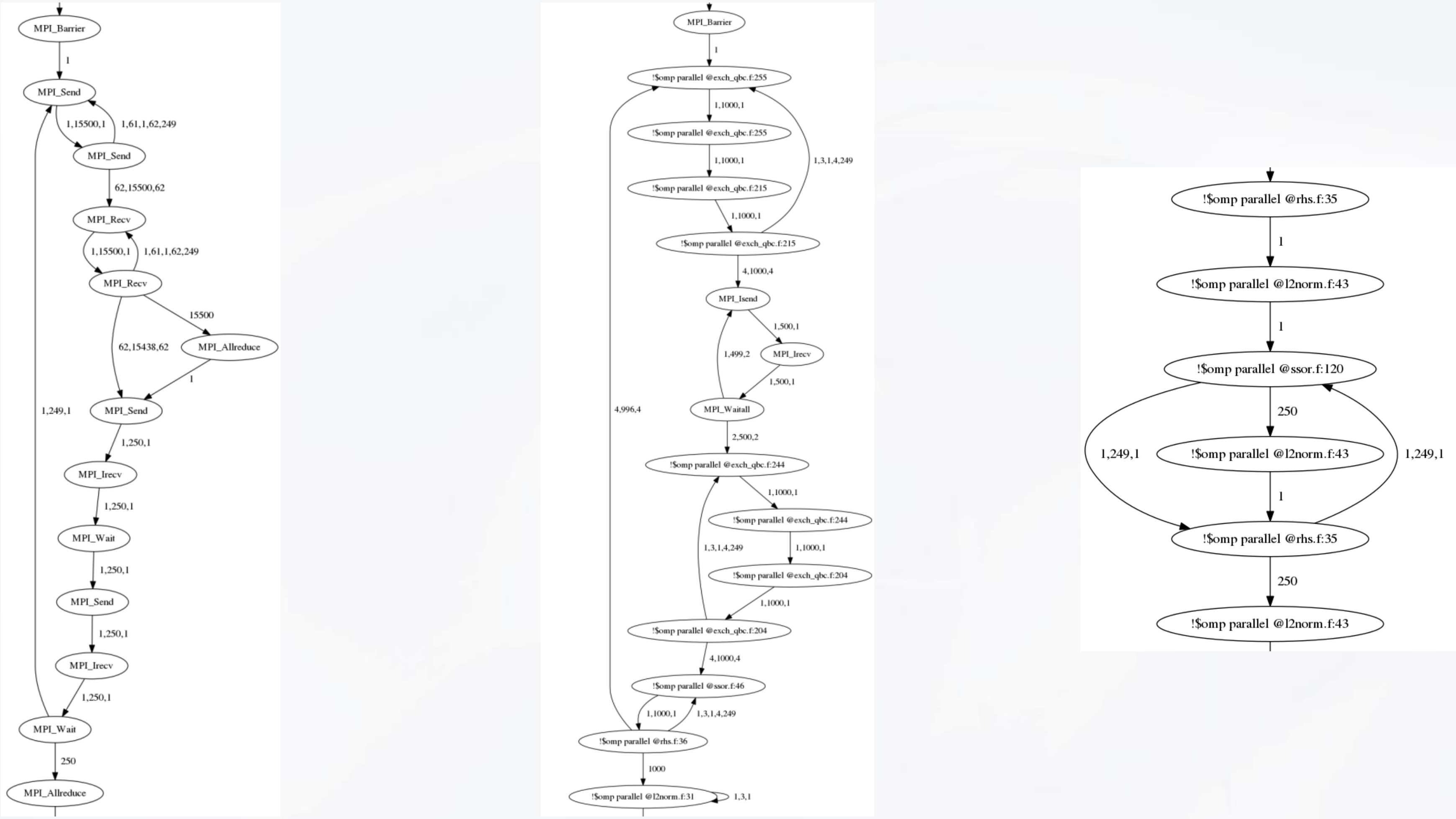
SPEC COSMO FD4 on 4 nodes / 96 ranks, MPI instrumentation, DVFS load balancing, HDEEM

Event Flow Graphs

- Record exclusive functions, no nested calls
- E.g., OpenMP parallel regions
- Node = parallel region with specific stack state
- Edge = possible successor
- Node label: region name
- Edge label:
  - No label: Only transition from node
  - Single number N: on Nth iteration
  - Three numbers i,j,k: transition is taken ith, i+kth,…jth execution of predecessor node
  - …


!$omp parallel @rhs.f:35
1
!$omp parallel @l2norm.f:43
1
!$omp parallel @ssor.f:120
250
1,249,1
!$omp parallel @l2norm.f:43
1,249,1
1
!$omp parallel @rhs.f:35
250
!$omp parallel @l2norm.f:43

Event flow graphs (NPB lu CLASS A, core loop)



(a) MPI (rank 0)

(b) MZ (rank 0, thread 0)

(c) OMP (thread 0)

Summary and Outlook

- Score-P as a common infrastructure for measurement, tuning, and debugging?
- Metric Plugins provide additional status information
- Substrate Plugins make use of Score-P's instrumentation frameworks and metrics to enhance functionality

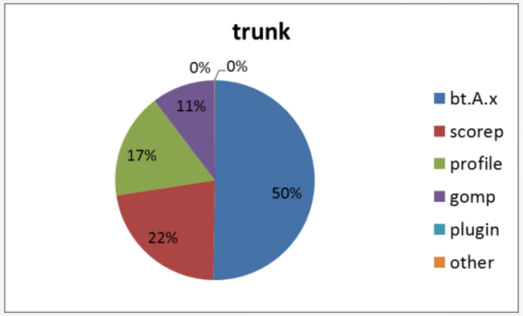
Substrate Plugin Interface **not final**. Any suggestions?

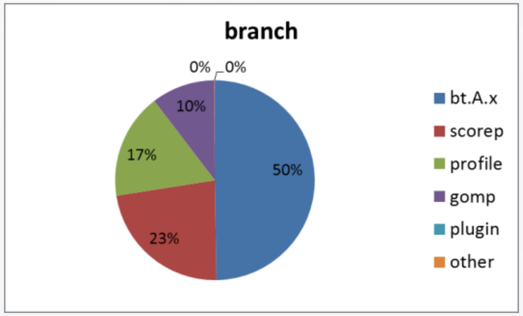
Talk to me in the next coffee break or write an email!

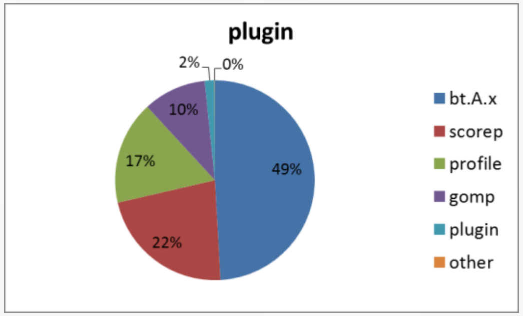
- [RLdS+09] *Barry Rountree, David K. Lownenthal, Bronis R. de Supinski, Martin Schulz, Vincent W. Freeh, and Tyler Bletsch*. Adagio: Making DVS Practical for Complex HPC Applications. In Proceedings of the 23rd international conference on Supercomputing, pages 460–469. ACM, 2009. DOI: 10.1145/1542275.1542340.
- [STHI11] *Robert Schöne, Ronny Tschüter, Daniel Hackenberg, and Thomas Ilsche*. The VampirTrace Plugin Counter Interface: Introduction and Examples. In Euro-Par 2010 Parallel Processing Workshops, volume 6586 of Lecture Notes in Computer Science, pages 501–511. Springer-Verlag, 2011. DOI: 10.1007/978-3-642-21878-1_62.
- [HIS+13] *Daniel Hackenberg, Thomas Ilsche, Robert Schone, Daniel Molka, Martin Schmidt, and Wolfgang E. Nagel*. Power Measurement Techniques on Standard Compute Nodes: A Quantitative Comparison. In Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on, pages 194–204. IEEE, 2013.
- [SM13] *Robert Schöne and Daniel Molka*. Integrating Performance Analysis and Energy Efficiency, Optimizations in a Unified Environment. Computer Science - Research and Development, pages 1–9, 2013. DOI: 10.1007/s00450-013-0243-7.
- [AFL14] *Xavier Aguilar, Karl Fürlinger, and Erwin Laure*. MPI Trace Compression Using Event Flow Graphs. In Euro-Par 2014 Parallel Processing, pages 1--12. Springer International Publishing, Cham, 2014. DOI: 10.1007/978-3-319-09873-9_1
- [HIS+14] *Daniel Hackenberg, Thomas Ilsche, Joseph Schuchart, Robert Schöne, Wolfgang E Nagel, Marc Simon, and Yiannis Georgiou*. HDEEM: High Definition Energy Efficiency Monitoring. In Energy Efficient Supercomputing Workshop (E2SC), 2014, pages 1–10. IEEE, 2014. DOI: 10.1109/E2SC.2014.13.

Backup Overhead Plugin

- Compare trunk, branch, branch with minimal registered plugin
- NPB OpenMP, BT Class A, 24 threads on Haswell dual-socket system
- many short regions, high overhead
- 5 measurements, use mean
- Average time increased by 1.5% if plugin registered, otherwise 0 overhead
- Sampling using perf record







Horizon 2020
European Union funding
for Research & Innovation