

Performance Engineering using MVAPICH and TAU

Sameer Shende
University of Oregon

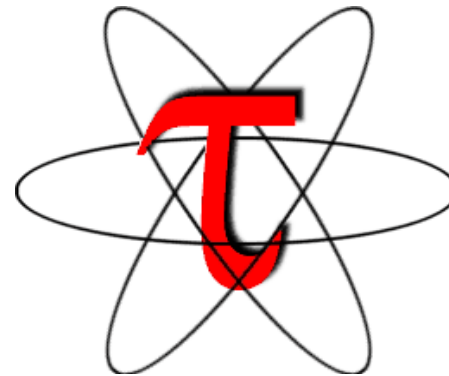
SC18 Talk
The Ohio State University Booth (#4404)
Wednesday, November 14, 2018, 2pm – 2:30pm
http://tau.uoregon.edu/tau_osu_sc18.pdf

Outline

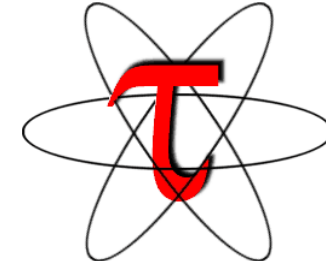
- **Introduction**
- **The MPI Tools Interfaces and Benefits**
- **Integrating TAU and MVAPICH2 with MPI_T**
- **Use Cases**
- **TAU Performance System[®]**

Acknowledgments

- **The MVAPICH2 team The Ohio State University**
 - <http://mvapich.cse.ohio-state.edu>
- **TAU team at the University of Oregon**
 - <http://tau.uoregon.edu>



TAU Performance System[®]



- **Tuning and Analysis Utilities (22+ year project)**
- **Comprehensive performance profiling and tracing**
 - Integrated, scalable, flexible, portable
 - Targets all parallel programming/execution paradigms
- **Integrated performance toolkit**
 - Instrumentation, measurement, analysis, visualization
 - Widely-ported performance profiling / tracing system
 - Performance data management and data mining
 - Open source (BSD-style license)
 - Uses performance and control variables to interface with MVAPICH2
- **Integrates with application frameworks**
- **<http://tau.uoregon.edu>**

Understanding Application Performance using TAU

- **How much time** is spent in each application routine and outer *loops*? Within loops, what is the contribution of each *statement*?
- **How many instructions** are executed in these code regions?
Floating point, Level 1 and 2 *data cache misses*, hits, branches taken?
- **What is the memory usage** of the code? When and where is memory allocated/de-allocated? Are there any memory leaks?
- **What are the I/O characteristics** of the code? What is the peak read and write *bandwidth* of individual calls, total volume?
- **What is the contribution of each phase** of the program? What is the time wasted/spent waiting for collectives, and I/O operations in Initialization, Computation, I/O phases?
- **How does the application scale?** What is the efficiency, runtime breakdown of performance across different core counts?
- **How can I tune MPI for better performance?** What performance and control does MVAPICH2 export to observe and control its performance?

Overview of the MVAPICH2 Project

High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)

- MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.1), Started in 2001, First version available in 2002
- MVAPICH2-X (MPI + PGAS), Available since 2011
- Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
- Support for Virtualization (MVAPICH2-Virt), Available since 2015
- Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
- Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
- **Used by more than 2,925 organizations in 86 countries**
- **More than 481,000 (> 0.48 million) downloads from the OSU site directly**
- Empowering many TOP500 clusters (Jun '18 ranking)
 - 2nd, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
 - 12th, 556,104 cores (Oakforest-PACS) in Japan
 - 15th, 367,024 cores (Stampede2) at TACC
 - 24th, 241,108-core (Pleiades) at NASA
 - 62nd, 76,032-core (Tsubame 2.5) at Tokyo Institute of Technology
- Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
- <http://mvapich.cse.ohio-state.edu>

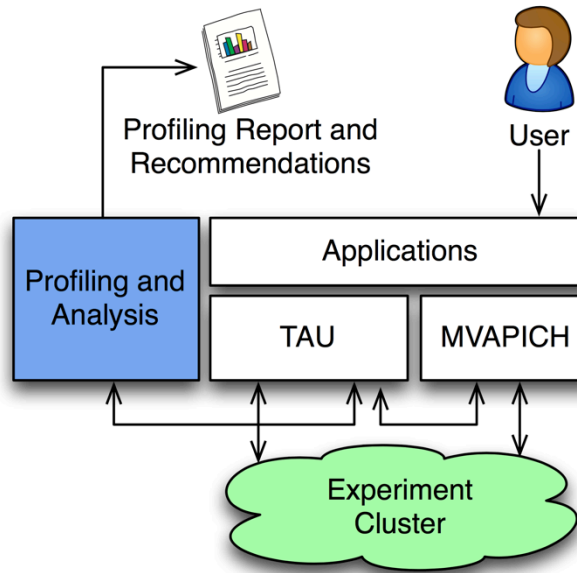


Empowering Top500 systems for over a decade

Outline

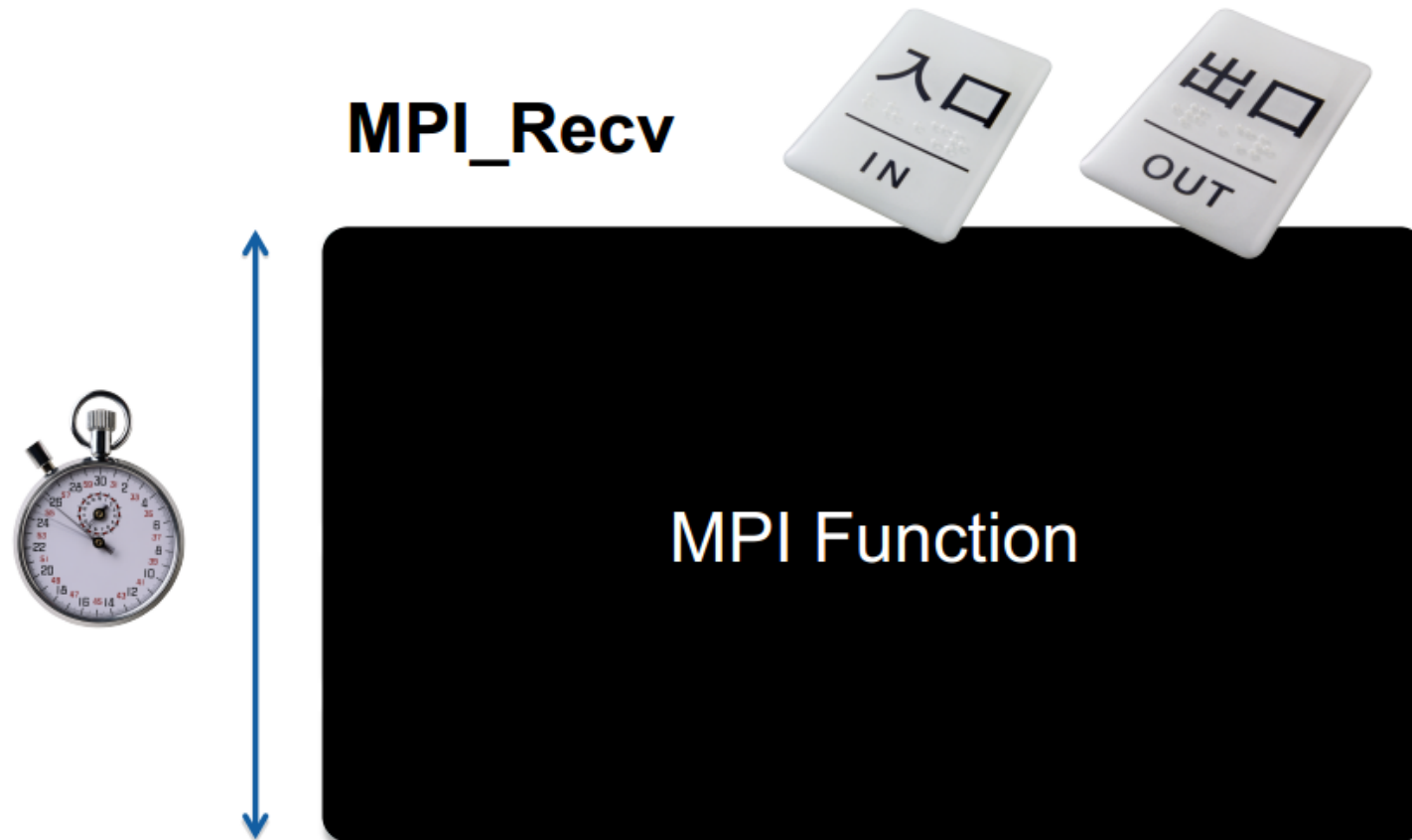
- Introduction
- **The MPI Tools Interfaces and Benefits**
- Integrating TAU and MVAPICH2 with MPI_T
- Use Cases
- TAU Performance System[®]

MVAPICH2 and TAU



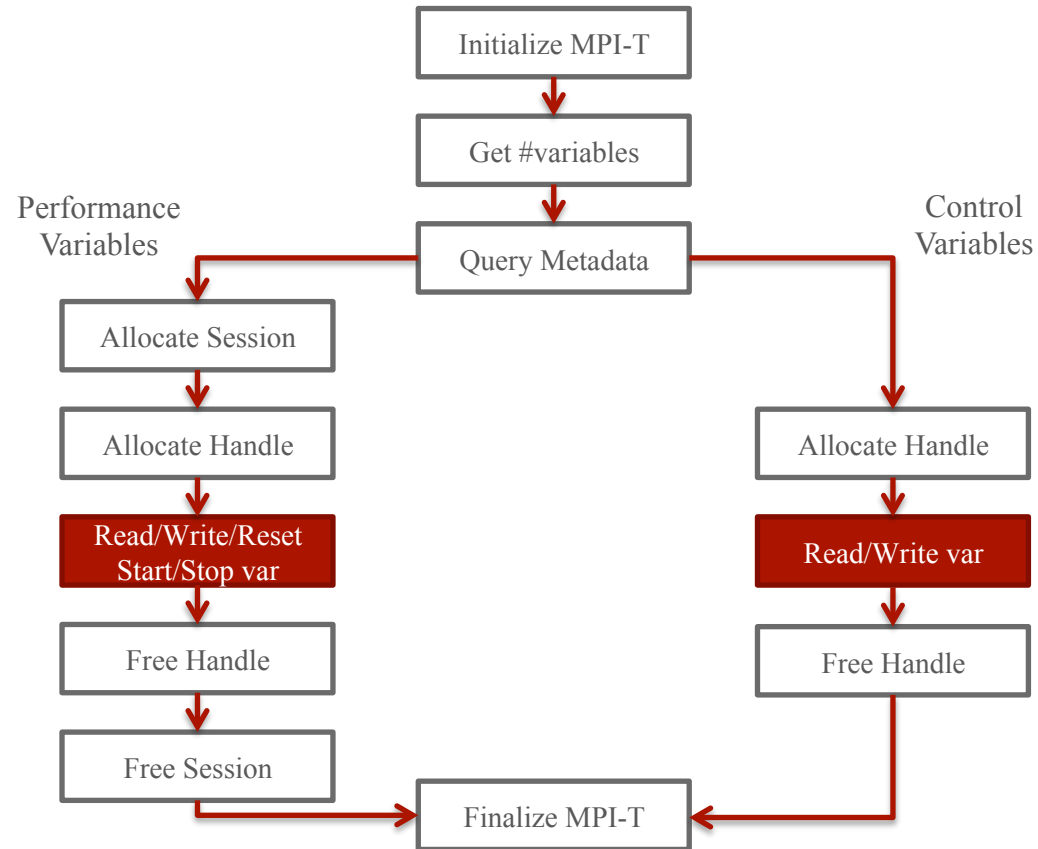
- TAU and MVAPICH2 are enhanced with the ability to generate recommendations and engineering performance report
- MPI libraries like MVAPICH2 are now “reconfigurable” at runtime
- TAU and MVAPICH2 communicate using the MPI-T interface

Why PMPI is not good enough?



- Takes a “black box” view of the MPI library

MPI_T usage semantics



```

int MPI_T_cvar_get_info(int cvar_index, char *name, int *name_len, int *verbosity,
int MPI_T_pvar_start(MPI_T_pvar_session session, MPI_T_pvar_handle handle)
int MPI_T_pvar_handle_alloc(MPI_T_pvar_session session, int pvar_index,
int MPI_T_pvar_handle_alloc(MPI_T_pvar_session session, MPI_T_pvar_handle handle, int *count);
int MPI_T_pvar_reset(MPI_T_pvar_session session, MPI_T_pvar_handle handle,
char *desc, int *desc_len, int *bind, int *scope);
  
```

MPI_T support with MVAPICH2

- Support performance variables (PVAR)
 - Variables to track different components within the MPI library
- Initial support for Control Variables (CVAR)
 - Variables to modify the behavior of MPI Library

Memory Usage:

- current level
- maximum watermark

InfiniBand N/W:

- #control packets
- #out-of-order packets

Pt-to-pt messages:

- unexpected queue length
- unexp. match attempts
- recvq. length

Registration cache:

- hits
- misses

Shared-memory:

- limic/ CMA
- buffer pool size & usage

Collective ops:

- comm. creation
- #algorithm invocations
- [Bcast – 8; Gather – 10]

Co-designing Applications to use MPI-T

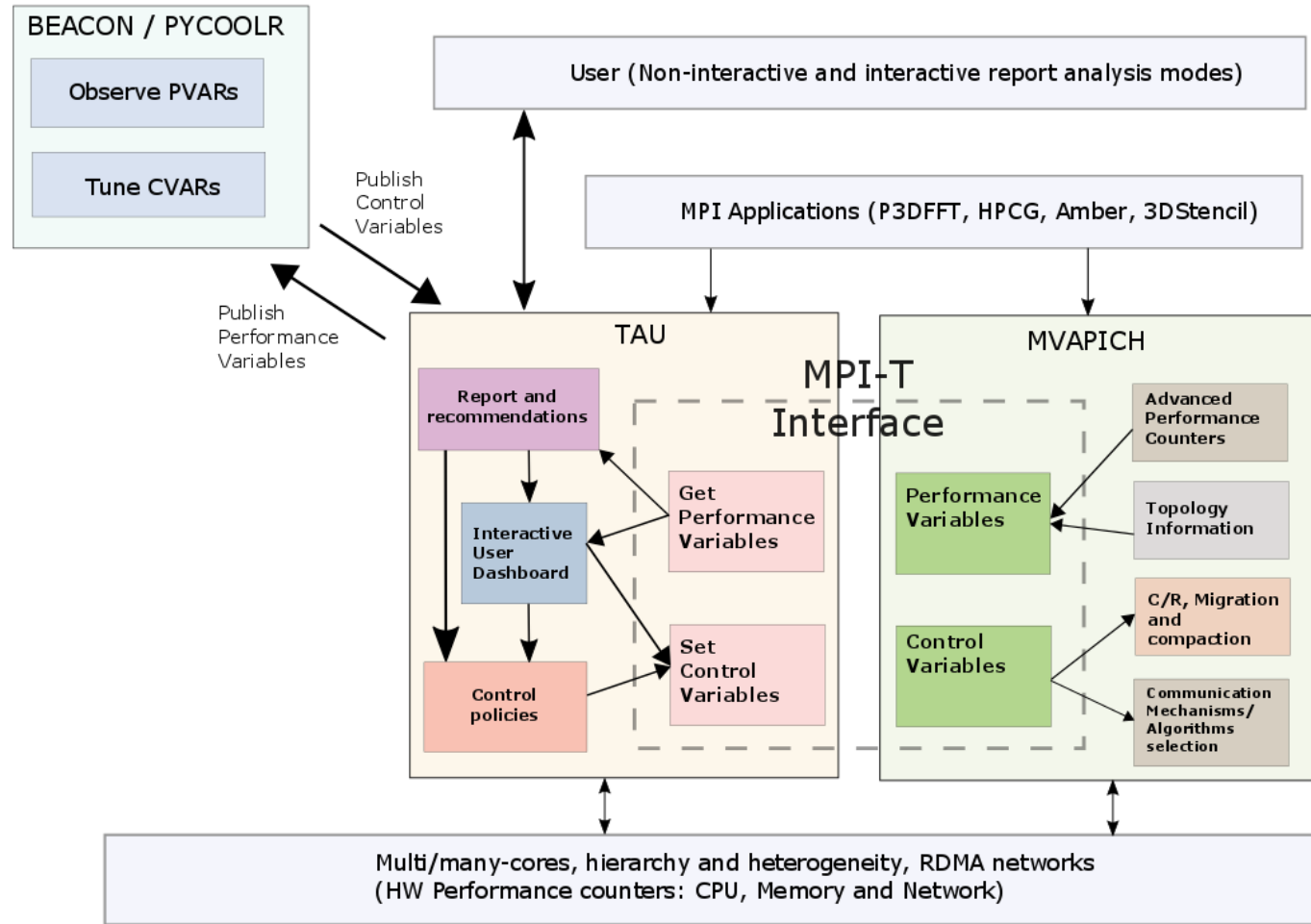
Example Pseudo-code: Optimizing the eager limit dynamically:

```
MPI_T_init_thread(..)  
MPI_T_cvar_get_info(MV2_EAGER_THRESHOLD)  
if (msg_size < MV2_EAGER_THRESHOLD + 1KB)  
    MPI_T_cvar_write(MV2_EAGER_THRESHOLD, +1024)  
MPI_Send(..)  
MPI_T_finalize(..)
```

Outline

- Introduction
- The MPI Tools Interfaces and Benefits
- **Integrating TAU and MVAPICH2 with MPI_T**
- **Use Cases**
- **TAU Performance System[®]**

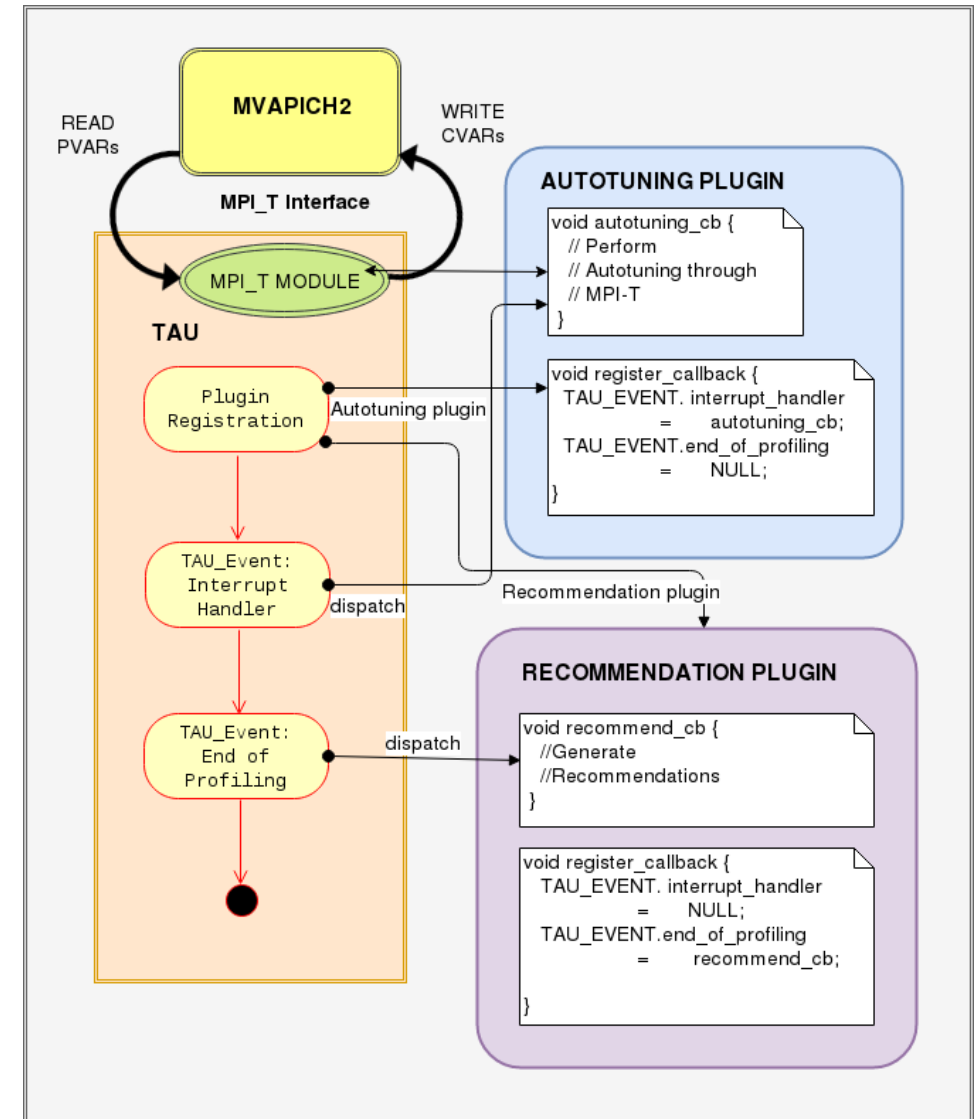
Interacting TAU with MVAPICH2 through MPI_T Interface



- Enhance existing support for MPI_T in MVAPICH2 to expose a richer set of performance and control variables
- Get and display MPI Performance Variables (PVARs) made available by the runtime in TAU
- Control the runtime's behavior via MPI Control Variables (CVARs)
- Add support to MVAPICH2 and TAU for interactive performance engineering sessions

Plugin-based Infrastructure for Non-Interactive Tuning

- Performance data collected by TAU
 - Support for PVARs and CVARs
 - Setting CVARs to control MVAPICH2
 - Studying performance data in TAU's ParaProf profile browser
 - Multiple plugins available for
 - Tuning application at runtime and
 - Generate post-run recommendations



Enhancing MPI_T Support

- **Introduced support for new MPI_T based CVARs to MVAPICH2**
 - `MPIR_CVAR_MAX_INLINE_MSG_SZ`
 - Controls the message size up to which “inline” transmission of data is supported by MVAPICH2
 - `MPIR_CVAR_VBUF_POOL_SIZE`
 - Controls the number of internal communication buffers (VBUFs) MVAPICH2 allocates initially. Also, `MPIR_CVAR_VBUF_POOL_REDUCED_VALUE[1] ([2...n])`
 - `MPIR_CVAR_VBUF_SECONDARY_POOL_SIZE`
 - Controls the number of VBUFs MVAPICH2 allocates when there are no more free VBUFs available
 - `MPIR_CVAR_IBA_EAGER_THRESHOLD`
 - Controls the message size where MVAPICH2 switches from eager to rendezvous protocol for large messages
- **TAU enhanced with support for setting MPI_T CVARs in a non-interactive mode for uninstrumented applications**

MVAPICH2

- **Several new MPI_T based PVARs added to MVAPICH2**
 - mv2_vbuf_max_use, mv2_total_vbuf_memory etc
- **Enhanced TAU with support for tracking of MPI_T PVARs and CVARs for uninstrumented applications**
 - ParaProf, TAU's visualization front end, enhanced with support for displaying PVARs and CVARs
 - TAU provides tau_exec, a tool to transparently instrument MPI routines
 - Uninstrumented:
% mpirun -np 1024 ./a.out
 - Instrumented:
% mpirun -np 1024 tau_exec [options] ./a.out
% paraprof

PVARs Exposed by MVAPICH2

TAU: ParaProf Manager

| TrialField | Value |
|--|--|
| MPI_T PVAR[0]: mem_allocated | Current level of allocated memory within the MPI library |
| MPI_T PVAR[10]: mv2_num_2level_comm_success | Number of successful 2-level comm creations |
| MPI_T PVAR[11]: mv2_num_shmem_coll_calls | Number of times MV2 shared-memory collective calls were invoked |
| MPI_T PVAR[12]: mpit_progress_poll | CH3 RDMA progress engine polling count |
| MPI_T PVAR[13]: mv2_smp_read_progress_poll | CH3 SMP read progress engine polling count |
| MPI_T PVAR[14]: mv2_smp_write_progress_poll | CH3 SMP write progress engine polling count |
| MPI_T PVAR[15]: mv2_smp_read_progress_poll_success | Unsuccessful CH3 SMP read progress engine polling count |
| MPI_T PVAR[16]: mv2_smp_write_progress_poll_succ... | Unsuccessful CH3 SMP write progress engine polling count |
| MPI_T PVAR[17]: rdma_ud_retransmissions | CH3 RDMA UD retransmission count |
| MPI_T PVAR[18]: mv2_coll_bcast_binomial | Number of times MV2 binomial bcast algorithm was invoked |
| MPI_T PVAR[19]: mv2_coll_bcast_scatter_doubling_all... | Number of times MV2 scatter+double allgather bcast algorithm was invoked |
| MPI_T PVAR[1]: mem_allocated | Maximum level of memory ever allocated within the MPI library |
| MPI_T PVAR[20]: mv2_coll_bcast_scatter_ring_allgather | Number of times MV2 scatter+ring allgather bcast algorithm was invoked |
| MPI_T PVAR[21]: mv2_coll_bcast_scatter_ring_allgath... | Number of times MV2 scatter+ring allgather shm bcast algorithm was invoked |
| MPI_T PVAR[22]: mv2_coll_bcast_shmem | Number of times MV2 shm bcast algorithm was invoked |
| MPI_T PVAR[23]: mv2_coll_bcast_knomial_intranode | Number of times MV2 knomial intranode bcast algorithm was invoked |
| MPI_T PVAR[24]: mv2_coll_bcast_knomial_intranode | Number of times MV2 knomial intranode bcast algorithm was invoked |
| MPI_T PVAR[25]: mv2_coll_bcast_mcast_intranode | Number of times MV2 mcast intranode bcast algorithm was invoked |
| MPI_T PVAR[26]: mv2_coll_bcast_pipelined | Number of times MV2 pipelined bcast algorithm was invoked |
| MPI_T PVAR[27]: mv2_coll_alltoall_inplace | Number of times MV2 in-place alltoall algorithm was invoked |
| MPI_T PVAR[28]: mv2_coll_alltoall_bruck | Number of times MV2 brucks alltoall algorithm was invoked |
| MPI_T PVAR[29]: mv2_coll_alltoall_rd | Number of times MV2 recursive-doubling alltoall algorithm was invoked |
| MPI_T PVAR[2]: num_malloc_calls | Number of MPIT_malloc calls |
| MPI_T PVAR[30]: mv2_coll_alltoall_sd | Number of times MV2 scatter-destination alltoall algorithm was invoked |
| MPI_T PVAR[31]: mv2_coll_alltoall_pw | Number of times MV2 pairwise alltoall algorithm was invoked |
| MPI_T PVAR[32]: mpit_alltoall_mv2_pw | Number of times MV2 pairwise alltoallv algorithm was invoked |
| MPI_T PVAR[33]: mv2_coll_allreduce_shm_rd | Number of times MV2 shm rd allreduce algorithm was invoked |
| MPI_T PVAR[34]: mv2_coll_allreduce_shm_rs | Number of times MV2 shm rs allreduce algorithm was invoked |
| MPI_T PVAR[35]: mv2_coll_allreduce_shm_intra | Number of times MV2 shm intra allreduce algorithm was invoked |
| MPI_T PVAR[36]: mv2_coll_allreduce_intra_p2p | Number of times MV2 intra p2p allreduce algorithm was invoked |
| MPI_T PVAR[37]: mv2_coll_allreduce_2lvl | Number of times MV2 two-level allreduce algorithm was invoked |
| MPI_T PVAR[38]: mv2_coll_allreduce_shmem | Number of times MV2 shm allreduce algorithm was invoked |
| MPI_T PVAR[39]: mv2_coll_allreduce_mcast | Number of times MV2 multicast-based allreduce algorithm was invoked |
| MPI_T PVAR[3]: num_malloc_calls | Number of MPIT_malloc calls |
| MPI_T PVAR[40]: mv2_reg_cache_hits | Number of registration cache hits |
| MPI_T PVAR[41]: mv2_reg_cache_misses | Number of registration cache misses |
| MPI_T PVAR[42]: mv2_vbuf_allocated | Number of VBUFs allocated |
| MPI_T PVAR[43]: mv2_vbuf_allocated_array | Number of VBUFs allocated |
| MPI_T PVAR[44]: mv2_vbuf_freed | Number of VBUFs freed |
| MPI_T PVAR[45]: mv2_ud_vbuf_allocated | Number of UD VBUFs allocated |
| MPI_T PVAR[46]: mv2_ud_vbuf_freed | Number of UD VBUFs freed |
| MPI_T PVAR[47]: mv2_vbuf_free_attempts | Number of time we attempted to free VBUFs |
| MPI_T PVAR[48]: mv2_vbuf_free_attempt_success_time | Average time for number of times we successfully freed VBUFs |
| MPI_T PVAR[49]: mv2_vbuf_free_attempt_success_time | Average time for number of times we successfully freed VBUFs |
| MPI_T PVAR[4]: num_memalign_calls | Number of MPIT_memalign calls |
| MPI_T PVAR[50]: mv2_vbuf_allocate_time | Average time for number of times we allocated VBUFs |
| MPI_T PVAR[51]: mv2_vbuf_allocate_time | Average time for number of times we allocated VBUFs |

CVARs Exposed by MVAPICH2

| TrialField | Value |
|--|--|
| Local Time | 2016-08-16T10:11:04-07:00 |
| MPI Processor Name | cerberus.nic.uoregon.edu |
| MPIR_CVAR_ABORT_ON_LEAKED_HANDLES | If true, MPI will call MPI_Abort at MPI_Finalize if any MPI object handles have been leaked. For example,... |
| MPIR_CVAR_ALLGATHERV_PIPELINE_MSG_SIZE | The smallest message size that will be used for the pipelined, large-message, ring algorithm in the MPI_... |
| MPIR_CVAR_ALLGATHER_LONG_MSG_SIZE | For MPI_Allgather and MPI_Allgatherv, the long message algorithm will be used if the send buffer size is ... |
| MPIR_CVAR_ALLGATHER_SHORT_MSG_SIZE | For MPI_Allgather and MPI_Allgatherv, the short message algorithm will be used if the send buffer size is... |
| MPIR_CVAR_ALLREDUCE_SHORT_MSG_SIZE | the short message algorithm will be used if the send buffer size is <= this value (in bytes) |
| MPIR_CVAR_ALLTOALL_MEDIUM_MSG_SIZE | the medium message algorithm will be used if the per-destination message size (sendcount*size(sendtyp... |
| MPIR_CVAR_ALLTOALL_SHORT_MSG_SIZE | the short message algorithm will be used if the per-destination message size (sendcount*size(sendtype)) ... |
| MPIR_CVAR_ALLTOALL_THROTTLE | max no. of irecv/sends posted at a time in some alltoall algorithms. Setting it to 0 causes all irecv/sen... |
| MPIR_CVAR_ASYNC_PROGRESS | If set to true, MPICH will initiate an additional thread to make asynchronous progress on all communicati... |
| MPIR_CVAR_BCAST_LONG_MSG_SIZE | Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and mediu... |
| MPIR_CVAR_BCAST_MIN_PROCS | Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and mediu... |
| MPIR_CVAR_BCAST_SHORT_MSG_SIZE | Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and mediu... |
| MPIR_CVAR_CH3_EAGER_MAX_MSG_SIZE | This cvar controls the message size at which CH3 switches from eager to rendezvous mode. |
| MPIR_CVAR_CH3_ENABLE_HCOLL | If true, enable HCOLL collectives. |
| MPIR_CVAR_CH3_INTERFACE_HOSTNAME | If non-NULL, this cvar specifies the IP address that other processes should use when connecting to this pr... |
| MPIR_CVAR_CH3_NOLOCAL | If true, force all processes to operate as though all processes are located on another node. For example,... |
| MPIR_CVAR_CH3_ODD_EVEN_CLIQUES | If true, odd procs on a node are seen as local to each other, and even procs on a node are seen as local t... |
| MPIR_CVAR_CH3_PORT_RANGE | The MPIR_CVAR_CH3_PORT_RANGE environment variable allows you to specify the range of TCP ports ... |
| MPIR_CVAR_CH3_RMA_ACC_IMMED | Use the immediate accumulate optimization |
| MPIR_CVAR_CH3_RMA_GC_NUM_COMPLETED | Threshold for the number of completed requests the runtime finds before it stops trying to find more co... |
| MPIR_CVAR_CH3_RMA_GC_NUM_TESTED | Threshold for the number of RMA requests the runtime tests before it stops trying to check more reques... |
| MPIR_CVAR_CH3_RMA_LOCK_IMMED | Issue a request for the passive target RMA lock immediately. Default behavior is to defer the lock requ... |
| MPIR_CVAR_CH3_RMA_MERGE_LOCK_OP_UNLOCK | Enable/disable an optimization that merges lock, op, and unlock messages, for single-operation passive ta... |
| MPIR_CVAR_CH3_RMA_NREQUEST_NEW_THRESHOLD | Threshold for the number of new requests since the last attempt to complete pending requests. Higher ... |
| MPIR_CVAR_CH3_RMA_NREQUEST_THRESHOLD | Threshold at which the RMA implementation attempts to complete requests while completing RMA oper... |
| MPIR_CVAR_CHOP_ERROR_STACK | If >0, truncate error stack output lines this many characters wide. If 0, do not truncate, and if <0 use a ... |
| MPIR_CVAR_COLL_ALIAS_CHECK | Enable checking of aliasing in collective operations |
| MPIR_CVAR_COMM_SPLIT_USE_QSORT | Use qsort(3) in the implementation of MPI_Comm_split instead of bubble sort. |
| MPIR_CVAR_CTXID_EAGER_SIZE | The MPIR_CVAR_CTXID_EAGER_SIZE environment variable allows you to specify how many words in th... |
| MPIR_CVAR_DEBUG_HOLD | If true, causes processes to wait in MPI_Init and MPI_Initthread for a debugger to be attached. Once the ... |
| MPIR_CVAR_DEFAULT_THREAD_LEVEL | Sets the default thread level to use when using MPI_INIT. |
| MPIR_CVAR_DUMP_PROVIDERS | If true, dump provider information at init |
| MPIR_CVAR_ENABLE_COLL_FT_RET | DEPRECATED! Will be removed in MPICH-3.2 Collectives called on a communicator with a failed process... |
| MPIR_CVAR_ENABLE_SMP_ALLREDUCE | Enable SMP aware allreduce. |
| MPIR_CVAR_ENABLE_SMP_BARRIER | Enable SMP aware barrier. |
| MPIR_CVAR_ENABLE_SMP_BCAST | Enable SMP aware broadcast (See also: MPIR_CVAR_MAX_SMP_BCAST_MSG_SIZE) |
| MPIR_CVAR_ENABLE_SMP_COLLECTIVES | Enable SMP aware collective communication. |
| MPIR_CVAR_ENABLE_SMP_REDUCE | Enable SMP aware reduce. |
| MPIR_CVAR_ERROR_CHECKING | If true, perform checks for errors, typically to verify valid inputs to MPI routines. Only effective when M... |
| MPIR_CVAR_GATHERV_INTER_SSEND_MIN_PROCS | Use Ssend (synchronous send) for intercommunicator MPI_Gatherv if the "group B" size is >= this value.... |
| MPIR_CVAR_GATHER_INTER_SHORT_MSG_SIZE | use the short message algorithm for intercommunicator MPI_Gather if the send buffer size is < this value... |
| MPIR_CVAR_GATHER_VSMALL_MSG_SIZE | use a temporary buffer for intracommunicator MPI_Gather if the send buffer size is < this value (in bytes... |
| MPIR_CVAR_IBA_EAGER_THRESHOLD | 0 (old) -> 204800 (new), This set the switch point between eager and rendezvous protocol |
| MPIR_CVAR_MAX_INLINE_SIZE | This set the maximum inline size for data transfer |
| MPIR_CVAR_MAX_SMP_ALLREDUCE_MSG_SIZE | Maximum message size for which SMP-aware allreduce is used. A value of '0' uses SMP-aware allreduce ... |

Outline

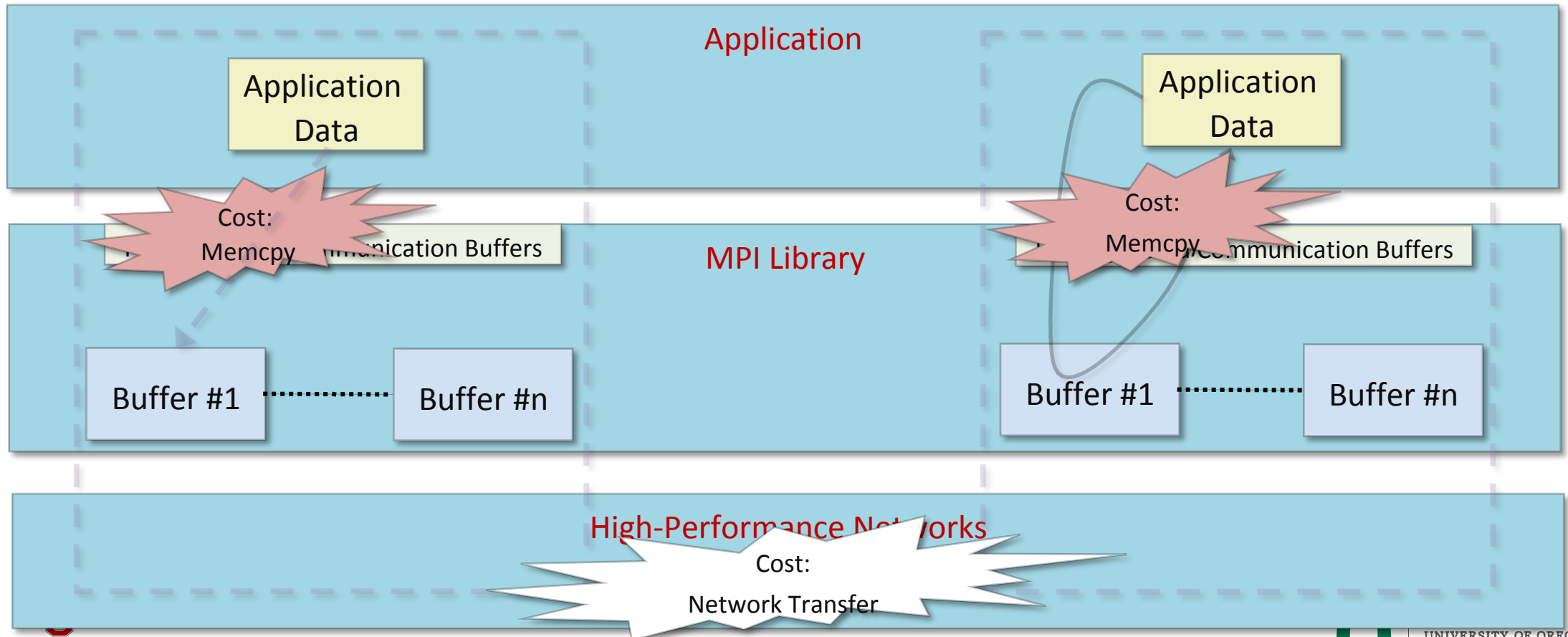
- Introduction
- The MPI Tools Interfaces and Benefits
- Integrating TAU and MVAPICH2 with MPI_T
- **Use Cases**
 - **Designing Dynamic and Adaptive MPI Point-to-point Protocols**
- **TAU Performance System[®]**

Point-to-point Communication Protocols in MPI

- **Eager Protocol**
 - Best communication performance for smaller messages
- **Rendezvous Protocol**
 - Best communication performance for larger messages

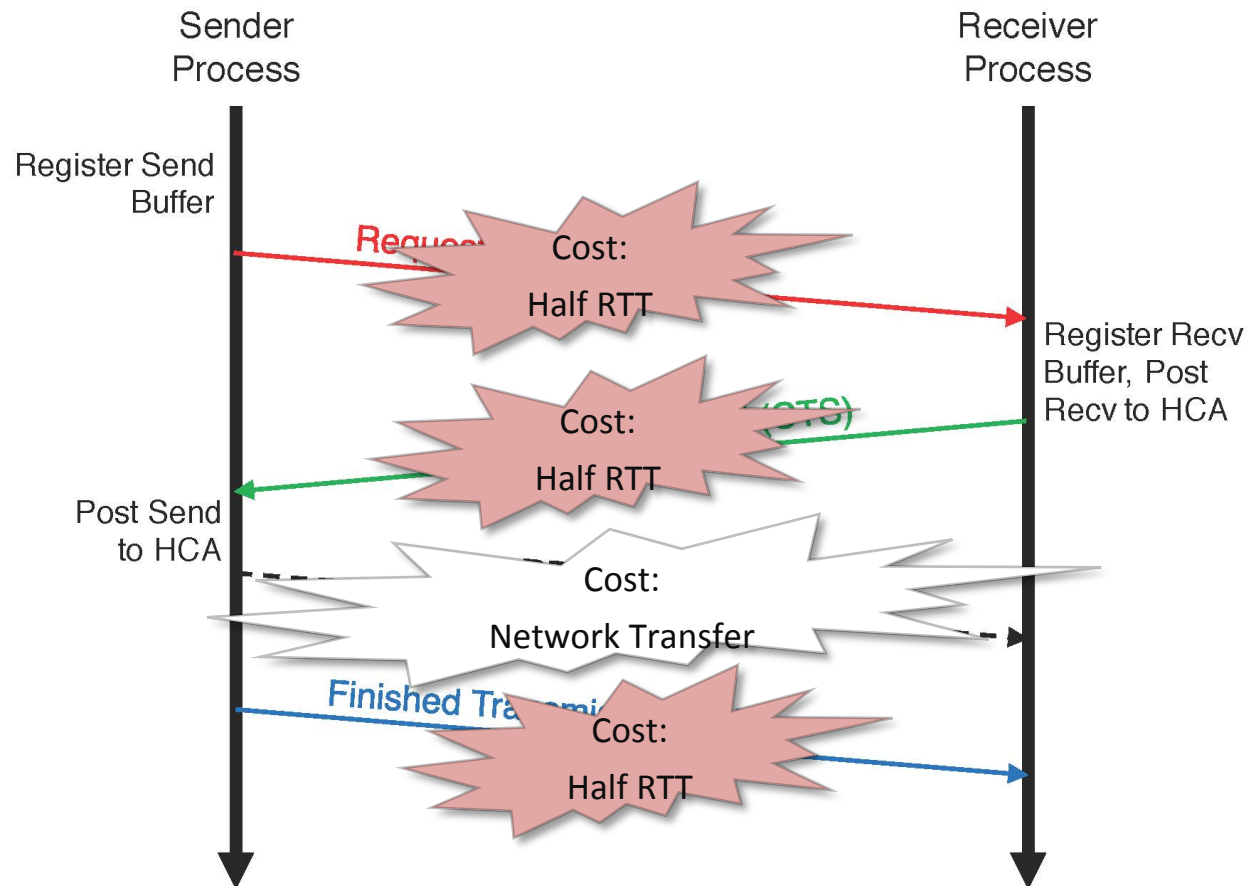
Analyzing Communication Costs of Point-to-point Protocols

- **Eager Protocol**
 - Best communication performance for smaller messages

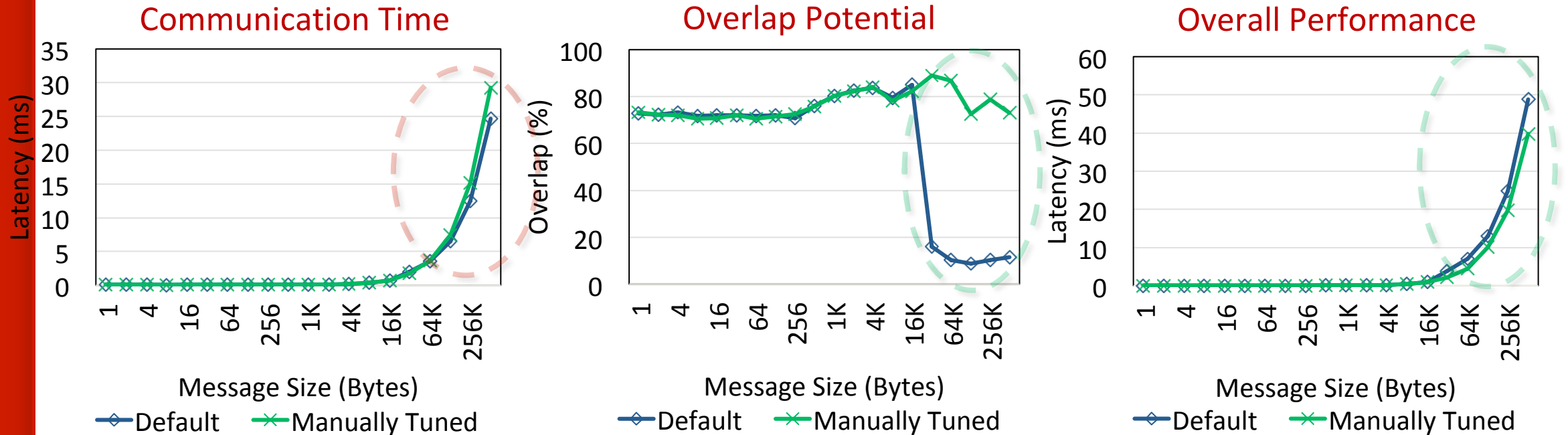


Analyzing Communication Costs of Point-to-point Protocols (Cont.)

- **Rendezvous Protocol**
 - Best communication performance for larger messages

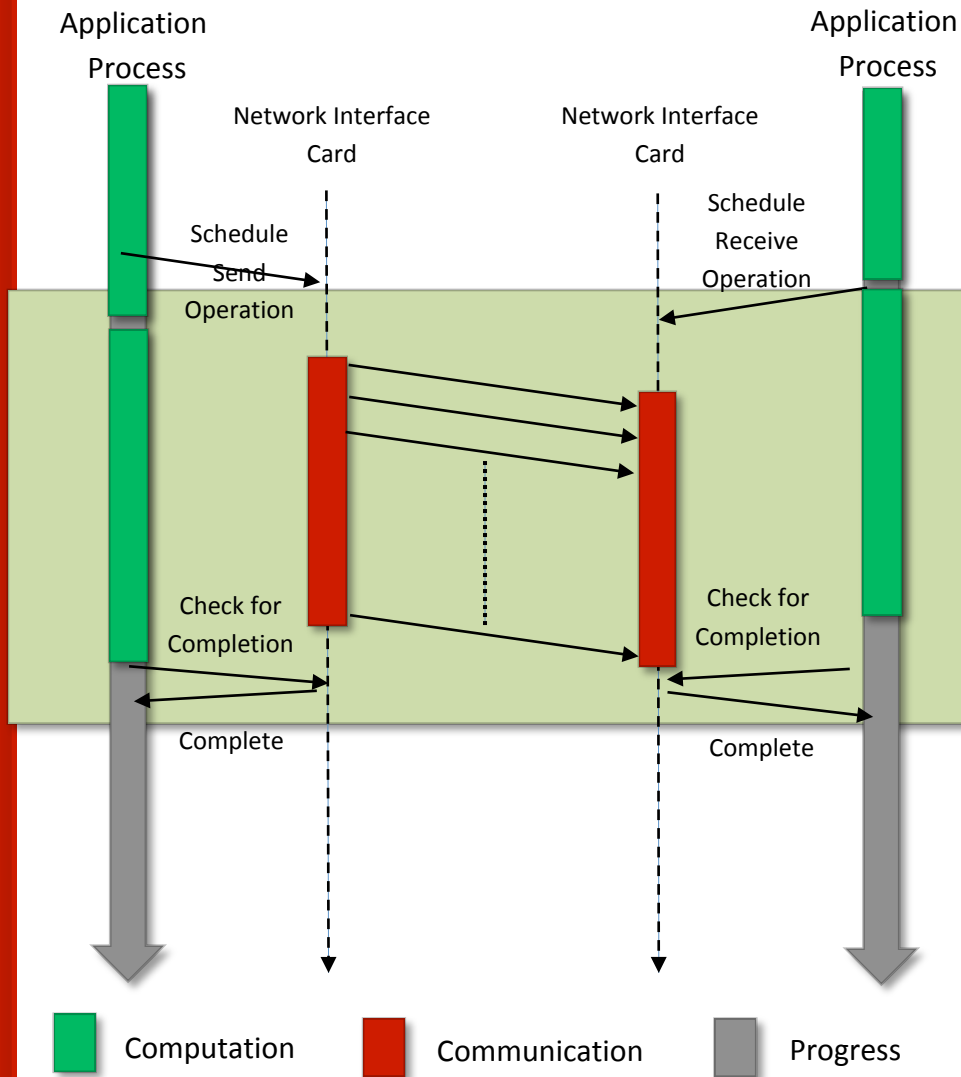


Studying the Performance and Overlap of 3D Stencil Benchmark

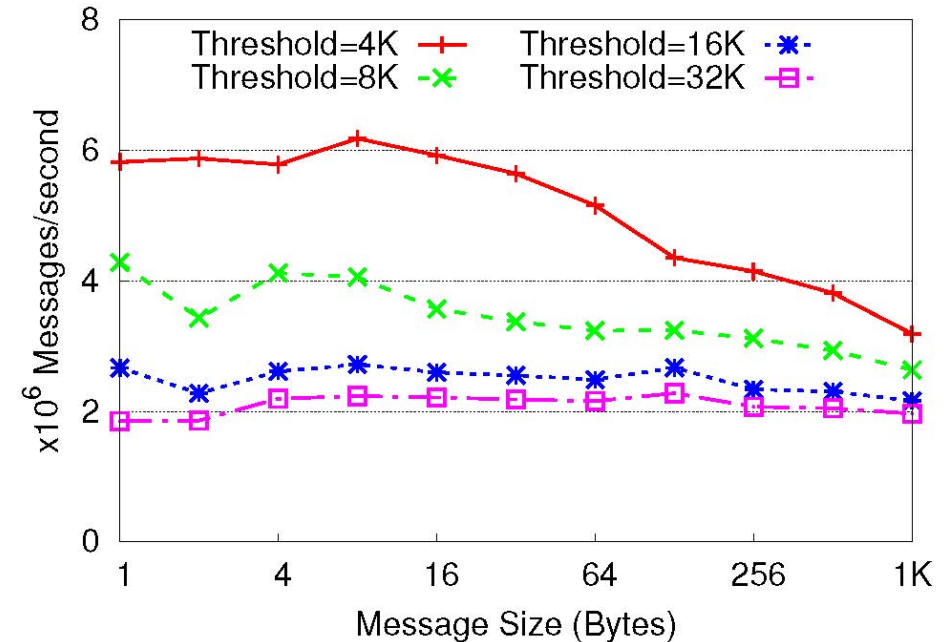


- **Default:** Uses eager protocol for small messages and rendezvous for large
- **Manually Tuned:** Forces the use of eager for all message sizes
- **Manually Tuned has degradation in raw communication performance**
- **Manually Tuned has significant benefits for overlap**
- **Manually Tuned better for overall application execution time**

Analyzing Overlap Potential of Eager Protocol

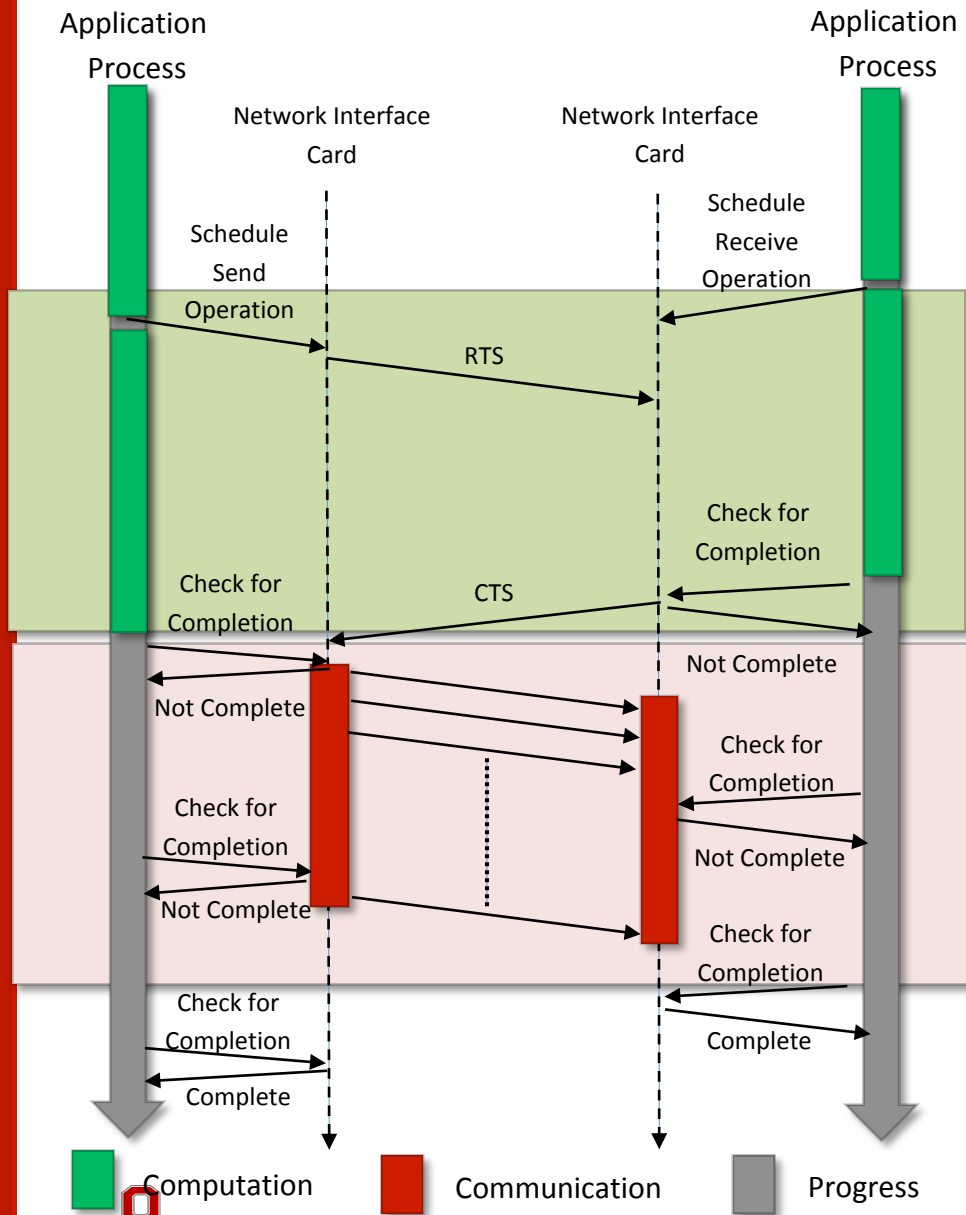


- Application processes schedule communication operation
- Network adapter progresses communication in the background
- Application process free to perform useful compute in the foreground
- **Overlap of computation and communication => Better Overall Application Performance**
- **Increased buffer requirement**
- **Poor communication performance if used for all types of communication operations**



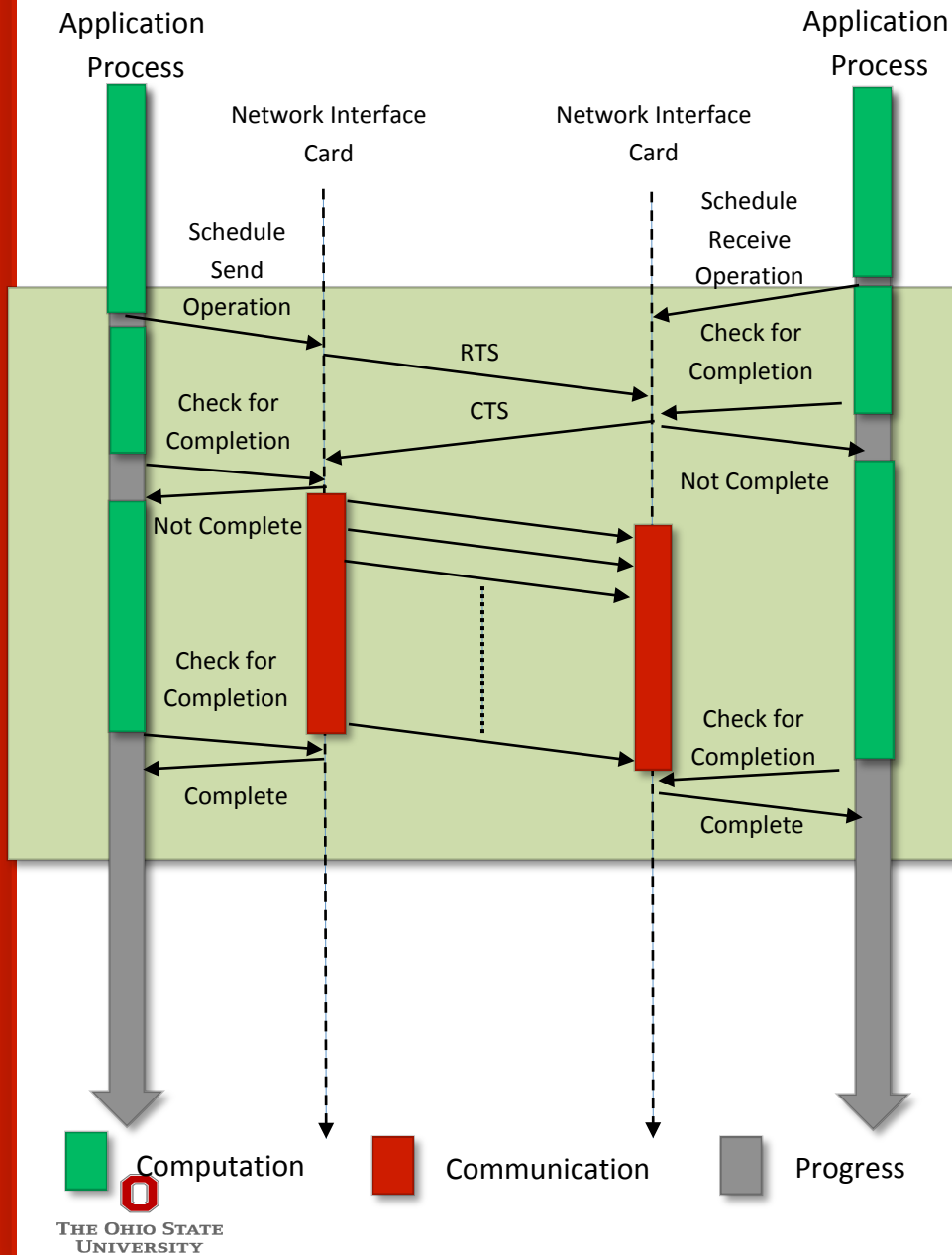
Impact of changing Eager Threshold on performance of multi-pair message-rate benchmark with 32 processes on Stampede

Analyzing Overlap Potential of Rendezvous Protocol



- Application processes schedule communication operation
- Application process free to perform useful compute in the foreground
- Little communication progress in the background
- All communication takes place at final synchronization
- Reduced buffer requirement
- Good communication performance if used for large message sizes and operations where communication library is progressed frequently
- Poor overlap of computation and communication => Poor Overall Application Performance

But... What if Applications Progress Communication Frequently?



- **Application processes schedule communication operation**
- **Application process free to perform useful compute in the foreground**
- **Overlap of computation and communication => Better Overall Application Performance**
- **Reduced buffer requirement**
- **Good communication performance as communication library is progressed frequently**
- **Harder to create such programs that progress communication at the exact time without causing overhead**
- **Communication support entities (threads, hardware engines, etc.) have their own complexities**

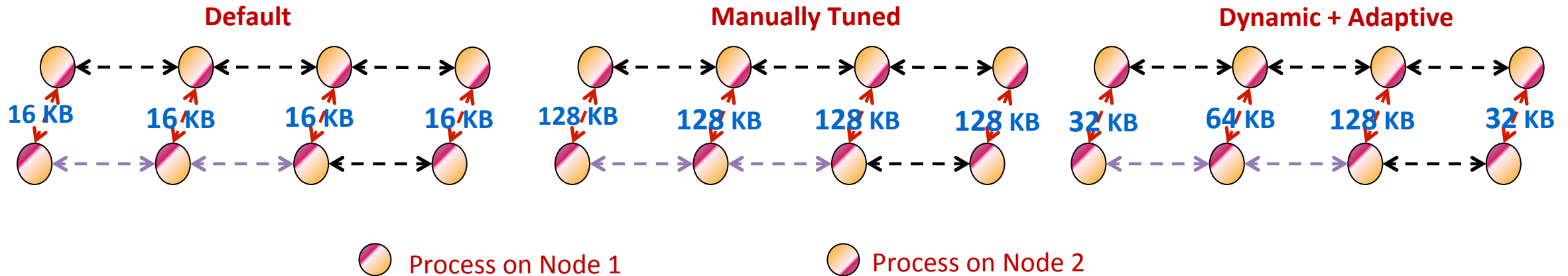
Broad Challenge

Can we design *dynamic and adaptive* point-to-point communication mechanisms that can deliver the best

1. *Communication performance*
2. *Overlap of computation and communication*
3. *Memory footprint*

Proposed Designs and Expected Benefits at a High-level

Eager Threshold for Example Communication Pattern with Different Designs



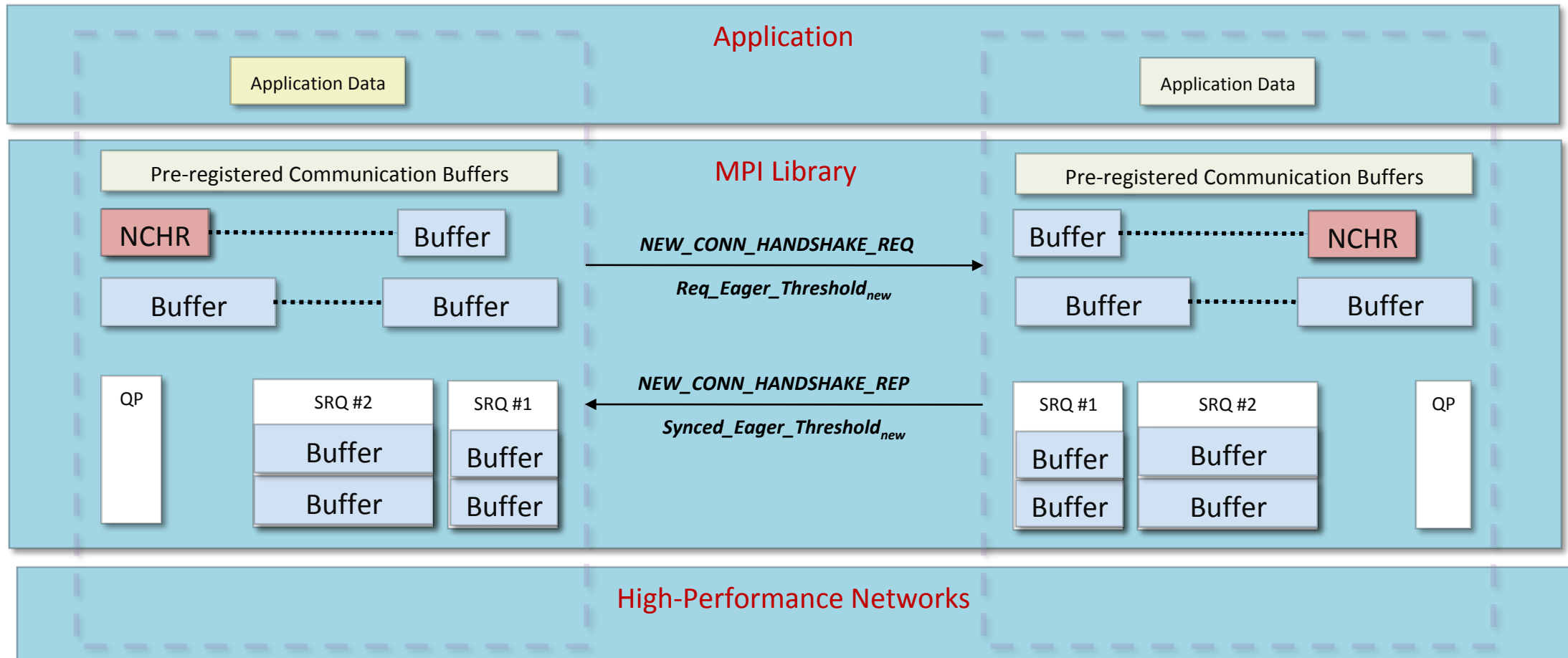
Desired Eager Threshold for Example Communication Pattern

| Process Pair | Eager Threshold (KB) |
|--------------|----------------------|
| 0 – 4 | 32 |
| 1 – 5 | 64 |
| 2 – 6 | 128 |
| 3 – 7 | 32 |

- Default
 - Poor overlap; Low memory requirement
- Manually Tuned
 - Good overlap; High memory requirement
- Dynamic + Adaptive
 - Good overlap; Optimal memory requirement

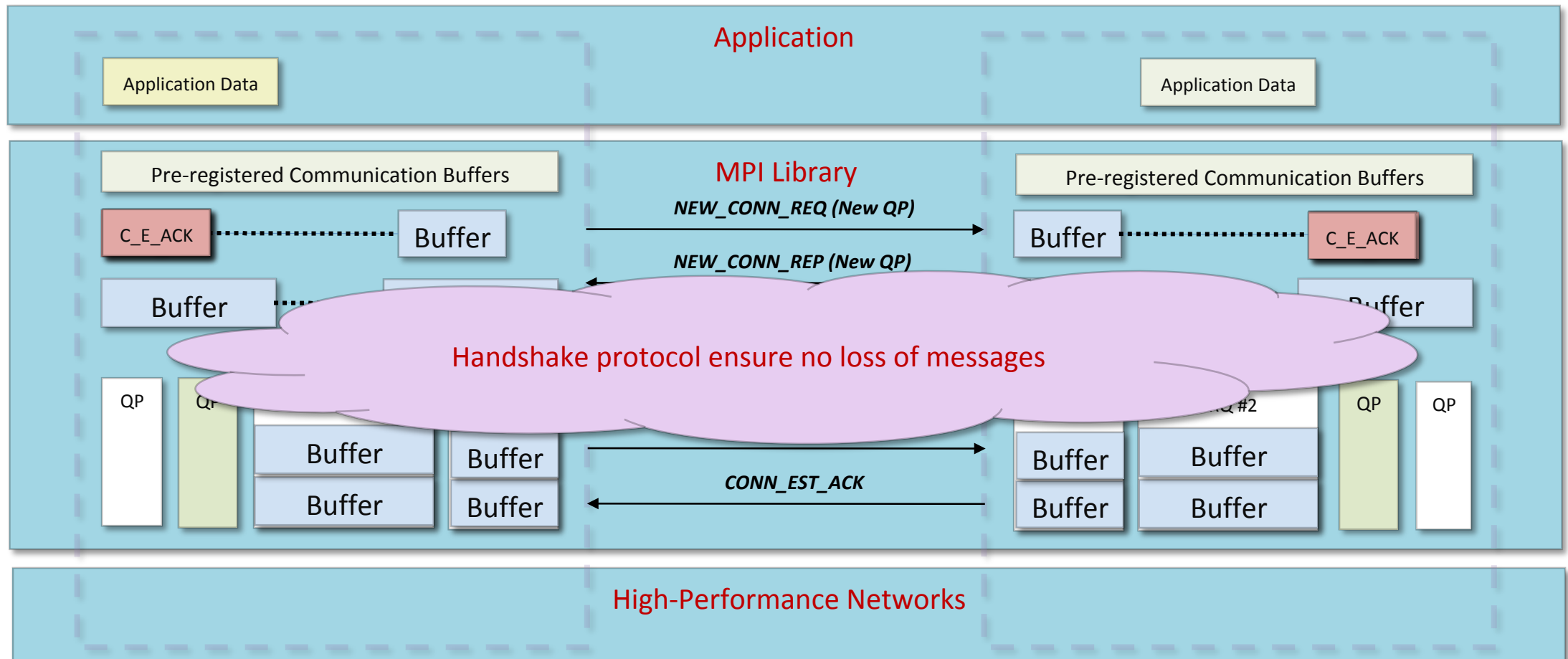
Identifying the New Eager-Threshold and Allocating Resources for Change

- $Threshold_{new} = 2^{\lceil \log_2 \left(\frac{\sum \text{sizeof}(Rndv\ Msg + Pkt\ Header)}{\text{Number of Rndv Msgs}} \right) \rceil} + offset$; **Failure: "-1"**
- **Allocate larger internal communication buffers of larger size and receive queues**



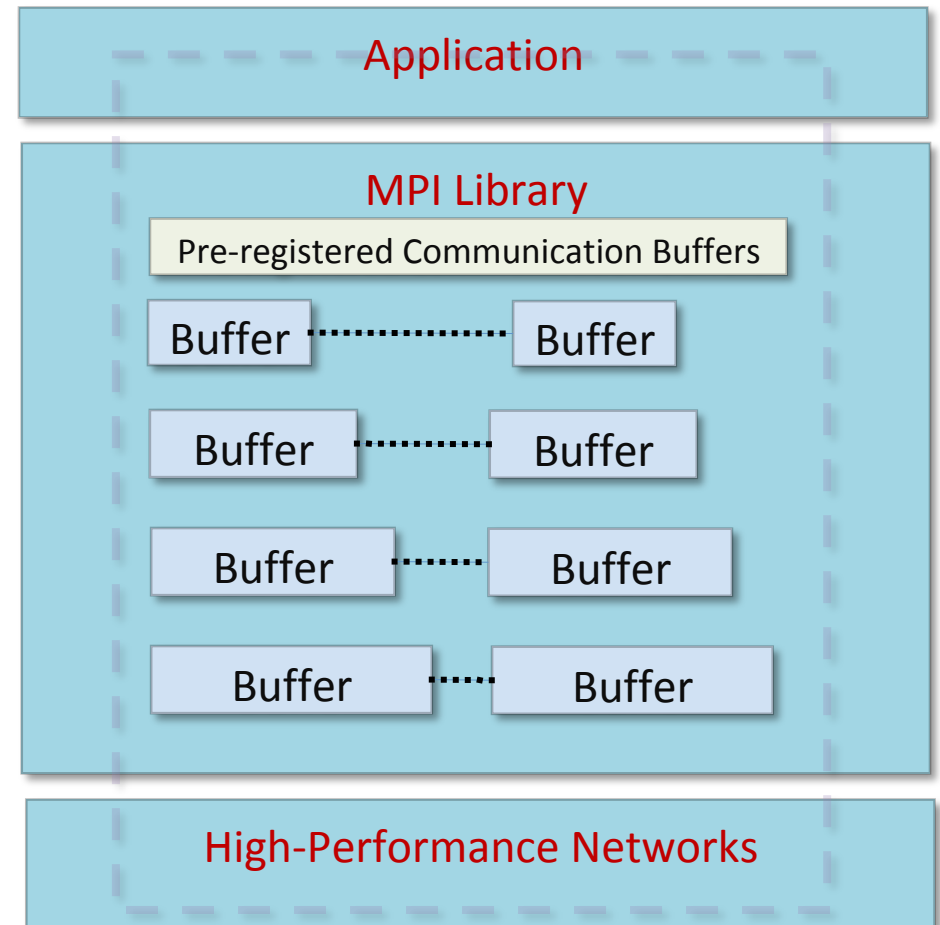
Designing Dynamic and Adaptive Point-to-point Protocols

- Process pair always has one active connection
- Messages will not have to wait for connection establishment



Mitigating Memory Footprint Requirements

- Increase in memory allocated is a concern
- Proposed design attempts to free them if
 - Buffer has not been in use continually for user defined period of time
 - Uses weights to determine which set of buffers are least recently used
 - Prevents trashing behavior where library gets into continuous loop of allocation and deallocation
- **Significantly** reduces memory overhead of dynamic and adaptive designs to less than 50% of what the manually tuned designs can offer.




Using MVAPICH2 and TAU

- To set CVARs or read PVARs using TAU for an uninstrumented binary:

```
% export TAU_TRACK_MPI_T_PVARS=1
% export TAU_MPI_T_CVAR_METRICS=
    MPIR_CVAR_VBUF_POOL_REDUCED_VALUE[1],
    MPIR_CVAR_IBA_EAGER_THRESHOLD
% export TAU_MPI_T_CVAR_VALUES=32,64000
% export PATH=/path/to/tau/x86_64/bin:$PATH
% mpirun -np 1024 tau_exec -T mvapich2,mpit ./a.out
% paraprof
```

VBUF usage without CVARs

TAU: ParaProf: Context Events for: node 0 - mpit_withoutcvar_bt.C.1k.ppk

| Name  | MaxValue | MinValue | MeanValue | Std. Dev. | NumSamples | Total |
|--|------------------|------------------|------------------|-----------|------------|------------------|
| mv2_total_vbuf_memory (Total amount of memory in bytes used for VBUFs) | 3,313,056 | 3,313,056 | 3,313,056 | 0 | 1 | 3,313,056 |
| mv2_ud_vbuf_allocated (Number of UD VBUFs allocated) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_available (Number of UD VBUFs available) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_freed (Number of UD VBUFs freed) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_inuse (Number of UD VBUFs inuse) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_max_use (Maximum number of UD VBUFs used) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_vbuf_allocated (Number of VBUFs allocated) | 320 | 320 | 320 | 0 | 1 | 320 |
| mv2_vbuf_available (Number of VBUFs available) | 255 | 255 | 255 | 0 | 1 | 255 |
| mv2_vbuf_freed (Number of VBUFs freed) | 25,545 | 25,545 | 25,545 | 0 | 1 | 25,545 |
| mv2_vbuf_inuse (Number of VBUFs inuse) | 65 | 65 | 65 | 0 | 1 | 65 |
| mv2_vbuf_max_use (Maximum number of VBUFs used) | 65 | 65 | 65 | 0 | 1 | 65 |
| num_calloc_calls (Number of MPIT_calloc calls) | 89 | 89 | 89 | 0 | 1 | 89 |
| num_free_calls (Number of MPIT_free calls) | 47,801 | 47,801 | 47,801 | 0 | 1 | 47,801 |
| num_malloc_calls (Number of MPIT_malloc calls) | 49,258 | 49,258 | 49,258 | 0 | 1 | 49,258 |
| num_memalign_calls (Number of MPIT_memalign calls) | 34 | 34 | 34 | 0 | 1 | 34 |
| num_memalign_free_calls (Number of MPIT_memalign_free calls) | 0 | 0 | 0 | 0 | 0 | 0 |

VBUF usage with CVARs

TAU: ParaProf: Context Events for: node 0 - bt-mz.E.vbuf_pool_16.1k.ppk

| Name Δ | MaxValue | MinValue | MeanValue | Std. Dev. | NumSamp... | Total |
|--|-----------|-----------|-----------|-----------|------------|-----------|
| mv2_total_vbuf_memory (Total amount of memory in bytes used for VBUFs) | 1,815,056 | 1,815,056 | 1,815,056 | 0 | 1 | 1,815,056 |
| mv2_ud_vbuf_allocated (Number of UD VBUFs allocated) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_available (Number of UD VBUFs available) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_freed (Number of UD VBUFs freed) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_inuse (Number of UD VBUFs inuse) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_ud_vbuf_max_use (Maximum number of UD VBUFs used) | 0 | 0 | 0 | 0 | 0 | 0 |
| mv2_vbuf_allocated (Number of VBUFs allocated) | 160 | 160 | 160 | 0 | 1 | 160 |
| mv2_vbuf_available (Number of VBUFs available) | 94 | 94 | 94 | 0 | 1 | 94 |
| mv2_vbuf_freed (Number of VBUFs freed) | 5,479 | 5,479 | 5,479 | 0 | 1 | 5,479 |
| mv2_vbuf_inuse (Number of VBUFs inuse) | 66 | 66 | 66 | 0 | 1 | 66 |
| mv2_vbuf_max_use (Maximum number of VBUFs used) | 66 | 66 | 66 | 0 | 1 | 66 |
| num_calloc_calls (Number of MPIT_calloc calls) | 89 | 89 | 89 | 0 | 1 | 89 |
| num_free_calls (Number of MPIT_free calls) | 130 | 130 | 130 | 0 | 1 | 130 |
| num_malloc_calls (Number of MPIT_malloc calls) | 1,625 | 1,625 | 1,625 | 0 | 1 | 1,625 |
| num_memalign_calls (Number of MPIT_memalign calls) | 56 | 56 | 56 | 0 | 1 | 56 |
| num_memalign_free_calls (Number of MPIT_memalign_free calls) | 0 | 0 | 0 | 0 | 0 | 0 |

TAU: ParaProf Manager

Applications

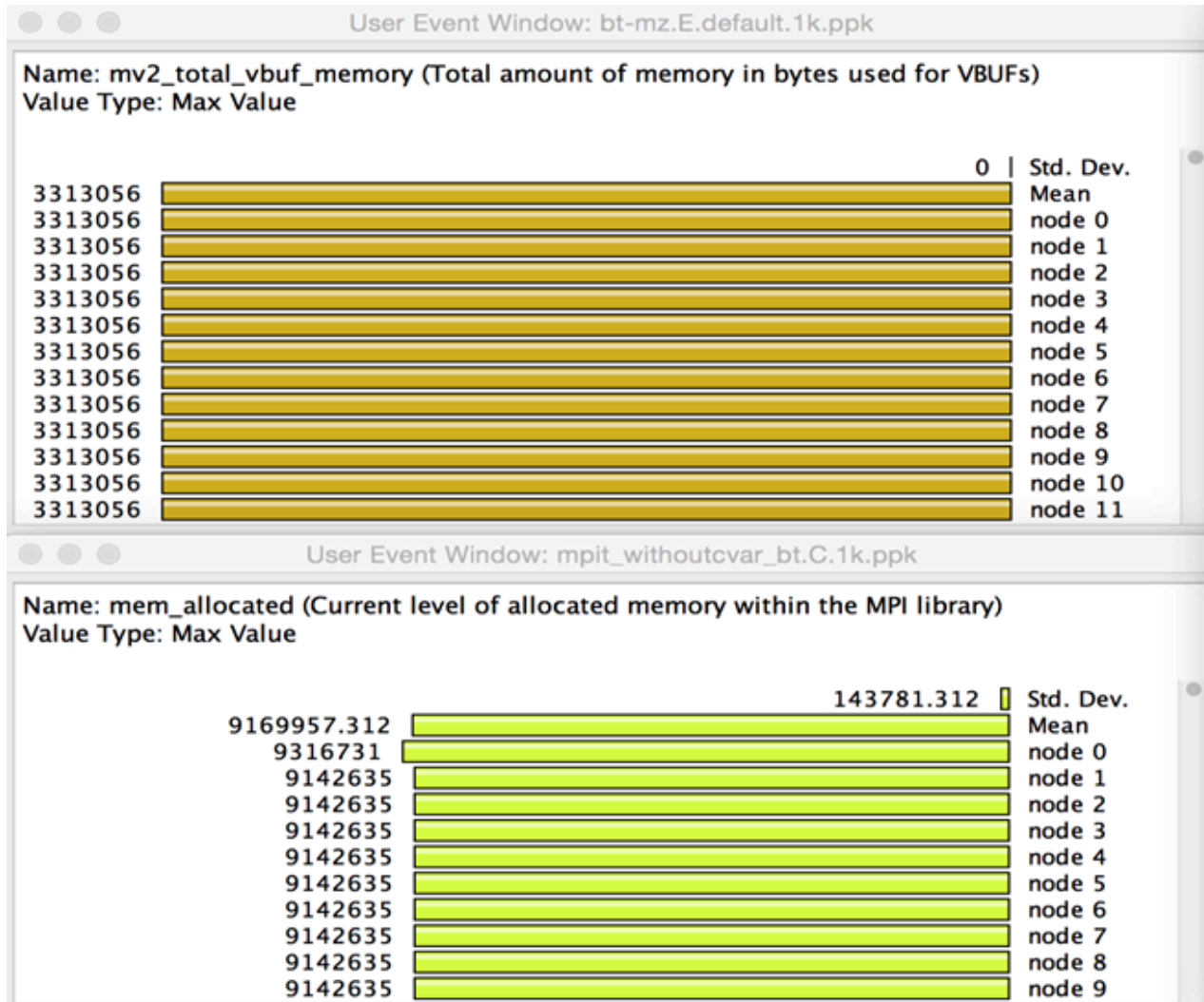
- Standard Applications
 - Default App
 - Default Exp
 - bt-mz.E.vbuf_pool_16.1k.pp
 - TIME

TrialField Value

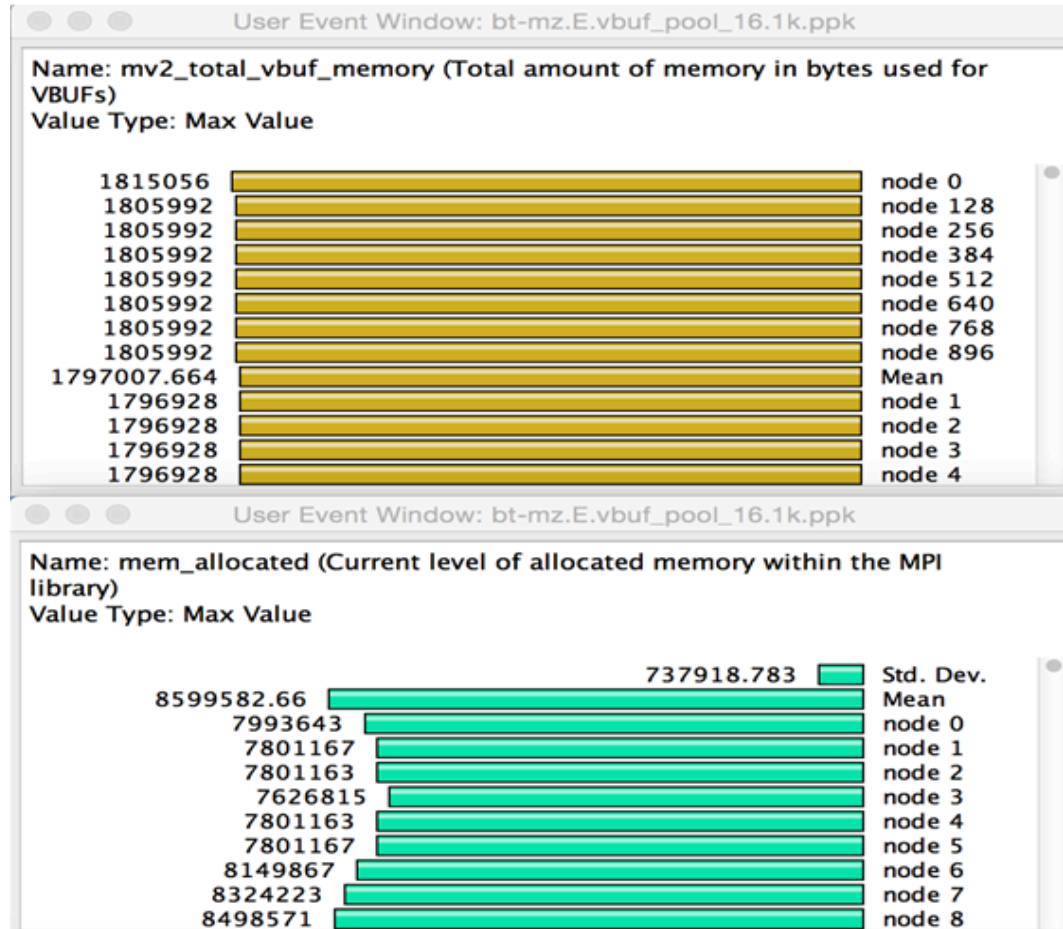
- MPI Processor Name c526-502.stampede.tacc.utexas.edu
- MPIR_CVAR_VBUF_POOL_SIZE 0 (old) -> 16 (new), This set the size of the VBUF pool

Total memory used by VBUFs is reduced from 3,313,056 to 1,815,056

VBUF Memory Usage Without CVAR



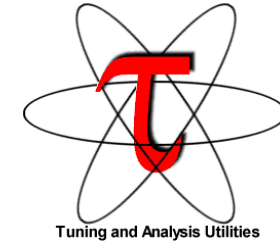
VBUF Memory Usage With CVAR



```
% export TAU_TRACK_MPI_T_PVARS=1  
% export TAU_MPI_T_CVAR_METRICS=MPIR_CVAR_VBUF_POOL_SIZE  
% export TAU_MPI_T_CVAR_VALUES=16  
% mpirun -np 1024 tau_exec -T mvapich2 .la.out
```

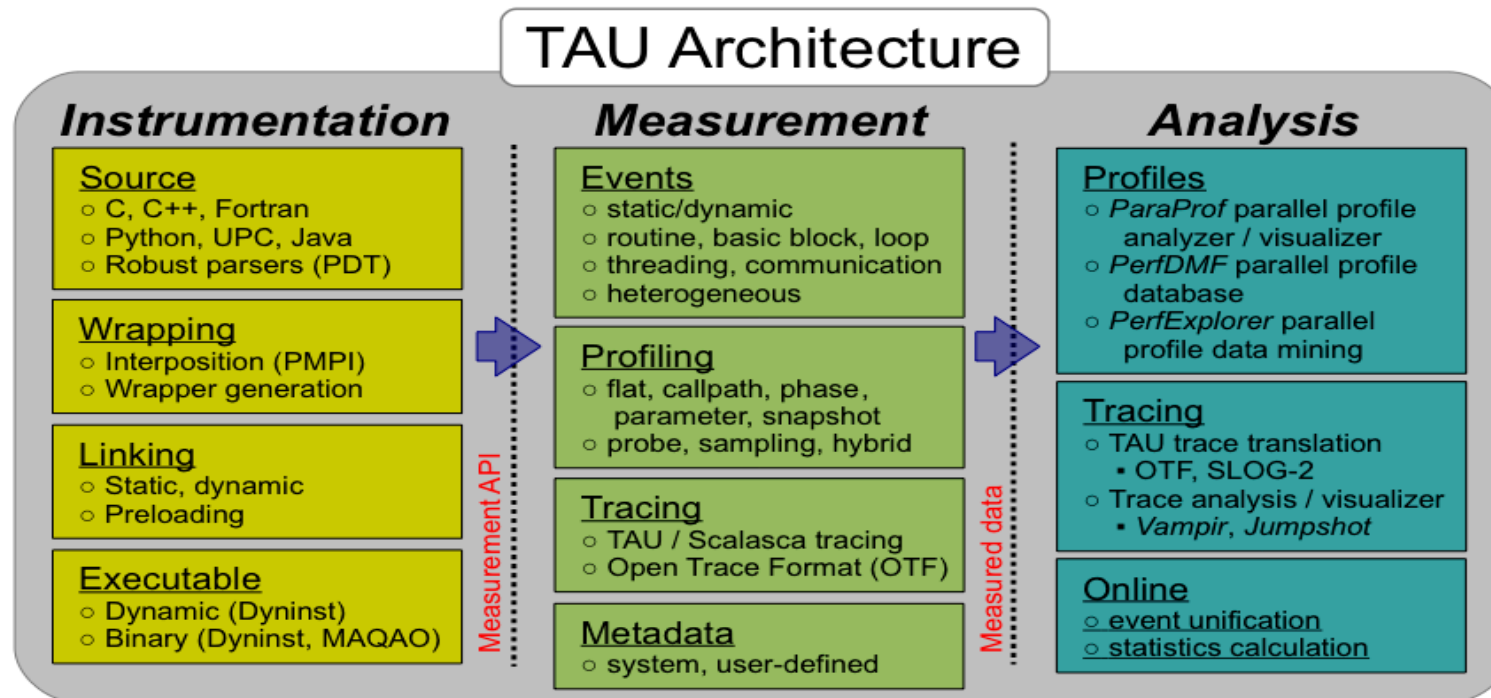
Outline

- Introduction
- The MPI Tools Interfaces and Benefits
- Integrating TAU and MVAPICH2 with MPI_T
- Use Cases
- **TAU Performance System[®]**



Parallel performance framework and toolkit

- Supports all HPC platforms, compilers, runtime system
- Provides portable instrumentation, measurement, analysis



TAU Performance System

Instrumentation

- Fortran, C++, C, UPC, Java, Python, Chapel, Spark
- Automatic instrumentation

Measurement and analysis support

- MPI, OpenSHMEM, ARMCI, PGAS, DMAPP
- pthreads, OpenMP, OMPT interface, hybrid, other thread models
- GPU, CUDA, OpenCL, OpenACC
- Parallel profiling and tracing
- Use of Score-P for native OTF2 and CUBEX generation

Analysis

- Parallel profile analysis (ParaProf), data mining (PerfExplorer)
- Performance database technology (TAUdb)
- 3D profile browser

Instrumentation

Add hooks in the code to perform measurements

Source instrumentation using a preprocessor

- Add timer start/stop calls in a copy of the source code.
- Use Program Database Toolkit (PDT) for parsing source code.
- Requires recompiling the code using TAU shell scripts (tau_cc.sh, tau_f90.sh)
- Selective instrumentation (filter file) can reduce runtime overhead and narrow instrumentation focus.

Compiler-based instrumentation

- Use system compiler to add a special flag to insert hooks at routine entry/exit.
- Requires recompiling using TAU compiler scripts (tau_cc.sh, tau_f90.sh...)

Runtime preloading of TAU's Dynamic Shared Object (DSO)

- No need to recompile code! Use **mpirun tau_exec ./app** with options.
- Requires dynamic executable (link using **-dynamic** on Theta).

TAU Instrumentation Approach

Supports both direct and indirect performance observation

- Direct instrumentation of program (system) code (probes)
- Instrumentation invokes performance measurement
- Event measurement: performance data, meta-data, context
- Indirect mode supports sampling based on periodic timer or hardware performance counter overflow based interrupts

Support for user-defined events

- **Interval** (Start/Stop) events to measure exclusive & inclusive duration
- **Atomic events** (Trigger at a single point with data, e.g., heap memory)
 - Measures total, samples, min/max/mean/std. deviation statistics
- **Context events** (are atomic events with executing context)
 - Measures above statistics for a given calling path

Direct Observation: Events

Event types

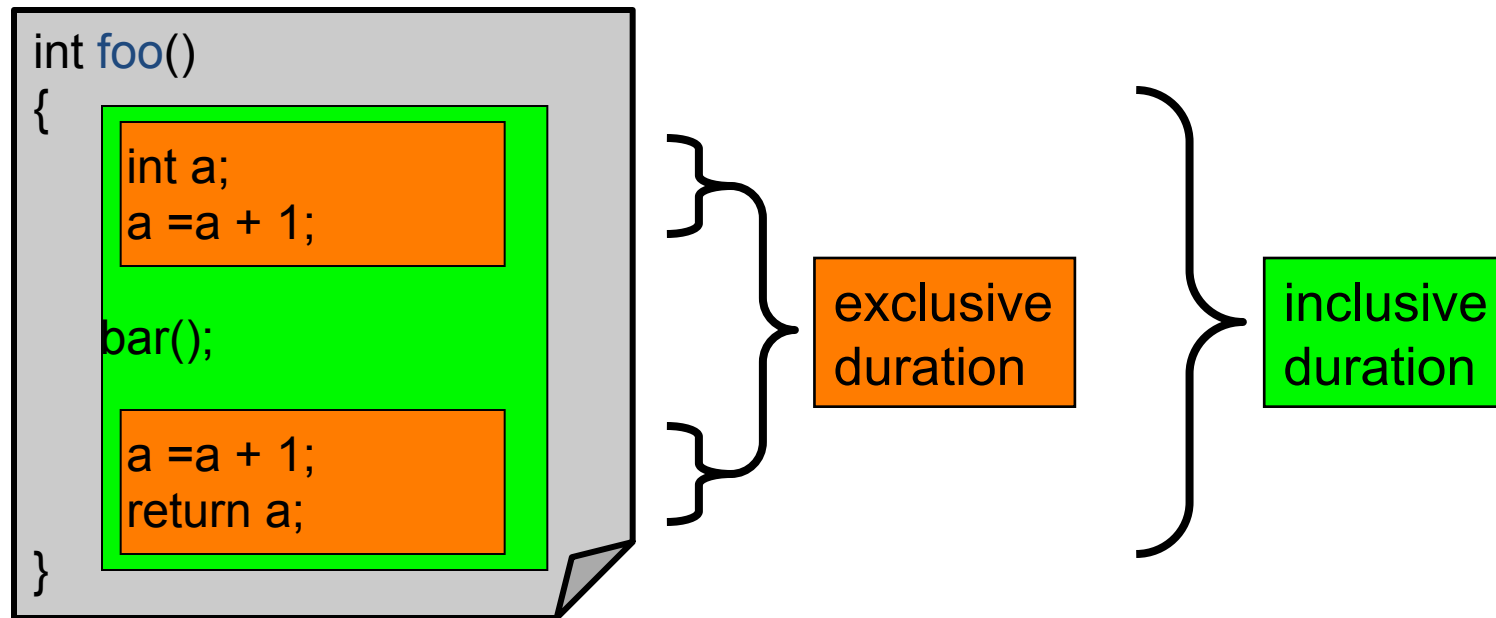
- Interval events (begin/end events)
 - Measures exclusive & inclusive durations between events
 - Metrics monotonically increase
- Atomic events (trigger with data value)
 - Used to capture performance data state
 - Shows extent of variation of triggered values (min/max/mean)

Code events

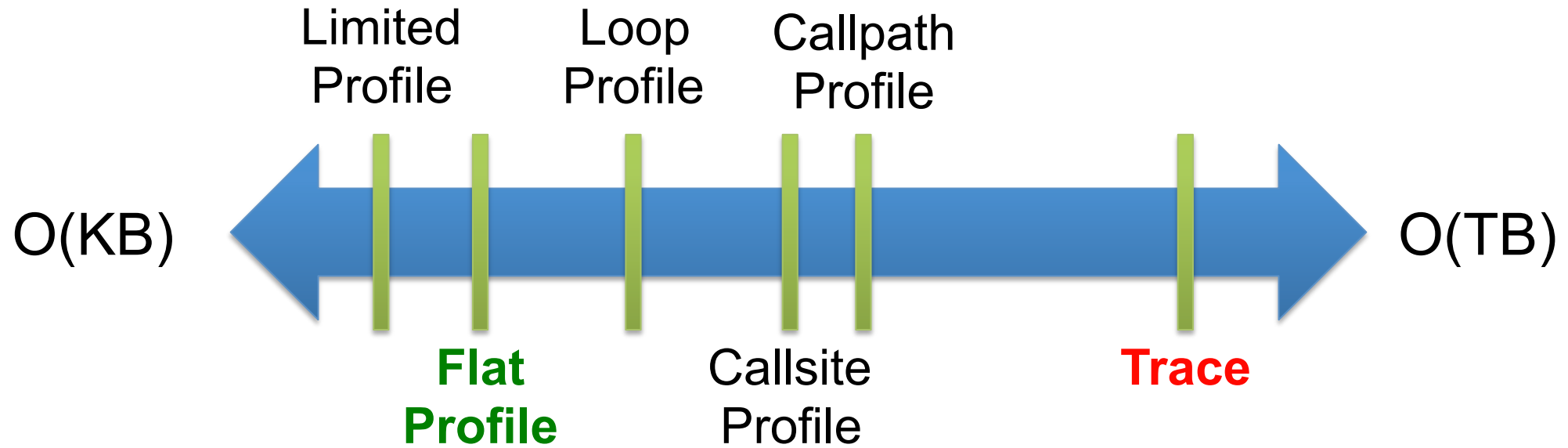
- Routines, classes, templates
- Statement-level blocks, loops

Inclusive and Exclusive Profiles

- Performance with respect to code regions
- Exclusive measurements for region only
- Inclusive measurements includes child regions



How much data do you want?



Types of Performance Profiles

Flat profiles

- Metric (e.g., time) spent in an event
- Exclusive/inclusive, # of calls, child calls, ...

Callpath profiles

- Time spent along a calling path (edges in callgraph)
- “*main=> f1 => f2 => MPI_Send*”
- Set the **TAU_CALLPATH** and **TAU_CALLPATH_DEPTH** environment variables

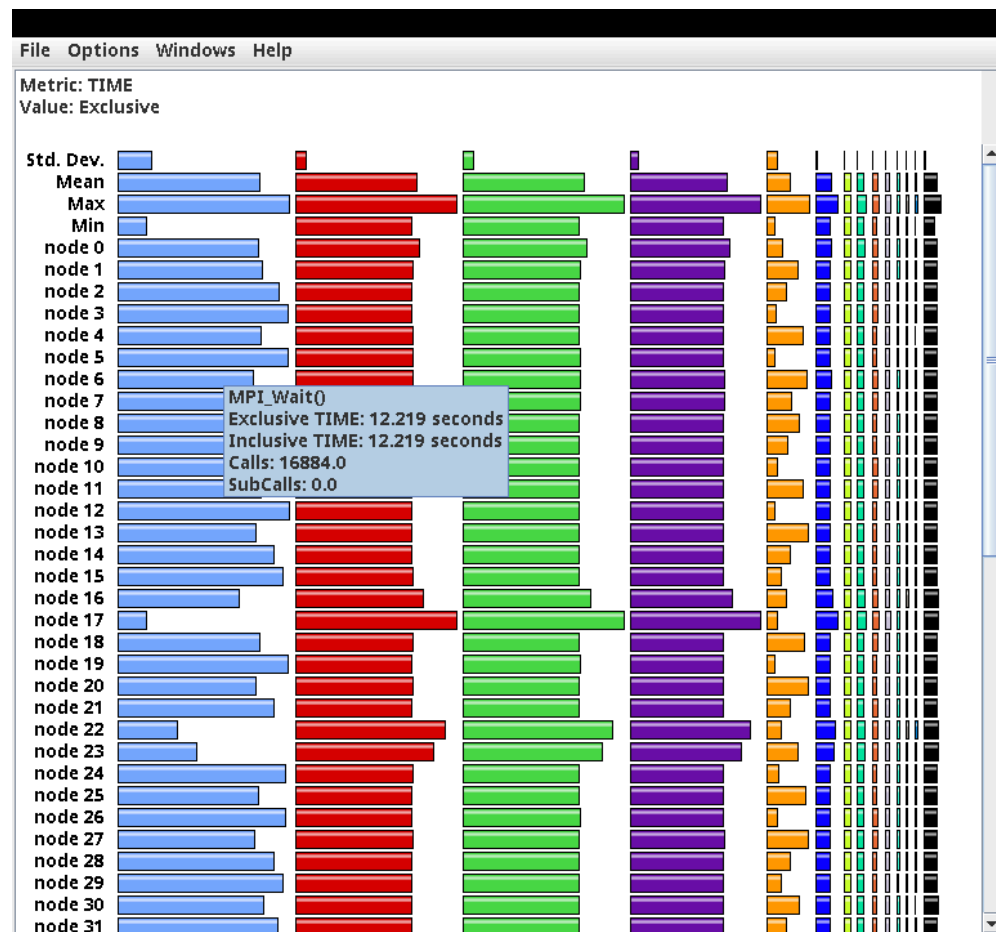
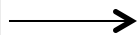
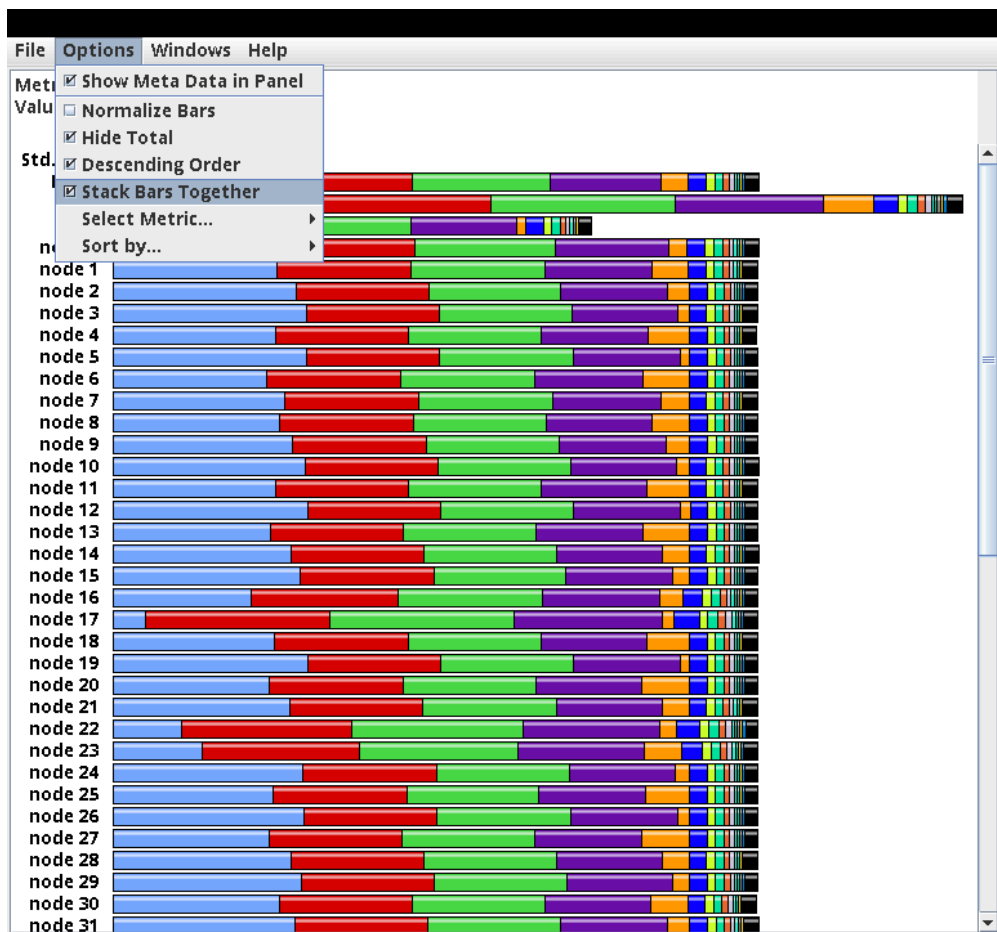
Callsite profiles

- Time spent along in an event at a given source location
- Set the **TAU_CALLSITE** environment variable

Phase profiles

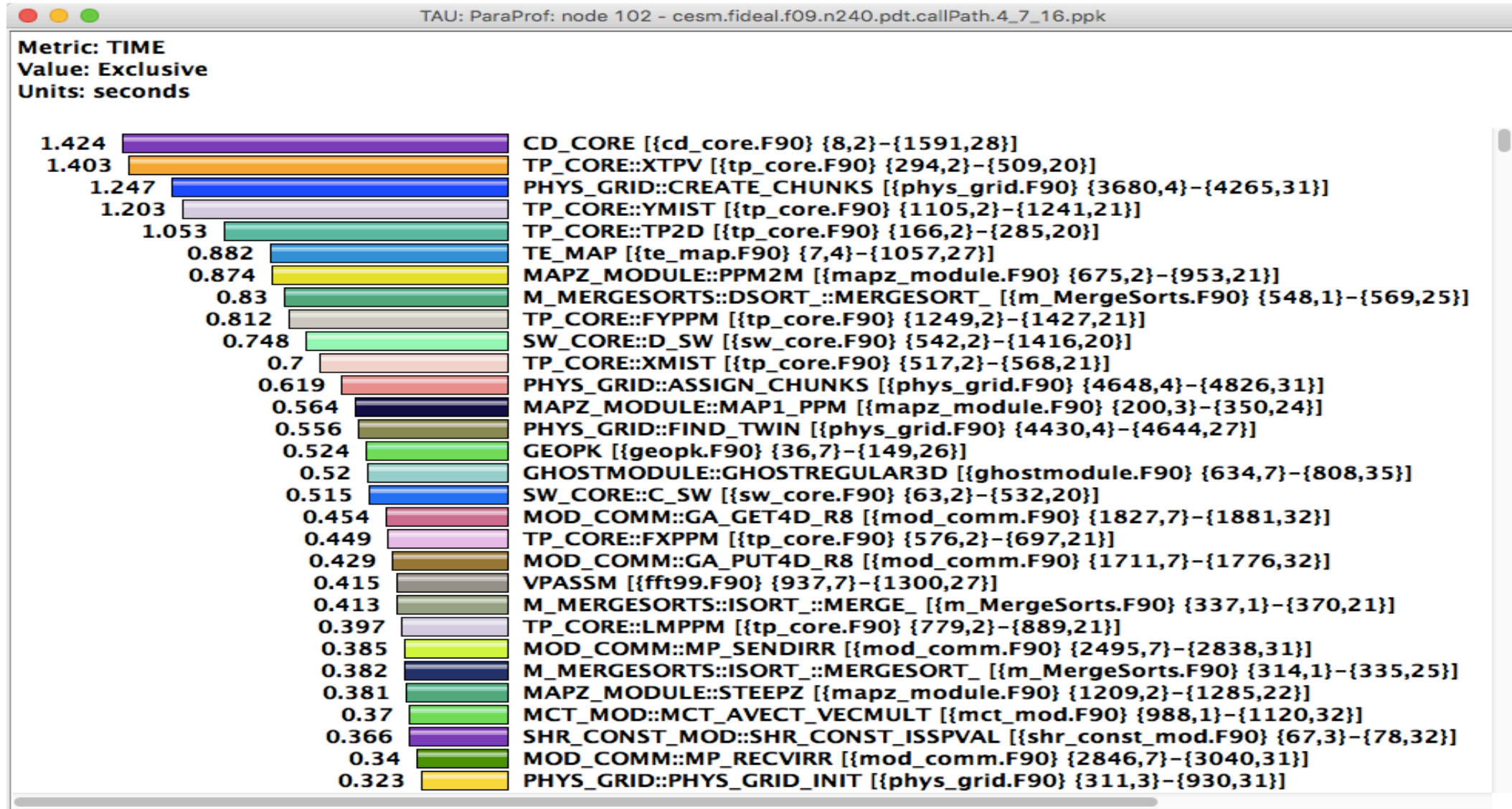
- Flat profiles under a phase (nested phases allowed)
- Default “main” phase
- Supports static or dynamic (e.g. per-iteration) phases

ParaProf Profile Browser

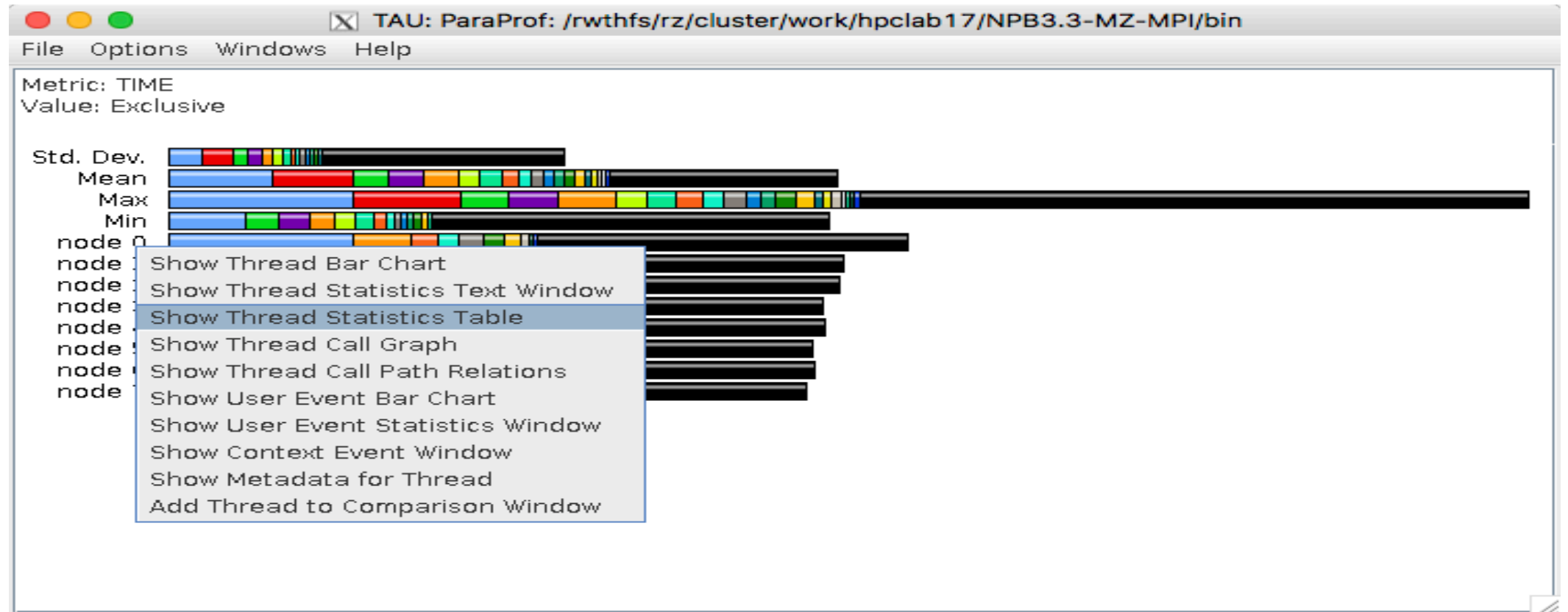


Click “node X” next to see details

TAU – Flat Profile



ParaProf Thread Statistics Table



Right click over "node X" and choose Show Thread Statistics Table

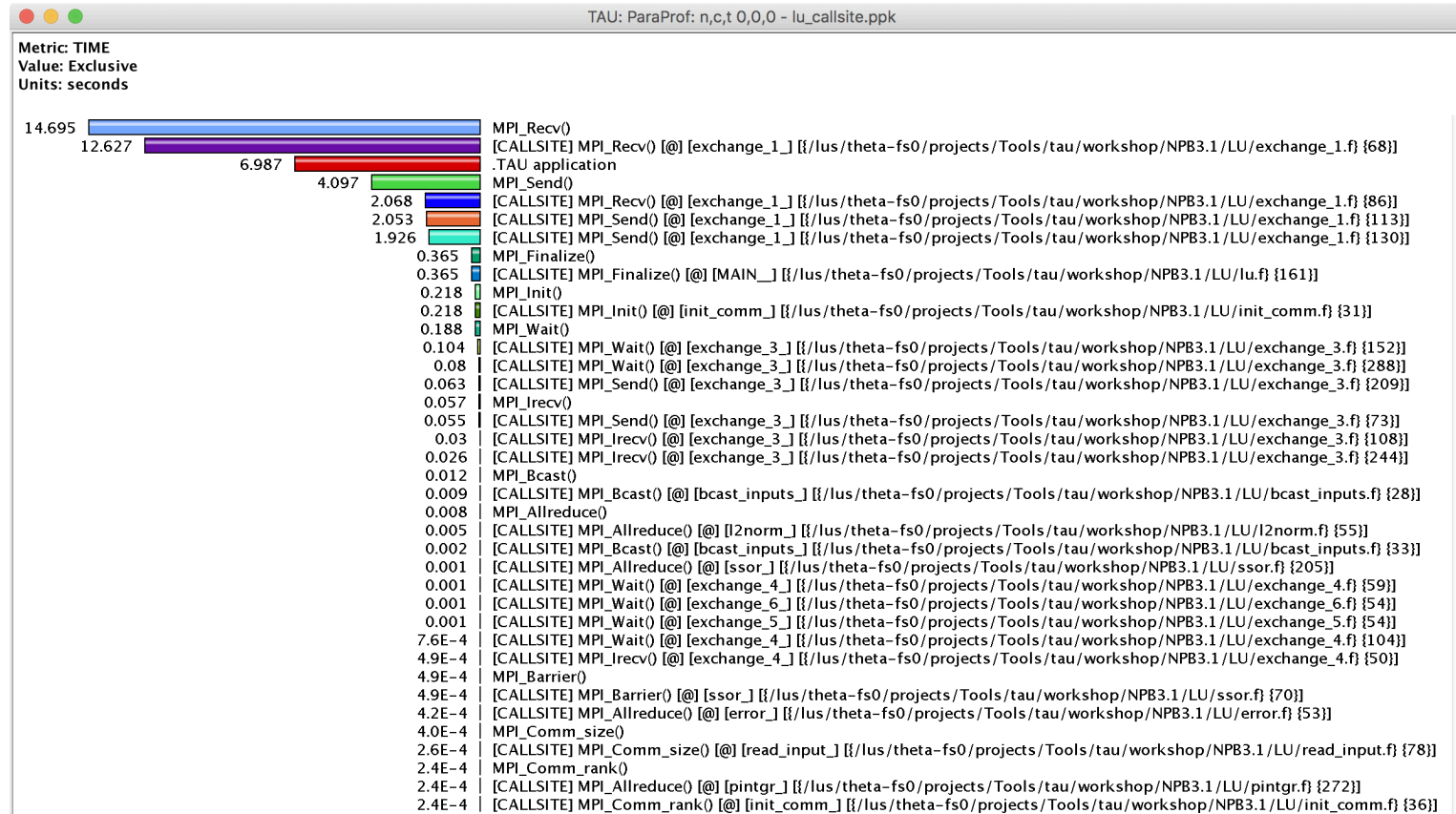
TAU – Callsite Profiling

TAU: ParaProf: Statistics for: node 0 - clover_callsite.ppk

| Name | Excl... | Inclu... | Calls | Chil... |
|---|---------|----------|-------|---------|
| .TAU application | 6.152 | 8.249 | 1 | 28,383 |
| [CALLSITE] void start_pes_(int *) [@@] [/nfsprojects/volta-projects/tau/tau-2.24.1/craycnl/lib/libTAUsh-gnu-papi-shmem-pdt.so] UNRESOLVED ADDR | 0.747 | 0.747 | 1 | 0 |
| void start_pes_(int *) | 0.747 | 0.747 | 1 | 0 |
| void shmem_barrier_all_0 | 0.624 | 0.624 | 9,229 | 0 |
| [CALLSITE] void shmem_barrier_all_0 [@@] [__clover_module_MOD_clover_exchange_message] [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {572}] | 0.401 | 0.401 | 4,610 | 0 |
| [CALLSITE] void shmem_finalize_0 [@@] [/nfsprojects/volta-projects/tau/tau-2.24.1/craycnl/lib/libTAUsh-gnu-papi-shmem-pdt.so] UNRESOLVED ADDR | 0.314 | 0.314 | 1 | 0 |
| void shmem_finalize_0 | 0.314 | 0.314 | 1 | 0 |
| [CALLSITE] void shmem_barrier_all_0 [@@] [__clover_module_MOD_clover_exchange_message] [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {643}] | 0.223 | 0.223 | 4,610 | 0 |
| void shmem_put64_nb_(void *, void *, int *, int *, void *) | 0.159 | 0.159 | 9,220 | 0 |
| void shmem_put64_(void *, void *, int *, int *) | 0.126 | 0.126 | 9,220 | 0 |
| void shmem_real8_max_to_all_(void *, void *, int *, int *, int *, int *, int *, int *, void *, long *) | 0.081 | 0.081 | 400 | 0 |
| [CALLSITE] void shmem_put64_nb_(void *, void *, int *, int *, void *) [@@] [__clover_module_MOD_clover_exchange_message] [{/home/ssshend/CloverLeaf_ | 0.07 | 0.07 | 4,610 | 0 |
| [CALLSITE] void shmem_put64_(void *, void *, int *, int *) [@@] [__clover_module_MOD_clover_exchange_message] [{/home/ssshend/CloverLeaf_OpenSHMEM | 0.063 | 0.063 | 4,610 | 0 |
| [CALLSITE] void shmem_real8_max_to_all_(void *, void *, int *, int *, int *, int *, void *, long *) [@@] [hydro_] [{/home/ssshend/CloverLeaf_OpenSHMEM/hydr | 0.046 | 0.046 | 200 | 0 |
| [CALLSITE] void shmem_real8_min_to_all_(void *, void *, int *, int *, int *, int *, void *, long *) [@@] [/nfsprojects/volta-projects/tau/tau-2.24.1/craycnl/lib/ | 0.04 | 0.04 | 200 | 0 |
| void shmem_real8_min_to_all_(void *, void *, int *, int *, int *, int *, void *, long *) | 0.04 | 0.04 | 200 | 0 |
| [CALLSITE] void shmem_real8_max_to_all_(void *, void *, int *, int *, int *, int *, void *, long *) [@@] [hydro_] [{/home/ssshend/CloverLeaf_OpenSHMEM/hydr | 0.036 | 0.036 | 200 | 0 |
| [CALLSITE] void shmem_put64_nb_(void *, void *, int *, int *, void *) [@@] [__clover_module_MOD_clover_exchange] [{/home/ssshend/CloverLeaf_OpenSHME | 0.028 | 0.028 | 601 | 0 |

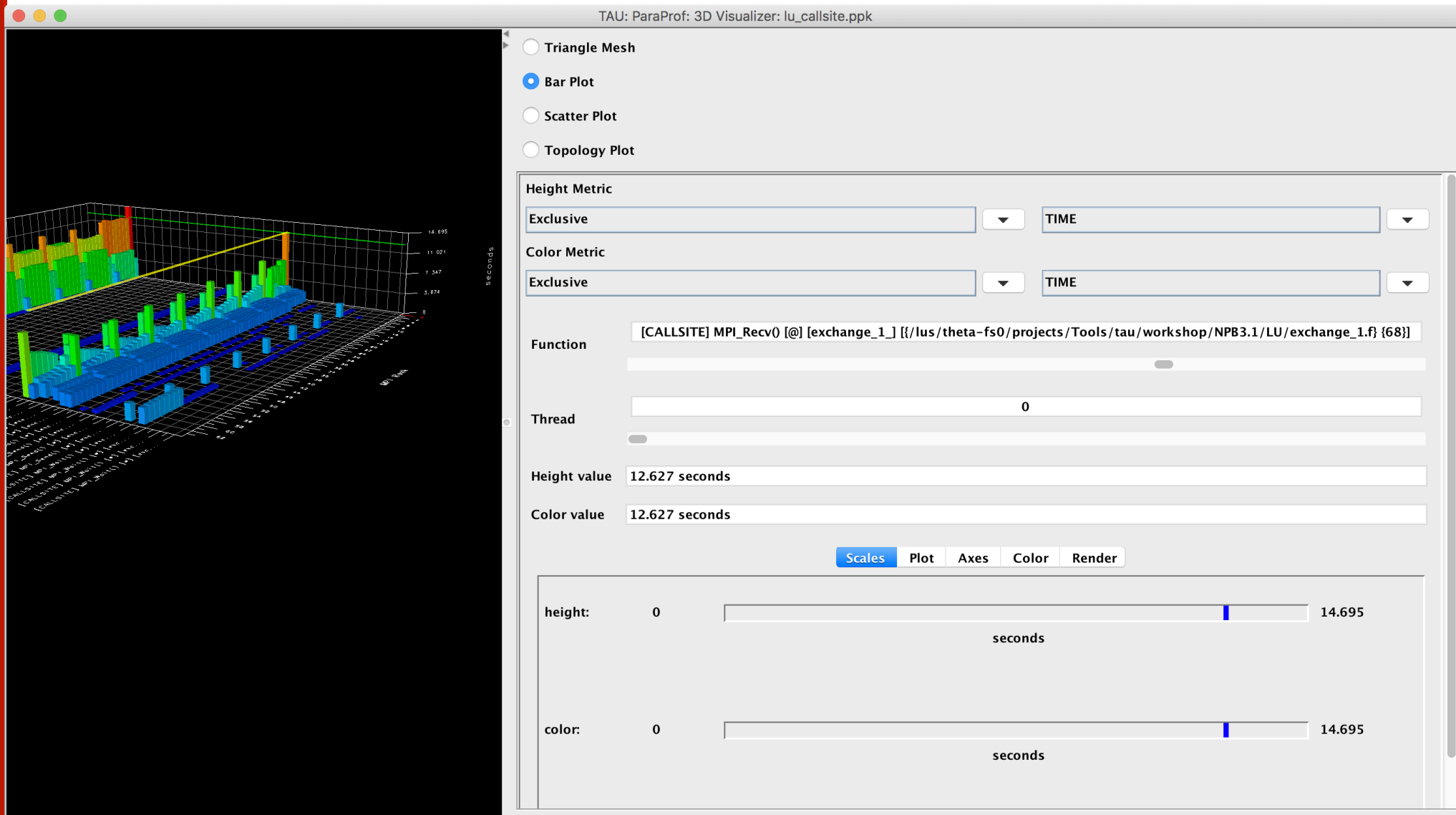
% export TAU_CALLSITE=1

Callsite Profiling and Tracing



% export TAU_CALLSITE=1

Callsite Profiling and Tracing



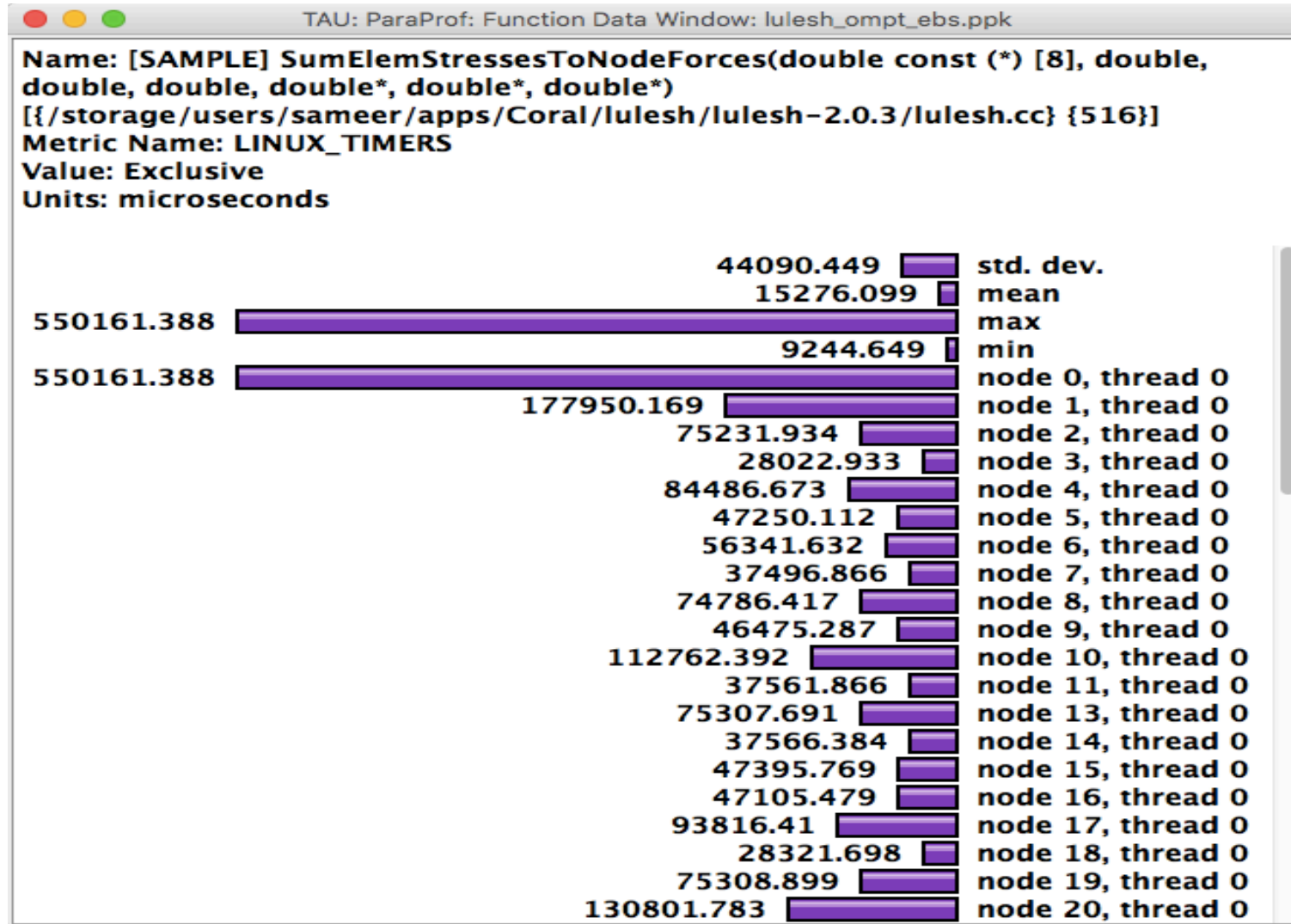
TAU – Callstack Sampling

TAU: ParaProf: Statistics for: n,c,t 0,0,0 - clover_gnu_ebs_unw_call.ppk

| Name | Inclusive... | Calls |
|--|--------------|--------|
| ▼ .TAU application | 34.979 | 1 |
| ▶ [CONTEXT] .TAU application | 31.647 | 632 |
| ▼ void shmем_barrier_all_0 | 1.219 | 46,029 |
| ▼ [CONTEXT] void shmем_barrier_all_0 | 1.599 | 32 |
| ▼ [UNWIND] [/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90.41] [@@] UNRESOLVED /lib64/libc-2.11.3.so | 1.599 | 32 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90.62 [@@] main [{/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90} {41}] | 0.85 | 17 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/advection.f90.102 [@@] hydro_ [{/home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90} {62}] | 0.55 | 11 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/update_halo.f90.36 [@@] __advection_module_MOD_advection [{/home/ssshend/CloverLeaf_OpenSHMEM/update_halo.f90} {36}] | 0.55 | 11 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/clover.f90.292 [@@] __update_halo_module_MOD_update_halo [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {292}] | 0.5 | 10 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/clover.f90.572 [@@] __clover_module_MOD_clover_exchange [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {572}] | 0.5 | 10 |
| ▼ [UNWIND] UNRESOLVED [@@] __clover_module_MOD_clover_exchange_message [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {572}] | 0.5 | 10 |
| ▼ [UNWIND] [/notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_opt_barrier.c.118] [@@] UNRESOLVED /nfsprojects/vol | 0.45 | 9 |
| ▶ [SAMPLE] _smai_smp_barrier_in [{/notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_opt_barrier.c} {118}] | 0.45 | 9 |
| ▶ [UNWIND] [/notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_internal.h.88] [@@] UNRESOLVED /nfsprojects/vol | 0.05 | 1 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/clover.f90.461 [@@] __update_halo_module_MOD_update_halo [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {461}] | 0.05 | 1 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/advection.f90.72 [@@] hydro_ [{/home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90} {62}] | 0.15 | 3 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/advection.f90.55 [@@] hydro_ [{/home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90} {62}] | 0.15 | 3 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90.52 [@@] main [{/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90} {41}] | 0.5 | 10 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90.54 [@@] main [{/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90} {41}] | 0.25 | 5 |
| ▶ void start_pes_(int *) | 0.508 | 1 |
| ▼ void shmем_real8_max_to_all_(void *, void *, int *, int *, int *, int *, void *, long *) | 0.325 | 2,000 |
| ▼ [CONTEXT] void shmем_real8_max_to_all_(void *, void *, int *, int *, int *, int *, int *, void *, long *) | 0.5 | 10 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90.41 [@@] UNRESOLVED /lib64/libc-2.11.3.so | 0.5 | 10 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90.58 [@@] main [{/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90} {41}] | 0.45 | 9 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/PdV.f90.107 [@@] hydro_ [{/home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90} {58}] | 0.45 | 9 |
| ▼ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/clover.f90.740 [@@] __pdv_module_MOD_pdv [{/home/ssshend/CloverLeaf_OpenSHMEM/PdV.f90} {107}] | 0.45 | 9 |
| ▼ [UNWIND] UNRESOLVED [@@] __clover_module_MOD_clover_check_error [{/home/ssshend/CloverLeaf_OpenSHMEM/clover.f90} {740}] | 0.45 | 9 |
| ▼ [UNWIND] [/notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_reduction.h.207] [@@] UNRESOLVED /nfsprojects/vol | 0.45 | 9 |
| ▼ [UNWIND] /notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_opt_reduction.h.788 [@@] pshmem_double_max_tc | 0.45 | 9 |
| ▼ [UNWIND] /notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_opt_reduction.h.107 [@@] _smai_opt_double_max | 0.45 | 9 |
| ▶ [SAMPLE] _smai_smp_reduce_double_max [{/notbackedup/tmp/ulib/mpt/nightly/7.2/062215-RC/sma_dmapp/src/shmem_opt_reduction.h} {107}] | 0.45 | 9 |
| ▶ [UNWIND] /home/ssshend/CloverLeaf_OpenSHMEM/hydro.f90.54 [@@] main [{/home/ssshend/CloverLeaf_OpenSHMEM/clover_leaf.f90} {41}] | 0.05 | 1 |

% export TAU_SAMPLING=1; export TAU_EBS_UNWIND=1

TAU – Event Based Sampling (EBS)



% export TAU_SAMPLING=1

TAU – Callpath Profiling

TAU: ParaProf: Statistics for: node 5 - fun3d_d19.ppk

| Name | Exclusive... | Inclusive... | Calls | Child... |
|--|--------------|--------------|-------|----------|
| ▾ .TAU application | 0 | 221.298 | 1 | 1 |
| ▾ ▾ NODET [{main.f90} {4,1}–{35,17}] | 0 | 221.298 | 1 | 105 |
| ▸ ▸ FLOW::ITERATE [{flow.F90} {1692,14}] | 0 | 197.989 | 100 | 500 |
| ▾ ▾ FLOW::INITIALIZE_DATA [{flow.F90} {465,14}] | 0 | 22.707 | 1 | 2 |
| ▾ ▾ ▾ FLOW::INITIALIZE_DATA2 [{flow.F90} {663,14}] | 0.002 | 22.705 | 1 | 197 |
| ▾ ▾ ▾ ▾ PPARTY_PREPROCESSOR::PPARTY_PREPROCESS [{pparty_preprocessor.f90} {28,14}] | 0 | 20.897 | 1 | 23 |
| ▾ ▾ ▾ ▾ ▾ PPARTY_PREPROCESSOR::PPARTY_READ_GRID [{pparty_preprocessor.f90} {735,14}] | 0 | 16.726 | 1 | 2 |
| ▾ ▾ ▾ ▾ ▾ ▾ PUNS3D_IO_C2N::PUNS3D_READ_VGRID_C2N [{puns3d_io_c2n.f90} {1543,14}] | 0.011 | 16.725 | 1 | 11 |
| ▾ ▾ ▾ ▾ ▾ ▾ ▾ PUNS3D_IO_C2N::PUNS3D_READ_VGRID_C2N_SM [{puns3d_io_c2n.f90} {1641,14}] | 0 | 16.656 | 1 | 5 |
| ▾ ▾ ▾ ▾ ▾ ▾ ▾ ▾ PUNS3D_IO_C2N::DISTRIBUTE_TET [{puns3d_io_c2n.f90} {1819,14}] | 0.117 | 16.572 | 1 | 5 |
| ▾ ▾ ▾ ▾ ▾ ▾ ▾ ▾ ▾ LMPI::INTEGR_MATRIX_BCAST [{lmpi.F90} {3240,3}–{3276,36}] | 0 | 16.448 | 4 | 4 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ MPI_Bcast() | 16.448 | 16.448 | 4 | 0 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ LMPI::LMPI_CONDITIONAL_STOP [{lmpi.F90} {611,3}–{672,38}] | 0 | 0.007 | 1 | 2 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PUNS3D_IO_C2N::DISTRIBUTE_XYZ [{puns3d_io_c2n.f90} {2448,14}] | 0.001 | 0.083 | 1 | 3 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ LMPI::INTEGR_SCALAR_BCAST [{lmpi.F90} {3151,3}–{3187,36}] | 0 | 0 | 3 | 3 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ LMPI::LMPI_CONDITIONAL_STOP [{lmpi.F90} {611,3}–{672,38}] | 0 | 0.058 | 1 | 2 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ LMPI::INTEGR_SCALAR_BCAST [{lmpi.F90} {3151,3}–{3187,36}] | 0 | 0 | 2 | 2 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ALLOCATIONS::INTEGER_4_MY_ALLOC_PTR2 [{allocations.f90} {1010,3}–{1026,40}] | 0 | 0 | 6 | 0 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PUNS3D_IO_C2N::DISTRIBUTE_FAST_C2N [{puns3d_io_c2n.f90} {4226,14}] | 0 | 0 | 1 | 0 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ LMPI::LMPI_CONDITIONAL_STOP [{lmpi.F90} {611,3}–{672,38}] | 0 | 0.001 | 1 | 2 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PPARTY_MIXED_ELEMENT::EDGE_POINTER_DRIVER [{pparty_mixed_element.f90} {74,3}–{50 | 0.65 | 0.873 | 1 | 174 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PPARTY::NODE_CELL_CHOPPER [{pparty.f90} {41,3}–{453,33}] | 0.288 | 0.86 | 1 | 175 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PPARTY_PUNS3D::RAW_GRID_CHECKER [{pparty_puns3d.f90} {623,14}] | 0.233 | 0.523 | 1 | 11 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PPARTY_METIS::MY_METIS [{pparty_metis.F90} {116,3}–{545,24}] | 0.313 | 0.436 | 1 | 13,132 |
| ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ ▸ PARTY_LMPI::PARTY_LMPI_SETUP_MPI_SM [{party_lmpi.f90} {613,3}–{686,40}] | 0.006 | 0.337 | 1 | 10 |

% export TAU_CALLPATH=1; export TAU_CALLPATH_DEPTH=100

TAU Atomic Events

TAU: ParaProf: Context Events for: node 0 - /Users/sameer/tmp

| Name ▾ | Total | NumSamples | MaxValue | MinValue | MeanValue | Std. Dev. |
|--|-------------|------------|----------|----------|-----------|-----------|
| Bytes Written <file=stdout> | 911 | 62 | 21 | 1 | 14.694 | 7.441 |
| Bytes Written <file=pipe> | 22 | 22 | 1 | 1 | 1 | 0 |
| Bytes Written <file=Process_Output/VelRsdl.dat> | 7,826 | 100 | 302 | 76 | 78.26 | 22.487 |
| Bytes Written <file=Process_Output/MomRsdl.dat> | 7,826 | 100 | 302 | 76 | 78.26 | 22.487 |
| Bytes Written <file=Process_Output/MassRsdl.dat> | 11,325 | 100 | 435 | 110 | 113.25 | 32.337 |
| Bytes Written <file=Grid_Output/bodyBndry.dat> | 9,724 | 5 | 8,192 | 4 | 1,944.8 | 3,174.201 |
| Bytes Written <file=/home/sameer/apps/sukra/RotCFD_Regression/case_catalog/UNS2D/N/ | 45 | 1 | 45 | 45 | 45 | 0 |
| Bytes Written <file=./Restarts/Restart_History//NACA0012_LargeGrid_00010.Rst> | 44,619,720 | 5,484 | 8,192 | 4 | 8,136.346 | 640.325 |
| Bytes Written <file=./Restarts/Restart_History//NACA0012_LargeGrid_00005.Rst> | 44,619,720 | 5,484 | 8,192 | 4 | 8,136.346 | 640.325 |
| Bytes Written <file=./Restarts//NACA0012_LargeGrid.Rst> | 44,619,720 | 5,484 | 8,192 | 4 | 8,136.346 | 640.325 |
| Bytes Written <file=./Process_Output/TurbRsdl.dat> | 4,271 | 72 | 224 | 57 | 59.319 | 19.544 |
| Bytes Written <file=./Process_Output/Solver.out> | 2,039 | 13 | 797 | 43 | 156.846 | 191.359 |
| Bytes Written <file=./Field_Solutions/Solution_History/NACA0012_LargeGrid_00010.Sln> | 4,356,976 | 534 | 8,192 | 4 | 8,159.131 | 501.319 |
| Bytes Written <file=./Field_Solutions/Solution_History/NACA0012_LargeGrid_00005.Sln> | 4,356,976 | 534 | 8,192 | 4 | 8,159.131 | 501.319 |
| Bytes Written <file=./Field_Solutions/NACA0012_LargeGrid.Sln> | 4,356,976 | 534 | 8,192 | 4 | 8,159.131 | 501.319 |
| Bytes Written <file=./Body_Pressure/NACA0012_LargeGrid_00010_body.Prs> | 65,986 | 9 | 8,190 | 1,300 | 7,331.778 | 2,133.204 |
| Bytes Written <file=./Body_Pressure/NACA0012_LargeGrid_00005_body.Prs> | 65,986 | 9 | 8,190 | 1,300 | 7,331.778 | 2,133.204 |
| Bytes Written <file=./Body_Pressure/FrcMnt.out> | 1,497 | 3 | 1,185 | 108 | 499 | 486.656 |
| Bytes Written | 147,107,546 | 18,550 | 8,192 | 1 | 7,930.326 | 1,420.552 |

TAU – Context Events

TAU: ParaProf: Context Events for thread: n,c,t, 1,0,0 – samarc_obe_4p_iomem_cp.ppk

| Name | Total | MeanValue | NumSamples | MinValue | MaxValue | Std. Dev. |
|---|-----------|------------|------------|----------|-----------|-------------|
| ▼ .TAU application | | | | | | |
| ▶ read() | | | | | | |
| ▶ fopen64() | | | | | | |
| ▶ fclose() | | | | | | |
| ▼ OurMain() | | | | | | |
| malloc size | 25,235 | 1,097.174 | 23 | 11 | 12,032 | 2,851.143 |
| free size | 22,707 | 1,746.692 | 13 | 11 | 12,032 | 3,660.642 |
| ▼ OurMain [{{wrapper.py}}{3}] | | | | | | |
| ▶ read() | | | | | | |
| malloc size | 3,877 | 323.083 | 12 | 32 | 981 | 252.72 |
| free size | 1,536 | 219.429 | 7 | 32 | 464 | 148.122 |
| ▶ fopen64() | | | | | | |
| ▶ fclose() | | | | | | |
| ▼ <module> [{{obe.py}}{8}] | | | | | | |
| ▼ writeRestartData [{{samarcInterface.py}}{145}] | | | | | | |
| ▼ samarcWriteRestartData | | | | | | |
| ▼ write() | | | | | | |
| WRITE Bandwidth (MB/s) <file="samarc/restore.00002/nodes.00004/proc.00001"> | | 74.565 | 117 | 0 | 2,156.889 | 246.386 |
| WRITE Bandwidth (MB/s) <file="samarc/restore.00001/nodes.00004/proc.00001"> | | 77.594 | 117 | 0 | 1,941.2 | 228.366 |
| WRITE Bandwidth (MB/s) | | 76.08 | 234 | 0 | 2,156.889 | 237.551 |
| Bytes Written <file="samarc/restore.00002/nodes.00004/proc.00001"> | 2,097,552 | 17,927.795 | 117 | 1 | 1,048,576 | 133,362.946 |
| Bytes Written <file="samarc/restore.00001/nodes.00004/proc.00001"> | 2,097,552 | 17,927.795 | 117 | 1 | 1,048,576 | 133,362.946 |
| Bytes Written | 4,195,104 | 17,927.795 | 234 | 1 | 1,048,576 | 133,362.946 |
| ▶ open64() | | | | | | |

Write bandwidth per file

Bytes written to each file

Interval and Atomic Events in TAU

```

xterm
NODE 0;CONTEXT 0;THREAD 0:
-----
%Time      Exclusive    Inclusive    #Call    #Subrs    Inclusive Name
           msec      total msec
-----
100.0      0.187        1,105        1         44        1105659 int main(int, char **) C
93.2       1,030        1,030        1          0        1030654 MPI_Init()
5.9        0.879        65           40        320       1637 void func(int, int) C
4.6         51          51           40         0        1277 MPI_Barrier()
1.2         13          13           120        0         111 MPI_Recv()
0.8         9           9            1          0        9328 MPI_Finalize()
0.0        0.137        0.137        120        0         1 MPI_Send()
0.0        0.086        0.086        40         0         2 MPI_Bcast()
0.0        0.002        0.002        1          0         2 MPI_Comm_size()
0.0        0.001        0.001        1          0         1 MPI_Comm_rank()
-----
  
```

Interval events
e.g., routines
(start/stop) show
duration

```

USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 0
-----
NumSamples  MaxValue  MinValue  MeanValue  Std. Dev.  Event Name
-----
365 5.138E+04 44.39 3.09E+04 1.234E+04 Heap Memory Used (KB) : Entry
365 5.138E+04 2064 3.115E+04 1.21E+04 Heap Memory Used (KB) : Exit
40 40 40 40 0 Message size for broadcast
-----
27.1
  
```

Atomic events
(triggered with
value) show
extent of variation
(min/max/mean)

```

% export TAU_CALLPATH_DEPTH=0
% export TAU_TRACK_HEAP=1
  
```

Atomic Events, Context Events

```
xterm
```

| %Time | Exclusive msec | Inclusive total msec | #Call | #Subrs | Inclusive usec/call | Name |
|-------|----------------|----------------------|-------|--------|---------------------|--------------------------|
| 100.0 | 0.253 | 1.106 | 1 | 44 | 1106701 | int main(int, char **) C |
| 93.2 | 1,031 | 1,031 | 1 | 0 | 1031311 | MPI_Init() |
| 6.0 | 1 | 66 | 40 | 320 | 1650 | void func(int, int) C |
| 5.7 | 63 | 63 | 40 | 0 | 1588 | MPI_Barrier() |
| 0.8 | 9 | 9 | 1 | 0 | 9119 | MPI_Finalize() |
| 0.1 | 1 | 1 | 120 | 0 | 10 | MPI_Recv() |
| 0.0 | 0.141 | 0.141 | 120 | 0 | 1 | MPI_Send() |
| 0.0 | 0.085 | 0.085 | 40 | 0 | 2 | MPI_Bcast() |
| 0.0 | 0.001 | 0.001 | 1 | 0 | 1 | MPI_Comm_size() |
| 0.0 | 0 | 0 | 1 | 0 | 0 | MPI_Comm_rank() |

Atomic events

USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 0

| NumSamples | MaxValue | MinValue | MeanValue | Std. Dev. | Event Name |
|------------|-----------|-----------|-----------|-----------|--|
| 40 | 40 | 40 | 40 | 0 | Message size for broadcast |
| 365 | 5.139E+04 | 44.39 | 3.091E+04 | 1.234E+04 | Heap Memory Used (KB) : Entry |
| 40 | 5.139E+04 | 3097 | 3.114E+04 | 1.227E+04 | Heap Memory Used (KB) : Entry : MPI_Barrier() |
| 40 | 5.139E+04 | 1.13E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : MPI_Bcast() |
| 1 | 2067 | 2067 | 2067 | 0 | Heap Memory Used (KB) : Entry : MPI_Comm_rank() |
| 1 | 2066 | 2066 | 2066 | 0 | Heap Memory Used (KB) : Entry : MPI_Comm_size() |
| 1 | 5.139E+04 | 5.139E+04 | 5.139E+04 | 0.0006905 | Heap Memory Used (KB) : Entry : MPI_Finalize() |
| 1 | 57.56 | 57.56 | 57.56 | 0 | Heap Memory Used (KB) : Entry : MPI_Init() |
| 120 | 5.139E+04 | 1.13E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : MPI_Recv() |
| 120 | 5.139E+04 | 1.129E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : MPI_Send() |
| 1 | 44.39 | 44.39 | 44.39 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C |
| 40 | 5.036E+04 | 2068 | 3.011E+04 | 1.227E+04 | Heap Memory Used (KB) : Entry : void func(int, int) C |

Context events = atomic event + executing context

% export TAU_CALLPATH_DEPTH=1

% export TAU_TRACK_HEAP=1

Controls depth of executing context shown in profiles

Context Events (Default)

```

NODE 0:CONTEXT 0;THREAD 0:
-----
%Time   Exclusive   Inclusive   #Call   #Subrs   Inclusive   Name
        msec     total msec                usec/call
-----
100.0   0.357       1,114      1        44       1114040    int main(int, char **) C
92.6    1.031       1,031      1         0       1031066    MPI_Init()
6.7     72          74         40       320      1865      void func(int, int) C
0.7     8           8          1         0       8002      MPI_Finalize()
0.1     1           1          120        0       12        MPI_Recv()
0.1     0.608      0.608     40         0       15        MPI_Barrier()
0.0     0.136      0.136    120         0       1         MPI_Send()
0.0     0.095      0.095     40         0       2         MPI_Bcast()
0.0     0.001      0.001     1          0       1         MPI_Comm_size()
0.0     0          0          1          0       0         MPI_Comm_rank()
-----

```

USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 0

| NumSamples | MaxValue | MinValue | MeanValue | Std. Dev. | Event Name |
|------------|-----------|-----------|-----------|-----------|---|
| 365 | 5.139E+04 | 44.39 | 3.091E+04 | 1.234E+04 | Heap Memory Used (KB) : Entry |
| 1 | 44.39 | 44.39 | 44.39 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C |
| 1 | 2068 | 2068 | 2068 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C => MPI_Comm_rank() |
| 1 | 2066 | 2066 | 2066 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C => MPI_Comm_size() |
| 1 | 5.139E+04 | 5.139E+04 | 5.139E+04 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C => MPI_Finalize() |
| 1 | 57.58 | 57.58 | 57.58 | 0 | Heap Memory Used (KB) : Entry : int main(int, char **) C => MPI_Init() |
| 40 | 5.036E+04 | 2069 | 3.011E+04 | 1.228E+04 | Heap Memory Used (KB) : Entry : int main(int, char **) C => void func(int, int) C |
| 40 | 5.139E+04 | 3098 | 3.114E+04 | 1.227E+04 | Heap Memory Used (KB) : Entry : void func(int, int) C => MPI_Barrier() |
| 40 | 5.139E+04 | 1.13E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : void func(int, int) C => MPI_Bcast() |
| 120 | 5.139E+04 | 1.13E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : void func(int, int) C => MPI_Recv() |
| 120 | 5.139E+04 | 1.13E+04 | 3.134E+04 | 1.187E+04 | Heap Memory Used (KB) : Entry : void func(int, int) C => MPI_Send() |
| 365 | 5.139E+04 | 2065 | 3.116E+04 | 1.21E+04 | Heap Memory Used (KB) : Exit |

```

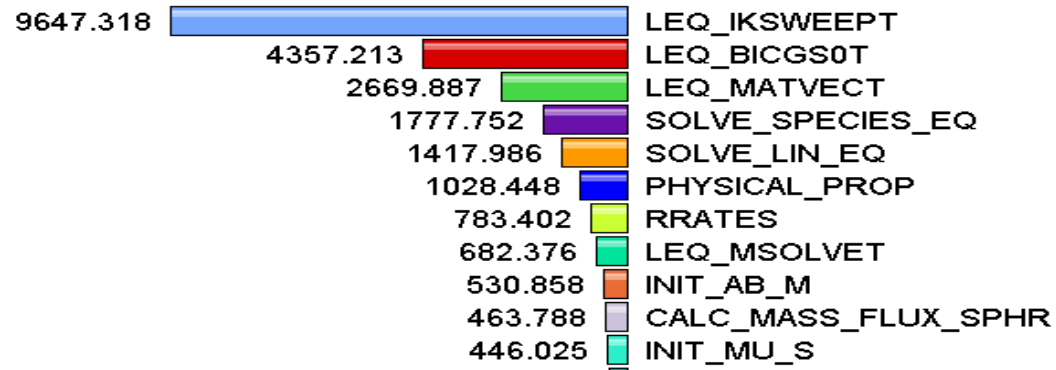
% export TAU_CALLPATH_DEPTH=2
% export TAU_TRACK_HEAP=1

```

Context event 3.7 1%

=atomic event + executing context

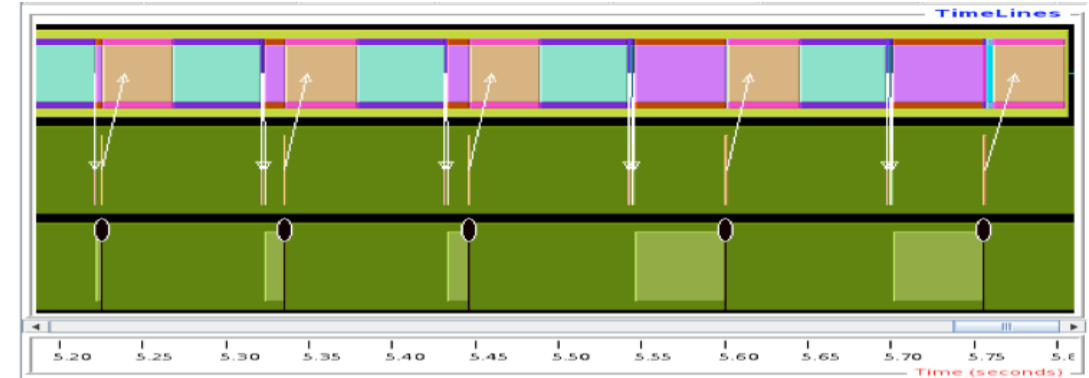
Profiling



Shows

how much time was spent in each routine

Tracing



Shows

when events take place on a timeline

Types of Performance Profiles

Flat profiles

- Metric (e.g., time) spent in an event
- Exclusive/inclusive, # of calls, child calls, ...

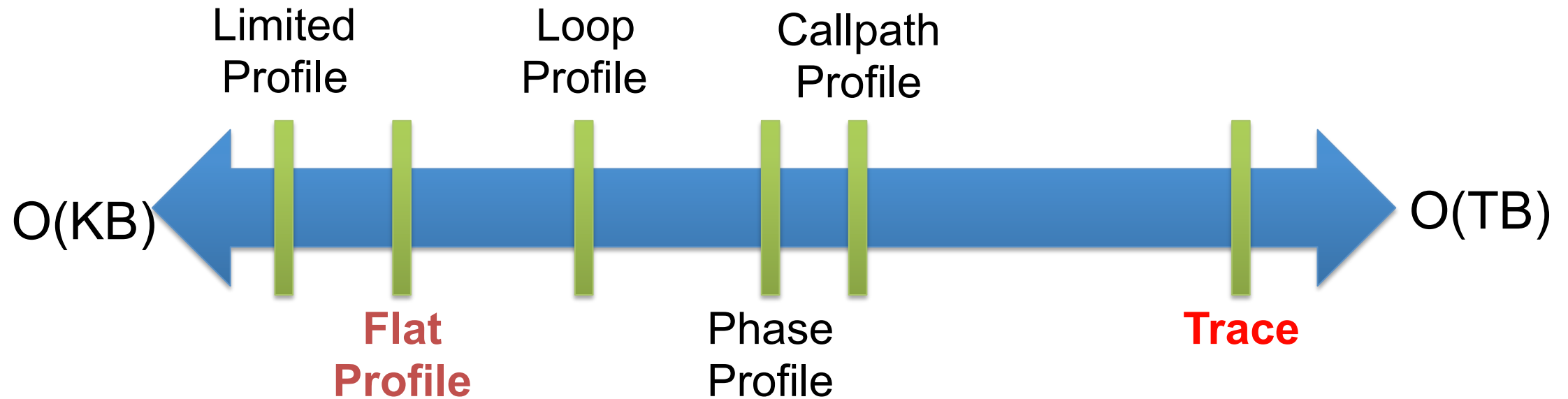
Callpath profiles

- Time spent along a calling path (edges in callgraph)
- “*main=> f1 => f2 => MPI_Send*”
- Set the **TAU_CALLPATH_DEPTH** environment variable

Phase profiles

- Flat profiles under a phase (nested phases allowed)
- Default “main” phase
- Supports static or dynamic (e.g. per-iteration) phases

How much data do you want?



All levels support multiple metrics/counters

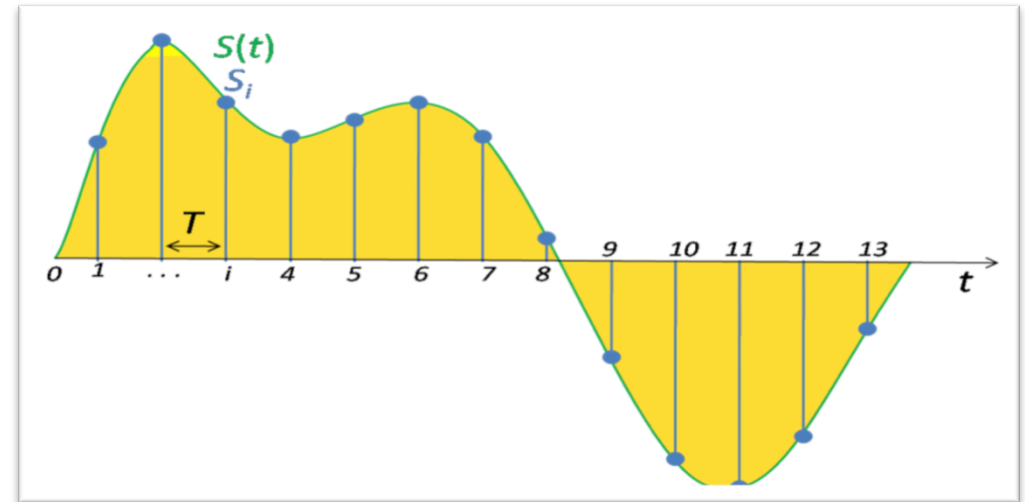
Performance Data Measurement

Direct via Probes

```
call TAU_START('potential')  
// code  
call TAU_STOP('potential')
```

- Exact measurement
- Fine-grain control
- Calls inserted into code

Indirect via Sampling



- No code modification
- Minimal effort
- Relies on debug symbols (-g option)
- TAU_SAMPLING=1

Examples

HPCLinux OVA: VirtualBox

Install VirtualBox from the USB Stick (dmg/exe file)

File -> Import Appliance -> Click browse -> HPCLinux -> <LITE>.ova -> Finish

Click on the appliance in VirtualBox -> Run

Username: livetau

Password:

% which paraprof

Step 1: Log-in to the Cluster (ri.cse.ohio-state.edu)

Linux/OS X (Mac)

Open your favorite Terminal

```
ssh -Y <username>@ri.cse.ohio-state.edu
```

Enter password

Example:

```
$ ssh -Y ritutXX@ri.cse.ohio-state.edu (replace XX with appropriate number from handout)
```

Enter Password:

```
$ ls -l workshop
```

```
total 76
```

```
drwxr-xr-x 3 root root 4096 Jul 25 11:24 3Dstencil
drwxr-xr-x 2 root root 4096 Jul 25 11:24 cthreads
drwxr-xr-x 2 root root 4096 Jul 25 11:24 debug
-rw-r--r-- 1 root root 706 Jul 25 11:24 handson.txt
drwxr-xr-x 2 root root 4096 Jul 25 11:24 hdf5_wrap
drwxr-xr-x 2 root root 4096 Jul 25 11:24 job_submission
drwxr-xr-x 2 root root 4096 Jul 25 11:24 manual
drwxr-xr-x 3 root root 4096 Jul 25 11:24 matmult
drwxr-xr-x 2 root root 4096 Jul 25 11:24 memoryleakdetect
drwxr-xr-x 4 root root 4096 Jul 25 11:24 miniGhost_ref
drwxr-xr-x 2 root root 4096 Jul 25 11:24 mm
drwxr-xr-x 2 root root 4096 Jul 25 11:24 mpic++
drwxr-xr-x 2 root root 4096 Jul 25 11:24 mpiposix
drwxr-xr-x 15 root root 4096 Jul 25 11:24 NPB3.1
drwxr-xr-x 2 root root 4096 Jul 25 11:24 opari
drwxr-xr-x 2 root root 4096 Jul 25 11:24 papi
drwxr-xr-x 4 root root 4096 Jul 25 11:24 py-c++-f90
-rw-r--r-- 1 root root 1707 Jul 25 11:24 README
drwxr-xr-x 2 root root 4096 Jul 25 11:24 sweep3d
```

Windows

Download Putty

- <https://www.putty.org/>

How to?

- <https://mediateple.net/community/products/dv/204404604/using-ssh-in-putty->
- <https://www.ssh.com/ssh/putty/windows/>

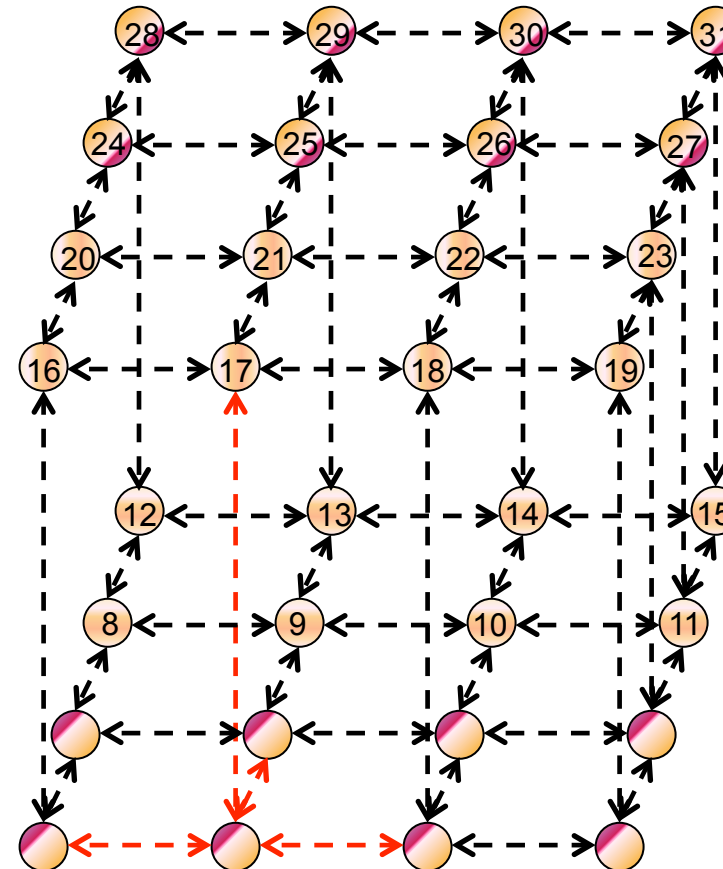
Driving Example (3D Stencil)

3D Stencil benchmark

- Each process talks to at most **six** neighbors
- Two in each Cartesian dimension
 - X-right, X-left
 - Y-right, Y-left
 - Z-right, Z-left
- Repeat same communication pattern for multiple iterations



3D Stencil communication pattern for a 32 process job scheduled on 4 nodes



3Dstencil on RI (OSU)

```
cd ~/workshop/3Dstencil  
sbatch -N 16 -p batch ./demo.sh  
ls *.ppk
```

Copy ppk files to your laptop and use VirtualBox image

```
% paraprof *.ppk &
```

Example Codes On Stampede2.tacc.utexas.edu

- `% cp ~tg457572/pkgs/workshop.tgz .`
- `% tar zxf workshop.tgz`
- `% source ~tg457572/tau.bashrc`
- `% iddev -m 50 -r pearc-tau`
(requests a 50 minute interactive node with reservation name pearc-tau)
- `% cd workshop/NPB3.1`
- `% make suite`
- `% cd bin`
- `% mpirun -np 64 ./lu.A.64`
- `% tau_pebil_rewrite lu.A.64 lu.i`
- `% mpirun -np 64 ./lu.i`
- `% pprof ; paraprof`

Simplifying the use of TAU!

Uninstrumented code:

- % mpif90 -g -O3 matmult.f90
- % mpirun -np 16 ./a.out

With TAU:

- % mpirun -np 16 **tau_exec** ./a.out
- % paraprof
- For more Information at the statement level:
- % mpirun -np 16 tau_exec **-ebs** ./a.out (or use TAU_SAMPLING=1)
- To rewrite the binary to instrument individual functions (using PEBIL):
- % tau_pebil_rewrite a.out a.inst; mpirun -np 16 ./a.inst (beta)
- % pprof -a | more
- % paraprof (GUI)

TAU for Heterogeneous Measurement

Multiple performance perspectives

Integrate Host-GPU support in TAU measurement framework

- Enable use of each measurement approach
- Include use of PAPI and CUPTI
- Provide profiling and tracing support

Tutorial

- Use TAU library wrapping of libraries
- Use `tau_exec` to work with binaries
 - % `./a.out` (uninstrumented)
 - % `tau_exec -T <configuration tags> -cupti ./a.out`
 - % `paraprof`

TAU Execution Command (tau_exec)

Uninstrumented execution

- % mpirun -np 256 ./a.out

Track GPU operations

- % mpirun -np 256 tau_exec -cupti ./a.out
- % mpirun -np 256 tau_exec -cupti -um ./a.out (for Unified Memory)
- % mpirun -np 256 tau_exec -opencl ./a.out
- % mpirun -np 256 tau_exec -openacc ./a.out

Track MPI performance

- % mpirun -np 256 tau_exec ./a.out

Track OpenMP, and MPI performance (MPI enabled by default)

- % export TAU_OMPT_SUPPORT_LEVEL=full;
% export TAU_OMPT_RESOLVE_ADDRESS_EAGERLY=1
- % mpirun -np 256 tau_exec -T ompt,tr6,mpi -ompt ./a.out

Track memory operations

- % export TAU_TRACK_MEMORY_LEAKS=1
- % mpirun -np 256 tau_exec -memory_debug ./a.out (bounds check)

Use event based sampling (compile with -g)

- % mpirun -np 256 tau_exec -ebs ./a.out
- Also -ebs_source=<PAPI_COUNTER> -ebs_period=<overflow_count>

Using TAU

TAU supports several measurement and thread options

Phase profiling, profiling with hardware counters (papi), MPI library, CUDA, Beacon (backplane for event notification – online monitoring), PDT (automatic source instrumentation) ...

Each measurement configuration of TAU corresponds to a unique stub makefile and library that is generated when you configure it

To instrument source code automatically using PDT

Choose an appropriate TAU stub makefile in <arch>/lib:

```
% source ~tg457572/tau.bashrc
```

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-mvapich2-icpc-mpi-pdt
```

```
% export TAU_OPTIONS='-optVerbose ...' (see tau_compiler.sh )
```

Use tau_f90.sh, tau_cxx.sh, tau_upc.sh, or tau_cc.sh as F90, C++, UPC, or C compilers respectively:

```
% mpif90 foo.f90      changes to
```

```
% tau_f90.sh foo.f90
```

Set runtime environment variables, execute application and analyze performance data:

```
% pprof (for text based profile display)
```

```
% paraprof (for GUI)
```

Choosing TAU_MAKEFILE

```
% source ~tg457572/tau.bashrc
% ls $TAU/Makefile.*
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-mvapich2-icpc-mpi-pdt
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-mvapich2-icpc-mpi-pthread-pdt
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-mvapich2-icpc-ompt-mpi-pdt-openmp
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-mvapich2-icpc-papi-mpi-pdt-mpit
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-mvapich2-icpc-papi-ompt-mpi-pdt-openmp
/home1/00494/tg457572/pkgs/tau_latest/x86_64/lib/Makefile.tau-pdt
```

For an MPI+F90 application with MPI, you may choose
Makefile.tau-mvapich2-icpc-mpi-pdt

- Supports MPI instrumentation, papi, and PDT for automatic source instrumentation

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-mvapich2-icpc-mpi-pdt
% tau_f90.sh matrix.f90 -o matrix
```

OR with build systems:

```
% make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh
% cmake -DCMAKE_Fortran_COMPILER=tau_f90.sh
    -DCMAKE_C_COMPILER=tau_cc.sh -
DCMAKE_CXX_COMPILER=tau_cxx.sh
% mpirun -np 1024 ./matrix
% paraprof
```

Configuration tags for tau_exec

```
% ./configure -pdt=<dir> -mpi -papi=<dir>; make install
```

Creates in \$TAU:

```
Makefile.tau-papi-mpi-pdt (Configuration parameters in stub makefile)  
shared-papi-mpi-pdt/libTAU.so
```

```
% ./configure -pdt=<dir> -mpi; make install creates
```

```
Makefile.tau-mpi-pdt  
shared-mpi-pdt/libTAU.so
```

To explicitly choose preloading of shared-<options>/libTAU.so change:

```
% mpirun -np 256 ./a.out to
```

```
% mpirun -np 256 tau_exec -T <comma_separated_options> ./a.out
```

```
% mpirun -np 256 tau_exec -T papi,mpi,pdt ./a.out
```

Preloads \$TAU/shared-papi-mpi-pdt/libTAU.so

```
% mpirun -np 256 tau_exec -T papi ./a.out
```

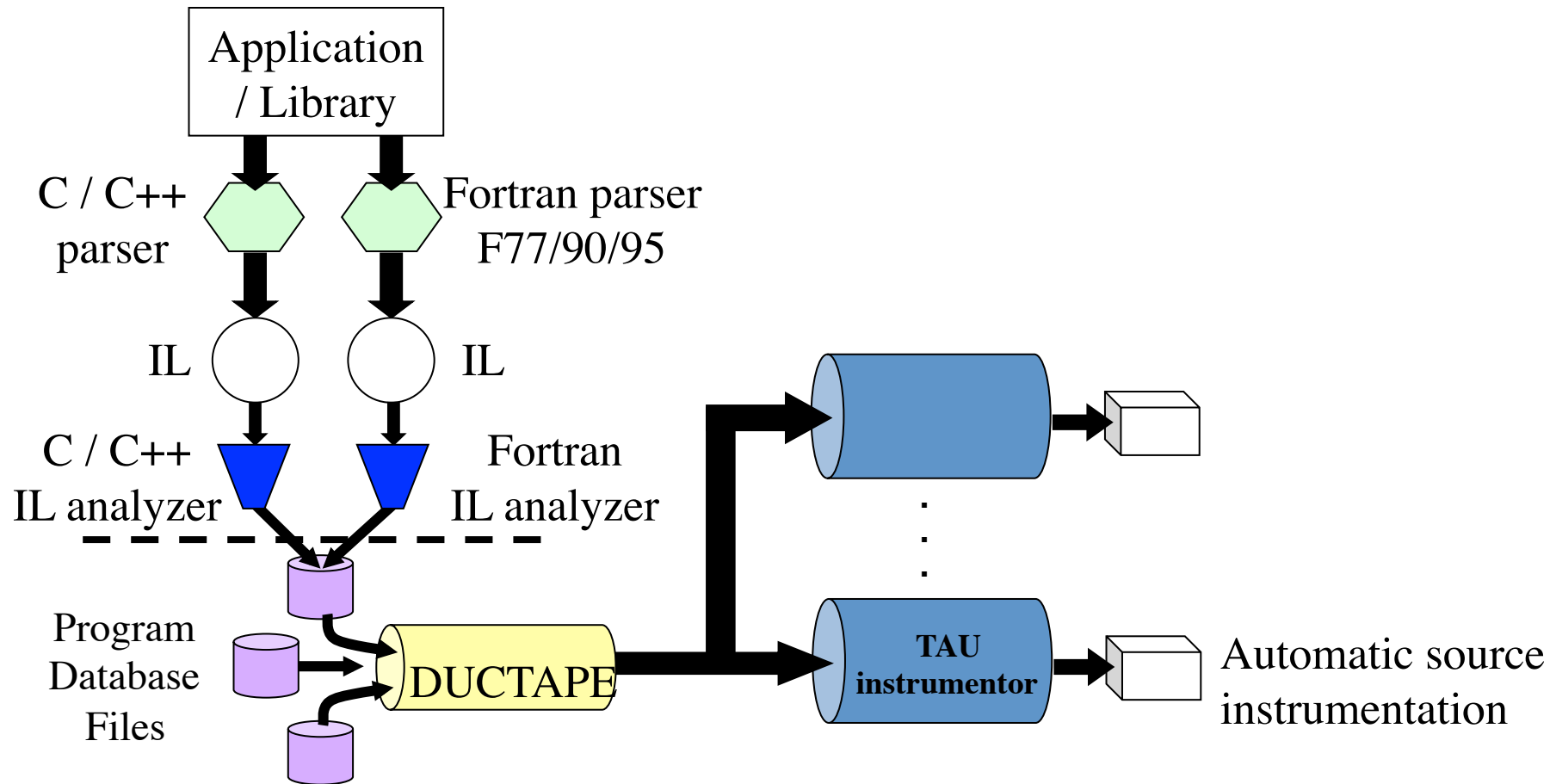
Preloads \$TAU/shared-papi-mpi-pdt/libTAU.so by matching.

```
% mpirun -np 256 tau_exec -T papi,mpi,pdt -s ./a.out
```

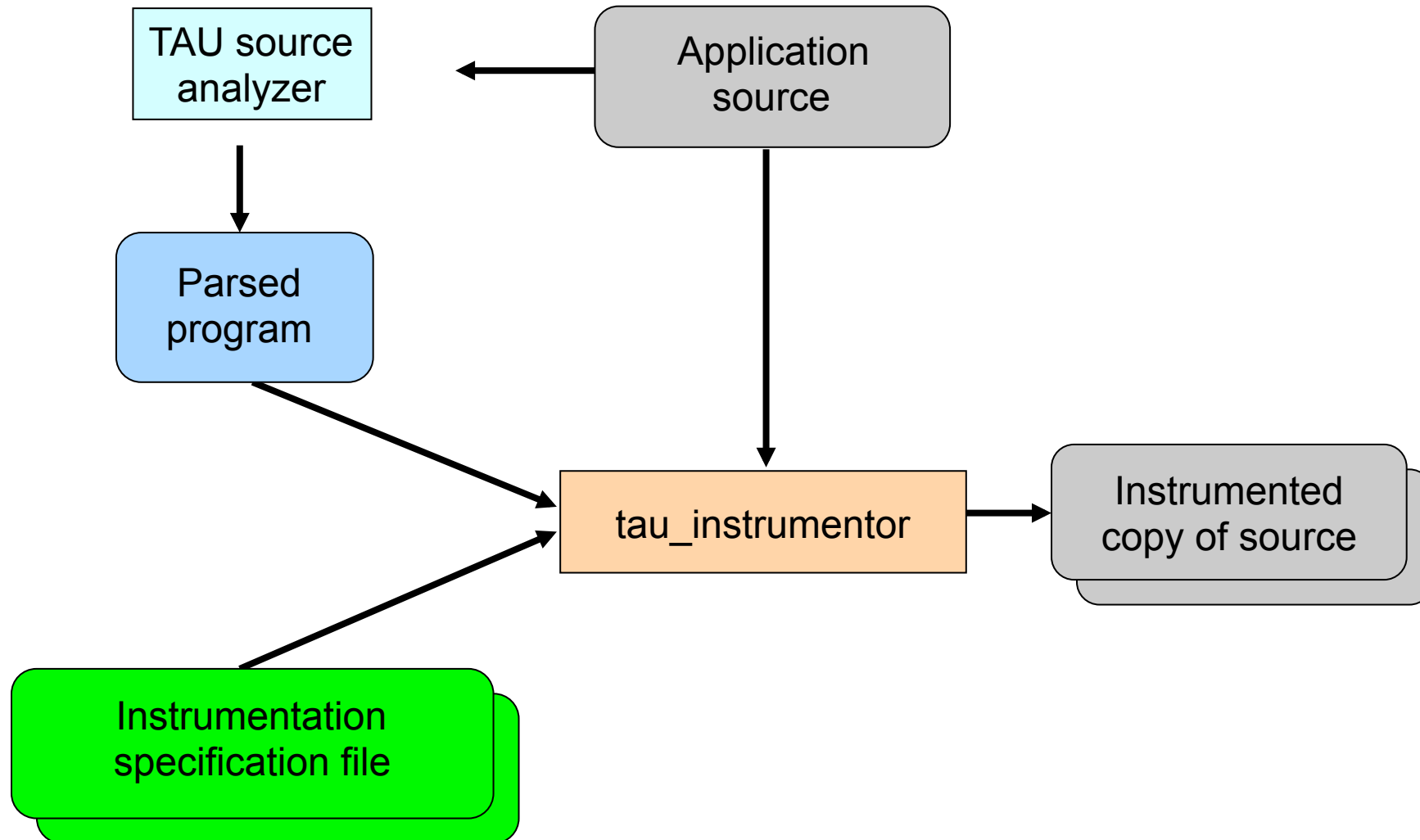
Does not execute the program. Just displays the library that it will preload if executed without the **-s** option.

NOTE: -mpi configuration is selected by default. Use **-T serial** for Sequential programs.

TAU's Static Analysis System: Program Database Toolkit (PDT)



Automatic Source Instrumentation using PDT



Automatic Instrumentation

- **Use TAU's compiler wrappers**
 - Simply replace `CXX` with `tau_cxx.sh`, etc.
 - Automatically instruments source code, links with TAU libraries.
- **Use `tau_cc.sh` for C, `tau_f90.sh` for Fortran, `tau_upc.sh` for UPC, etc.**

Before

```
% cat Makefile
CXX = mpicxx
F90 = mpif90
CXXFLAGS =
LIBS = -lm
OBJS = f1.o f2.o f3.o ... fn.o

app: $(OBJS)
    $(CXX) $(LDFLAGS) $(OBJS) -o $@
    $(LIBS)
.cpp.o:
    $(CXX) $(CXXFLAGS) -c $<

% make
```

After

```
% cat Makefile
CXX = tau_cxx.sh
F90 = tau_f90.sh
CXXFLAGS =
LIBS = -lm
OBJS = f1.o f2.o f3.o ... fn.o

app: $(OBJS)
    $(CXX) $(LDFLAGS) $(OBJS) -o $@
    $(LIBS)
.cpp.o:
    $(CXX) $(CXXFLAGS) -c $<

% export TAU_MAKEFILE=
    $TAU/Makefile.tau-papi-mpi-pdt
% make
```


Selective Instrumentation File: Compile time, Runtime

```
% export TAU_OPTIONS='-optTauSelectFile=select.tau ...'  
% cat select.tau  
BEGIN_INCLUDE_LIST  
int main#  
int dgemv#  
END_INCLUDE_LIST  
BEGIN_FILE_INCLUDE_LIST  
Main.c  
Blas/*.f77  
END_FILE_INCLUDE_LIST  
# replace include with exclude list  
  
BEGIN_INSTRUMENT_SECTION  
loops routine="foo"  
loops routine="int main#"  
END_INSTRUMENT_SECTION  
# It can be used at compile time or at runtime:  
% export TAU_SELECT_FILE = select.tau
```

Installing and Configuring TAU

•Installing PDT:

- `wget tau.uoregon.edu/pdt_lite.tgz`
- `./configure --prefix=<dir>; make ; make install`

•Installing TAU:

- `wget tau.uoregon.edu/tau.tgz; tar xzf tau.tgz; cd tau-2.<ver>`
- `wget http://tau.uoregon.edu/ext.tgz`
- `./configure --mpi -bfd=download -pdt=<dir> -papi=<dir> ...`
- `make install`

•Using TAU:

- `export TAU_MAKEFILE=<taudir>/x86_64/
lib/Makefile.tau-<TAGS>`
- `% export TAU_OPTIONS='-optTauSelectFile=select.tau'`
- `make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh`

INSTALLING TAU on Laptops

Installing TAU under Mac OS X:

- Download Java
- <http://tau.uoregon.edu/java.dmg>
- Install java.dmg
- `wget http://tau.uoregon.edu/tau.dmg`
- Install tau.dmg

Installing TAU under Windows

- <http://tau.uoregon.edu/tau.exe>

Installing TAU under Linux

- <http://tau.uoregon.edu/tau.tgz>
- `./configure; make install`
- `export PATH=<taudir>/x86_64/bin:$PATH`

NPB-MZ-MPI Suite

The NAS Parallel Benchmark suite (MPI+OpenMP version)

- Available from:

<http://www.nas.nasa.gov/Software/NPB>

- 3 benchmarks in Fortran77
- Configurable for various sizes & classes

Move into the NPB3.3-MZ-MPI root directory

```
% ls
bin/      common/  jobscript/  Makefile  README.install  SP-MZ/
BT-MZ/   config/  LU-MZ/      README    README.tutorial  sys/
```

Subdirectories contain source code for each benchmark

- plus additional configuration and common code

The provided distribution has already been configured for the tutorial, such that it's ready to “make” one or more of the benchmarks and install them into a (tool-specific) “bin” subdirectory

NPB-MZ-MPI / BT (Block Tridiagonal Solver)

What does it do?

- Solves a discretized version of the unsteady, compressible Navier-Stokes equations in three spatial dimensions
- Performs 200 time-steps on a regular 3-dimensional grid

Implemented in 20 or so Fortran77 source modules

Uses MPI & OpenMP in combination

- 16 processes each with 4 threads should be reasonable
- bt-mz.B.16 should take around 1 minute

NPB-MZ-MPI / BT: config/make.def

```
#           SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#-----
#-----
# Configured for generic MPI with GCC compiler
#-----
#OPENMP   = -fopenmp           # GCC compiler
OPENMP    = -qopenmp -extend-source      # Intel compiler
...
#-----
# The Fortran compiler used for MPI programs
#-----
F77 = mpif90 # Intel compiler
# Alternative variant to perform instrumentation
...

```

Default (no instrumentation)

Building an NPB-MZ-MPI Benchmark

```
% make
=====
=      NAS PARALLEL BENCHMARKS 3.3      =
=      MPI+OpenMP Multi-Zone Versions   =
=      F77                                =
=====

To make a NAS multi-zone benchmark type

    make <benchmark-name> CLASS=<class> NPROCS=<nprocs>

where <benchmark-name> is "bt-mz", "lu-mz", or "sp-mz"
      <class>           is "S", "W", "A" through "F"
      <nprocs>         is number of processes

[...]

*****
* Custom build configuration is specified in config/make.def *
* Suggested tutorial exercise configuration for HPC systems: *
*      make bt-mz CLASS=C NPROCS=8                          *
*****
```

Type “make” for
instructions
make **suite**

TAU Source Instrumentation

Edit `config/make.def` to adjust build configuration

- Uncomment specification of compiler/linker: `F77 = tau_f77.sh` or use `make F77=tau_f77.sh`

Make clean and build new tool-specific executable

Change to the directory containing the new executable before running it with the desired tool configuration

tau_exec

```
$ tau_exec
```

```
Usage: tau_exec [options] [--] <exe> <exe options>
```

Options:

```
-v          Verbose mode
-s          Show what will be done but don't actually do anything (dryrun)
-qsub       Use qsub mode (BG/P only, see below)
-io         Track I/O
-memory     Track memory allocation/deallocation
-memory_debug Enable memory debugger
-cuda       Track GPU events via CUDA
-cupti      Track GPU events via CUPTI (Also see env. variable TAU_CUPTI_API)
-opencl     Track GPU events via OpenCL
-openacc    Track GPU events via OpenACC (currently PGI only)
-ompt       Track OpenMP events via OMPT interface
-armci      Track ARMCI events via PARMCI
-ebs        Enable event-based sampling
-ebs_period=<count> Sampling period (default 1000)
-ebs_source=<counter> Counter (default itimer)
-um         Enable Unified Memory events via CUPTI
-T <DISABLE,GNU,ICPC,MPI,OMPT,OPENMP,PAPI,PDT,PROFILE,PTHREAD,SCOREP,SERIAL> : Specify TAU tags
-loadlib=<file.so> : Specify additional load library
-XrunTAUsh-<options> : Specify TAU library directly
-gdb        Run program in the gdb debugger
```

Notes:

```
Defaults if unspecified: -T MPI
MPI is assumed unless SERIAL is specified
```

**tau_exec preloads
the TAU wrapper
libraries and
performs
measurements.**

No need to recompile the application!

tau_exec Example (continued)

Example:

```
mpirun -np 2 tau_exec -T icpc,ompt,mpi -ompt ./a.out
mpirun -np 2 tau_exec -io ./a.out
```

Example - event-based sampling with samples taken every 1,000,000 FP instructions

```
mpirun -np 8 tau_exec -ebs -ebs_period=1000000 -ebs_source=PAPI_FP_INS ./ring
```

Examples - GPU:

```
tau_exec -T serial,cupti -cupti ./matmult (Preferred for CUDA 4.1 or later)
tau_exec -openacc ./a.out
tau_exec -T serial -opencl ./a.out (OPENCL)
mpirun -np 2 tau_exec -T mpi,cupti,papi -cupti -um ./a.out (Unified Virtual Memory in CUDA 6.0+)
```

qsub mode (IBM BG/Q only):

Original:

```
qsub -n 1 --mode smp -t 10 ./a.out
```

With TAU:

```
tau_exec -qsub -io -memory -- qsub -n 1 ... -t 10 ./a.out
```

Memory Debugging:

-memory option:

Tracks heap allocation/deallocation and memory leaks.

-memory_debug option:

Detects memory leaks, checks for invalid alignment, and checks for array overflow. This is exactly like setting TAU_TRACK_MEMORY_LEAKS=1 and TAU_MEMDBG_PROTECT_ABOVE=1 and running with -memory

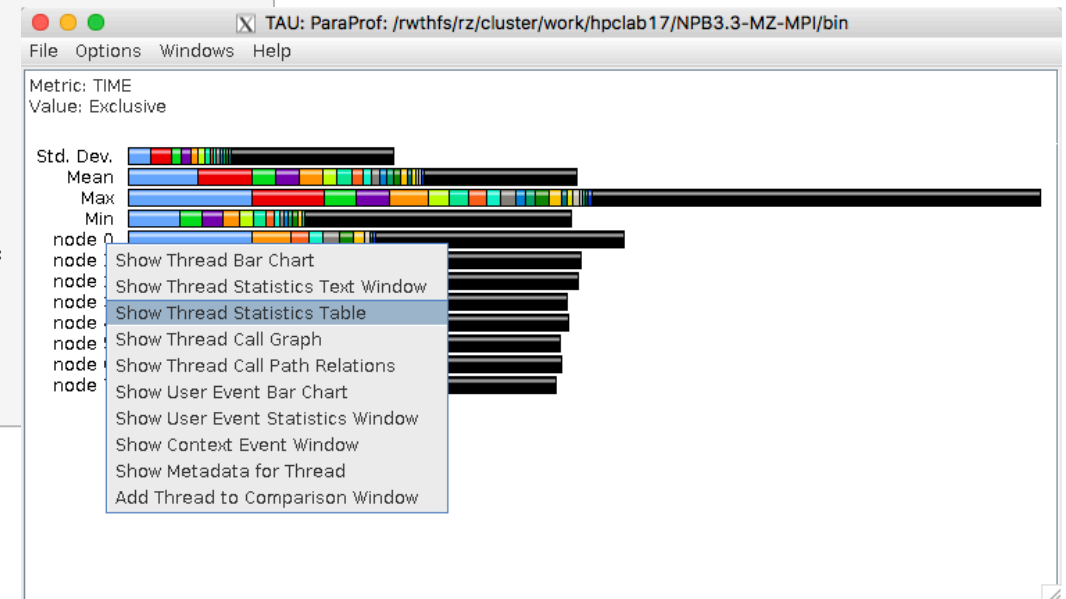
tau_exec can enable event based sampling while launching the executable using env **TAU_SAMPLING=1 or tau_exec **-ebs****

Event Based Sampling with TAU

Launch paraprof

```
% cd MZ-NPB3.3-MPI; cat README
% make clean;
% make suite
% cd bin
% idev -m 50 -r pearc-tau
% source ~tg457572/tau.bashrc
% export OMP_NUM_THREADS=4
% mpirun -np 4 tau_exec -T ompt -ebs ./bt-mz.B.4
% On head node:
% source ~tg457572/tau.bashrc
% paraprof
```

**Right Click on Node 0 and choose
Show Thread Statistics Table**



ParaProf

Click on Columns:
to sort by incl time

Open binvcrhs

Click on Sample


TAU: ParaProf: Statistics for: node 0 - /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/bin

| Name | Exclusive TIME | Inclusive TIME | Calls | Child Calls |
|--|----------------|----------------|-------|-------------|
| .TAU application | 9.167 | 9.167 | 1 | 2,432 |
| [CONTEXT] .TAU application | 0 | 9.019 | 901 | 0 |
| [SUMMARY] binvcrhs_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/ | 2.89 | 2.89 | 288 | 0 |
| [SUMMARY] matmul_sub_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT | 1.27 | 1.27 | 127 | 0 |
| [SUMMARY] x_solve_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/x | 1.16 | 1.16 | 116 | 0 |
| [SUMMARY] z_solve_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/z | 1.08 | 1.08 | 108 | 0 |
| [SUMMARY] y_solve_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/y | 1.08 | 1.08 | 108 | 0 |
| [SUMMARY] compute_rhs_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/B | 0.83 | 0.83 | 83 | 0 |
| [SUMMARY] matvec_sub_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT | 0.49 | 0.49 | 49 | 0 |
| [SUMMARY] lhsinit_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/in | 0.08 | 0.08 | 8 | 0 |
| [SAMPLE] add_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/add.f | 0.05 | 0.05 | 5 | 0 |
| [SUMMARY] binvrhs_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/ε | 0.04 | 0.04 | 4 | 0 |
| [SUMMARY] exact_solution_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/ | 0.02 | 0.02 | 2 | 0 |
| [SAMPLE] copy_x_face [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ | 0.01 | 0.01 | 1 | 0 |
| [SUMMARY] exact_rhs_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-M | 0.01 | 0.01 | 1 | 0 |
| [SAMPLE] initialize_ [{} /rwthfs/rz/cluster/work/hpclub17/NPB3.3-MZ-MPI/BT-MZ/in | 0.009 | 0.009 | 1 | 0 |
| MPI_Init_thread() | 0.155 | 0.155 | 1 | 0 |
| MPI_Finalize() | 0.022 | 0.022 | 1 | 0 |
| MPI_Waitall() | 0.018 | 0.018 | 804 | 0 |
| MPI_Irecv() | 0.004 | 0.004 | 804 | 0 |
| MPI_Isend() | 0.001 | 0.001 | 804 | 0 |
| MPI_Comm_split() | 0 | 0 | 1 | 0 |
| MPI_Bcast() | 0 | 0 | 9 | 0 |
| MPI_Reduce() | 0 | 0 | 3 | 0 |
| MPI_Barrier() | 0 | 0 | 2 | 0 |
| MPI_Comm_size() | 0 | 0 | 1 | 0 |
| MPI_Comm_rank() | 0 | 0 | 2 | 0 |

ParaProf

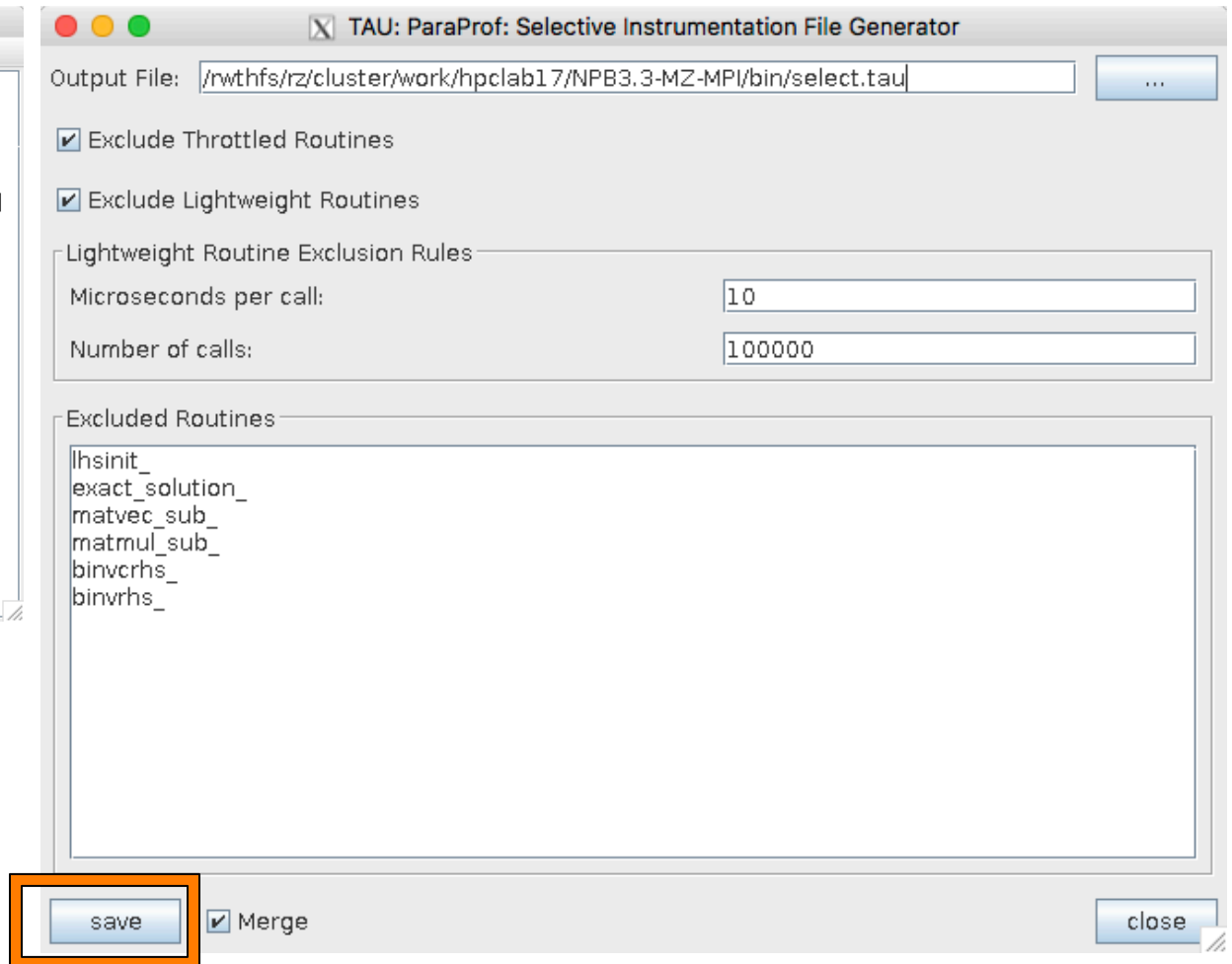
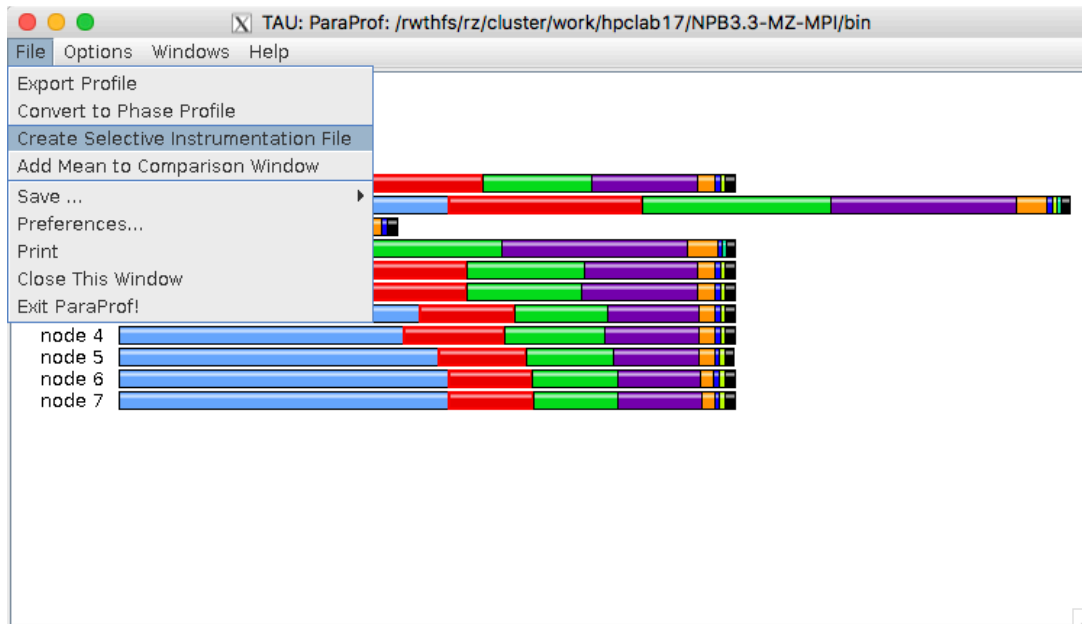
TAU: ParaProf: Statistics for: node 0 - /rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/bin

File Options Windows Help

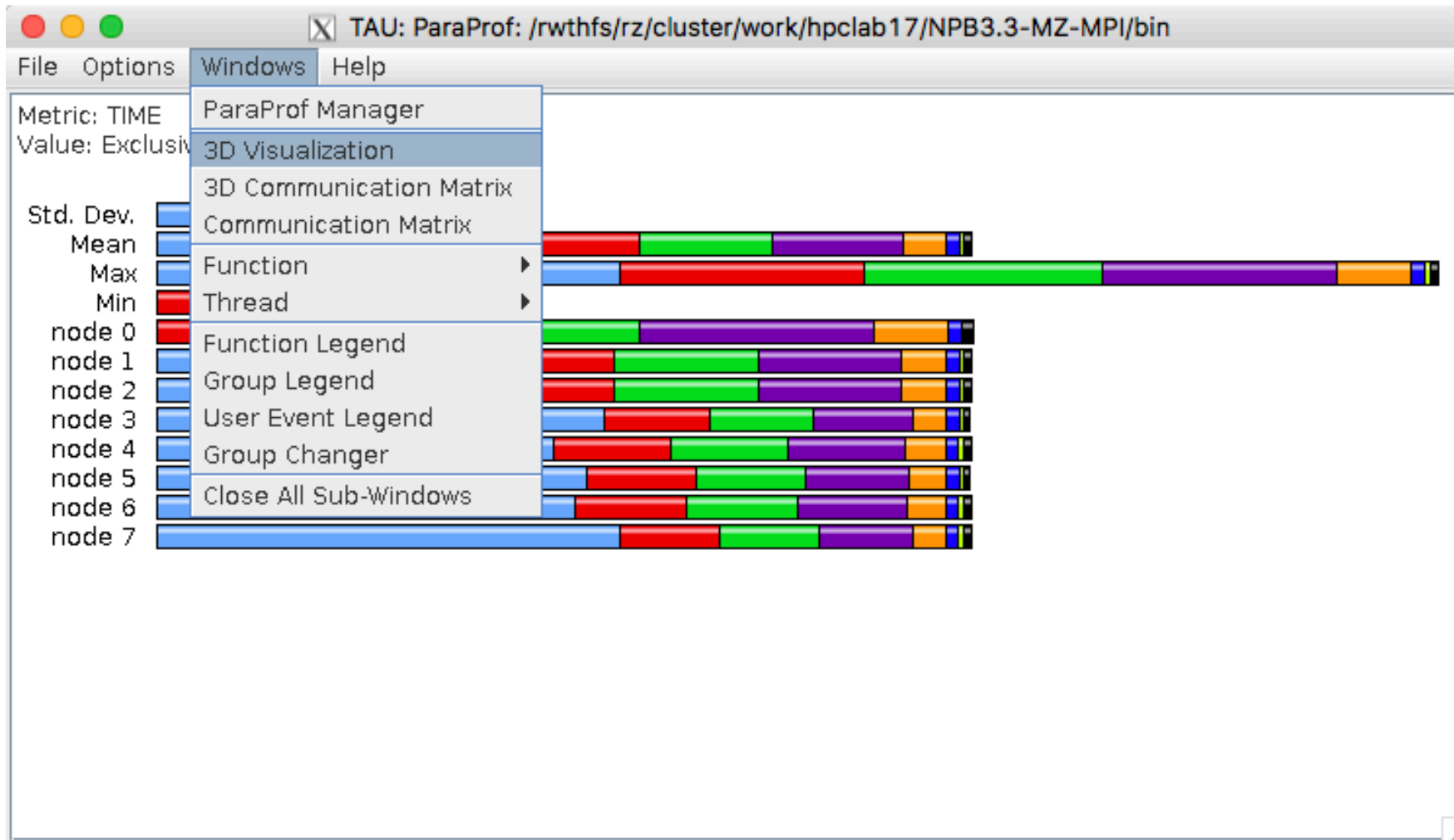


| Name | Exclusive TIME | Inclusive TIME ▾ | Calls | Child Calls |
|--|----------------|------------------|-------|-------------|
| .TAU application | 9.167 | 9.368 | 1 | 2,432 |
| [CONTEXT] .TAU application | 0 | 9.019 | 901 | 0 |
| [SUMMARY] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] | 2.89 | 2.89 | 288 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.14 | 0.14 | 14 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.09 | 0.09 | 9 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.09 | 0.09 | 9 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.06 | 0.06 | 6 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.06 | 0.06 | 6 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.06 | 0.06 | 6 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {228} | 0.06 | 0.06 | 6 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {244} | 0.05 | 0.05 | 5 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {332} | 0.05 | 0.05 | 5 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {275} | 0.05 | 0.05 | 5 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {331} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {445} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {254} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {314} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {343} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {403} | 0.04 | 0.04 | 4 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {389} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {415} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {247} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {300} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {309} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {444} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {468} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {242} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {407} | 0.03 | 0.03 | 3 | 0 |
| [SAMPLE] binvcrhs_ [/{rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}] {412} | 0.03 | 0.03 | 3 | 0 |

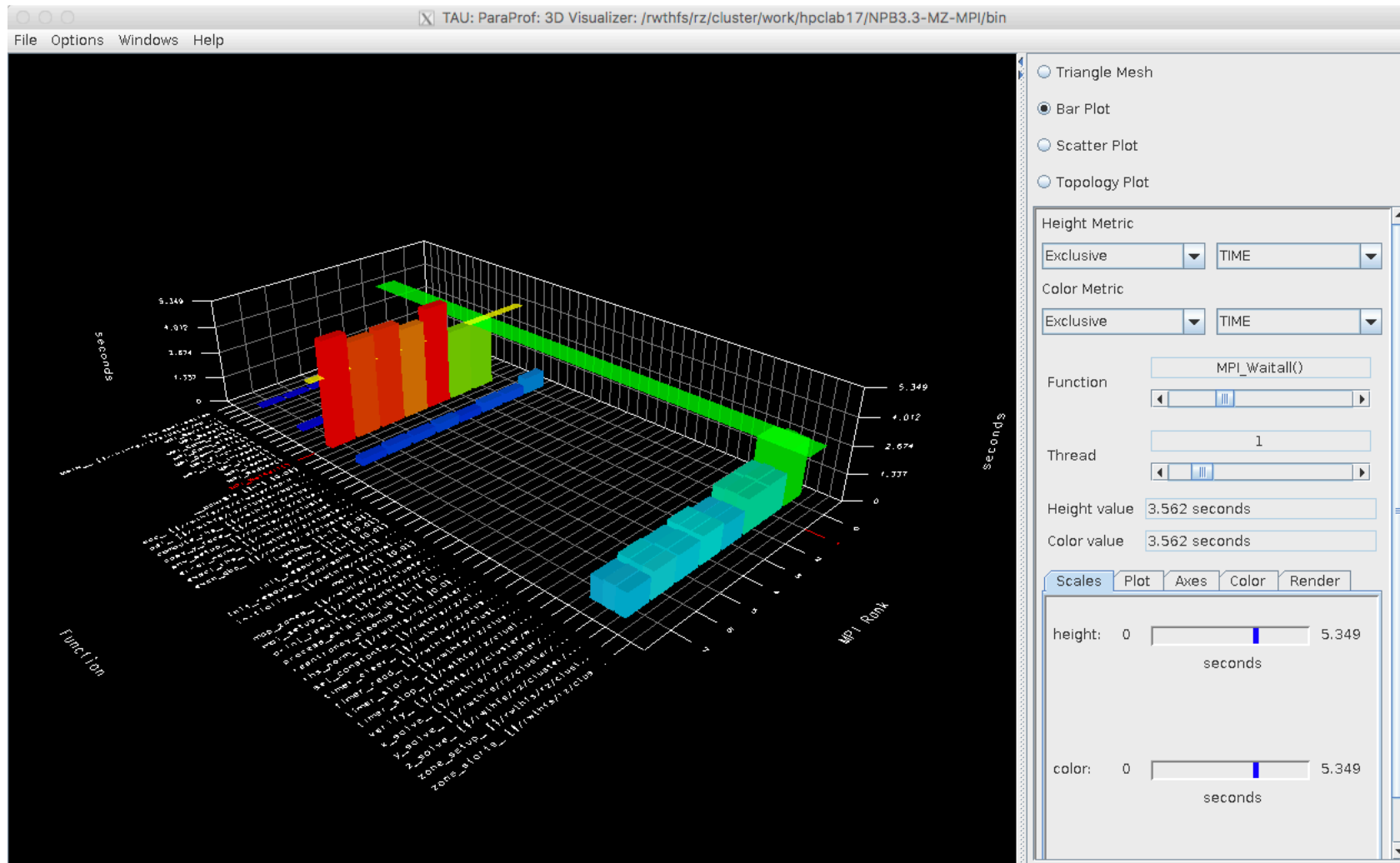
Create a Selective Instrumentation File, Re-instrument, Re-run



ParaProf with Optimized Instrumentation

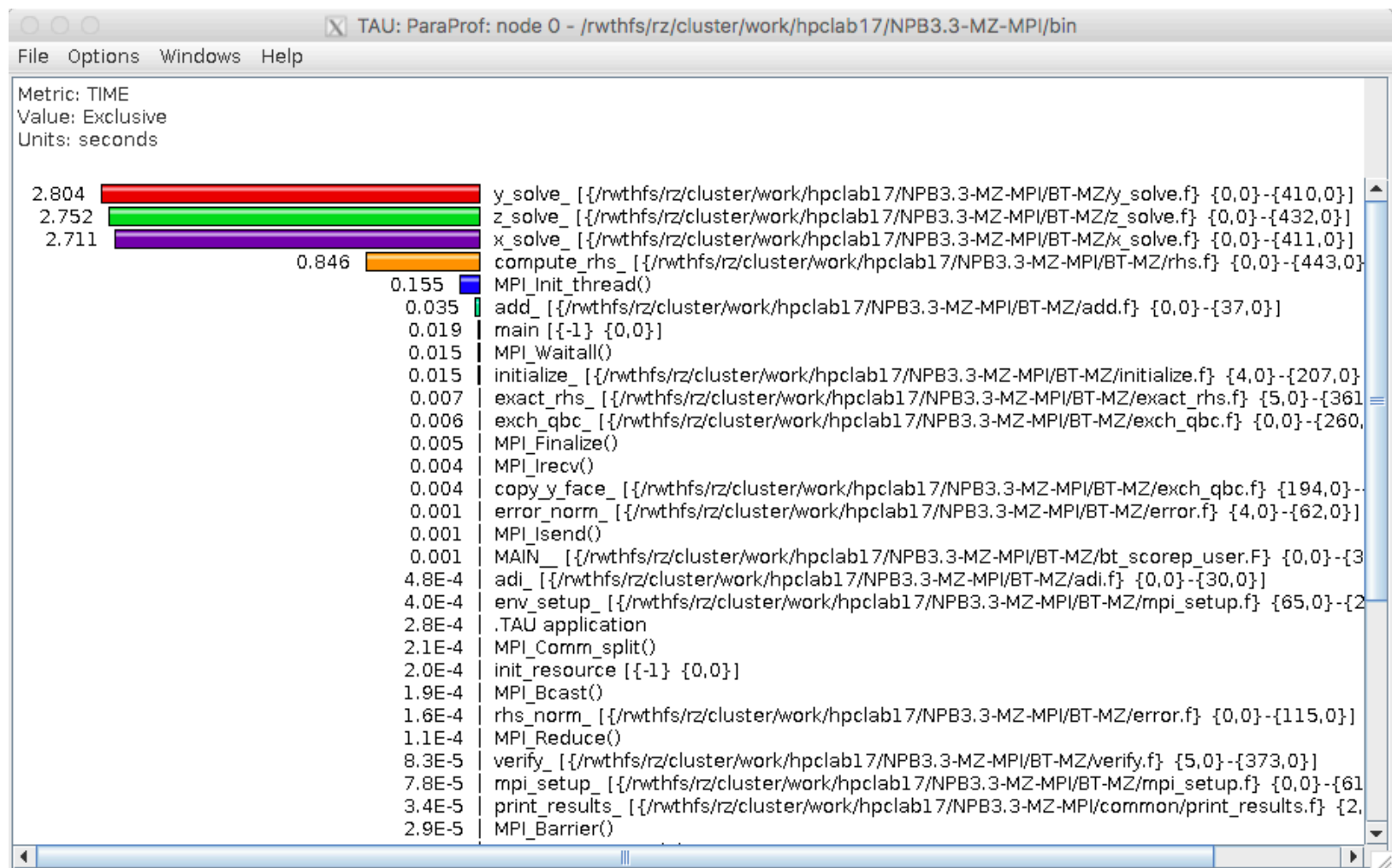


3D Visualization with ParaProf



ParaProf: Node 0

Optimized
instrumentation!



Compile-Time Options

Optional parameters for the TAU_OPTIONS environment variable:

% tau_compiler.sh

- optVerbose Turn on verbose debugging messages
- optCompInst Use compiler based instrumentation
- optNoCompInst Do not revert to compiler instrumentation if source instrumentation fails.
- optTrackIO Wrap POSIX I/O call and calculates vol/bw of I/O operations (Requires TAU to be configured with *-iowrapper*)
- optTrackGOMP Enable tracking GNU OpenMP runtime layer (used without *-opari*)
- optMemDbg Enable runtime bounds checking (see TAU_MEMDBG_* env vars)
- optKeepFiles Does not remove intermediate .pdb and .inst.* files
- optPreProcess Preprocess sources (OpenMP, Fortran) before instrumentation
- optTauSelectFile="*<file>*" Specify selective instrumentation file for *tau_instrumentor*
- optTauWrapFile="*<file>*" Specify path to *link_options.tau* generated by *tau_gen_wrapper*
- optHeaderInst Enable Instrumentation of headers
- optTrackUPCR Track UPC runtime layer routines (used with tau_upc.sh)
- optLinking="" Options passed to the linker. Typically
\$(TAU_MPI_FLIBS) \$(TAU_LIBS) \$(TAU_CXXLIBS)
- optCompile="" Options passed to the compiler. Typically
\$(TAU_MPI_INCLUDE) \$(TAU_INCLUDE) \$(TAU_DEFS)
- optPdtF95Opts="" Add options for Fortran parser in PDT (f95parse/gfparse) ...

Compile-Time Options (contd.)

Optional parameters for the TAU_OPTIONS environment variable:

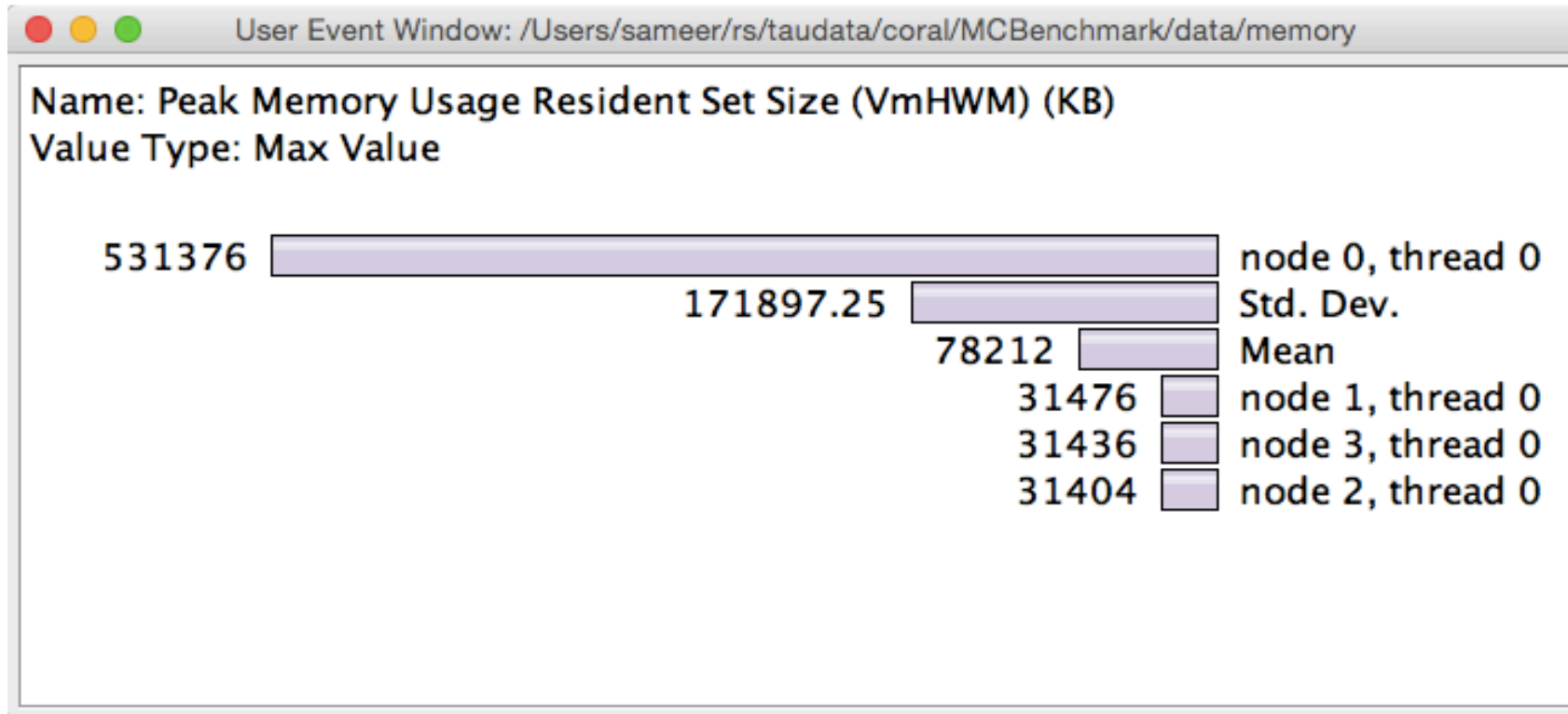
% tau_compiler.sh

| | |
|-------------------------|--|
| -optShared | Use TAU's shared library (libTAU.so) instead of static library (default) |
| -optPdtCxxOpts="" | Options for C++ parser in PDT (cxxparse). |
| -optPdtF90Parser="" | Specify a different Fortran parser |
| -optPdtCleanscapeParser | Specify the Cleanscape Fortran parser instead of GNU gfparser |
| -optTau="" | Specify options to the tau_instrumentor |
| -optTrackDMAPP | Enable instrumentation of low-level DMAPP API calls on Cray |
| -optTrackPthread | Enable instrumentation of pthread calls |

See tau_compiler.sh for a full list of TAU_OPTIONS.

...

Measuring Memory Footprint



```
% export TAU_TRACK_MEMORY_FOOTPRINT=1
```

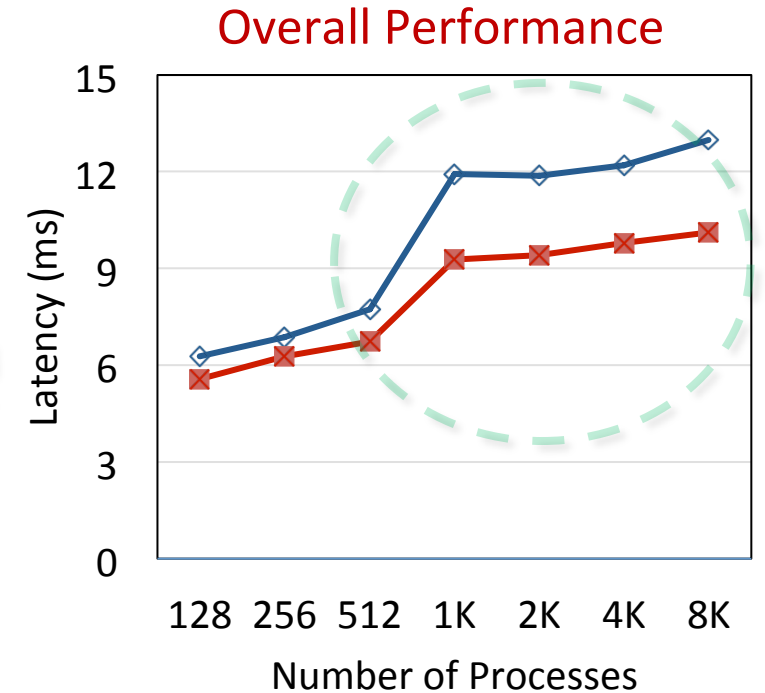
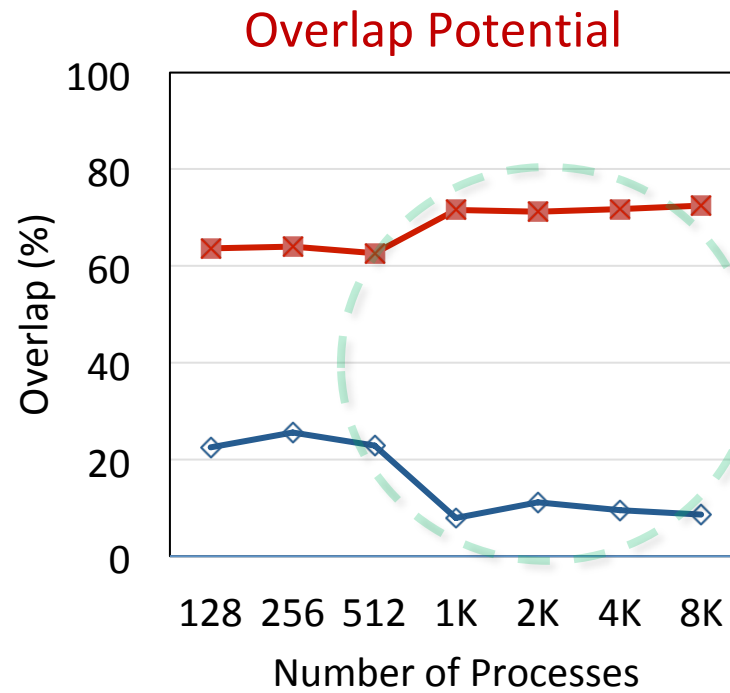
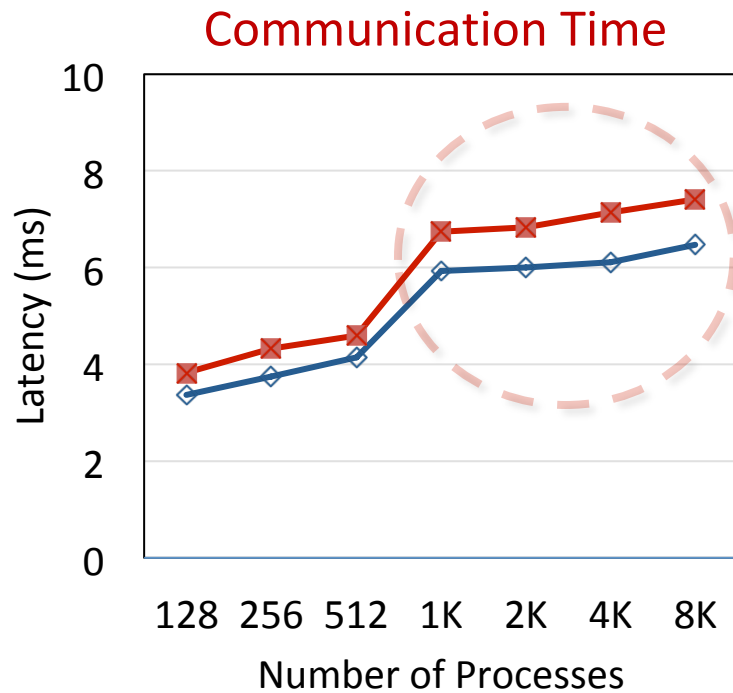
Paraprof:

Right click on a node -> Show Context Event Window -> see memory events

Usage Scenarios with MVAPICH2

- TAU measures the high water mark of total memory usage (TAU_TRACK_MEMORY_FOOTPRINT=1), finds that it is at 98% of available memory, and queries MVAPICH2 to find out how much memory it is using. Based on the number of pools allocated and used, it requests it to reduce the number of VBUF pools and controls the size of these pools using the MPI-T interface. The total memory footprint of the application reduces.
- TAU tracks the message sizes of messages (TAU_COMM_MATRIX=1), detects excessive time spent in MPI_Wait and other synchronization operations. It compares the average message size with the eager threshold and sets the new eager threshold value to match the message size. This could be done offline by re-executing the application with the new CVAR setting for eager threshold or online.
- TAU uses Beacon (backplane for event and control notification) to observe the performance of a running application (for e.g., vbuf pool statistics, high water mark of total and vbuf memory usage, message size statistics).

Performance/Overlap with 128KB Messages at Different Process Counts



◆ Default ■ Dynamic Threshold

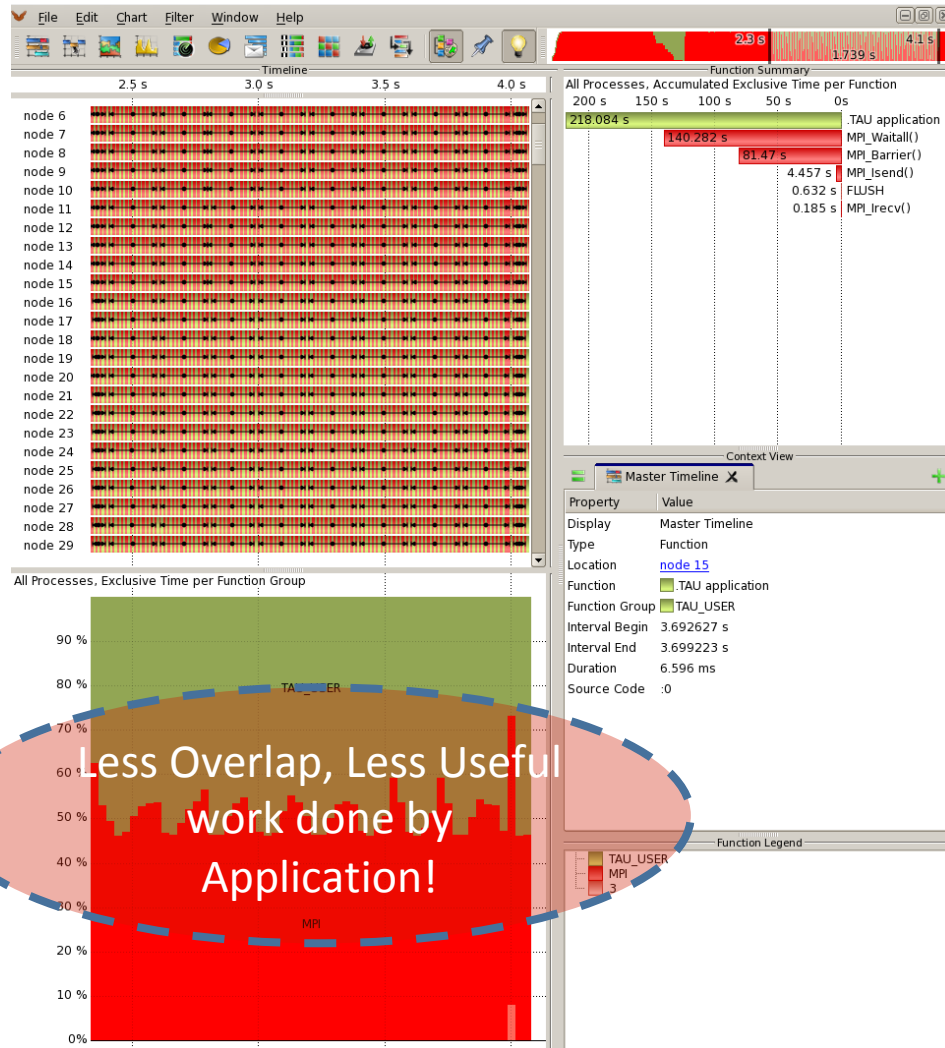
◆ Default ■ Dynamic Threshold

◆ Default ■ Dynamic Threshold

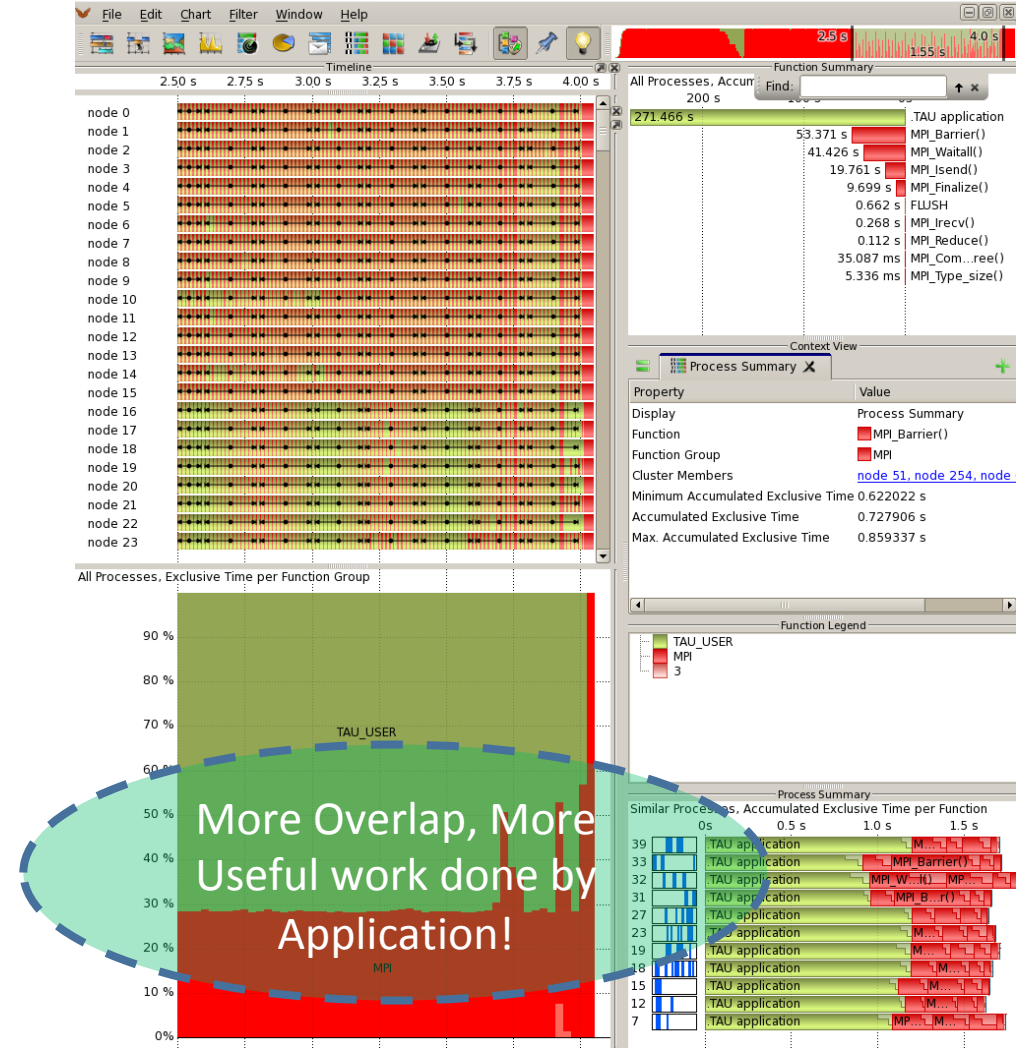
- Dynamic Threshold has degradation in raw communication performance
- Dynamic Threshold has significant benefits for overlap
- Dynamic Threshold better for overall application execution time

Introspecting Impact of Eager Threshold on 3D Stencil Benchmark

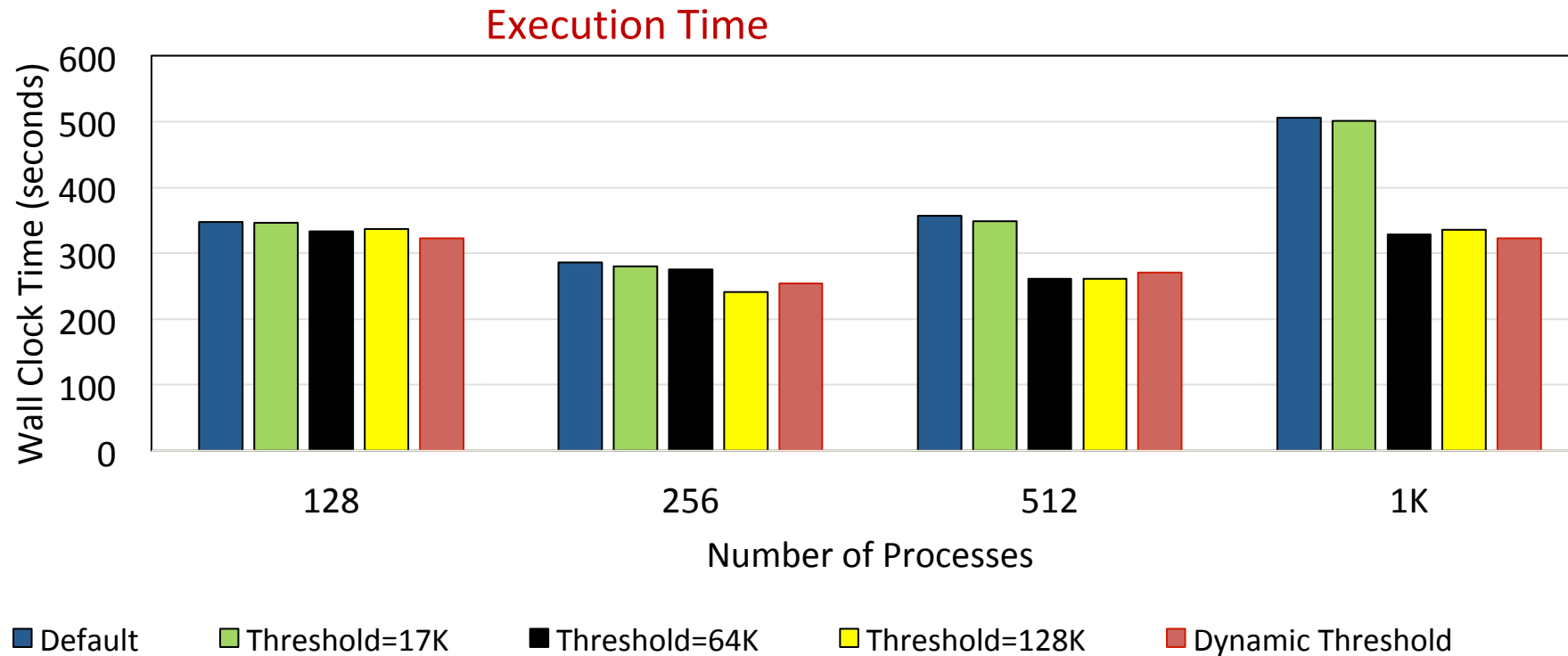
Default



Optimized

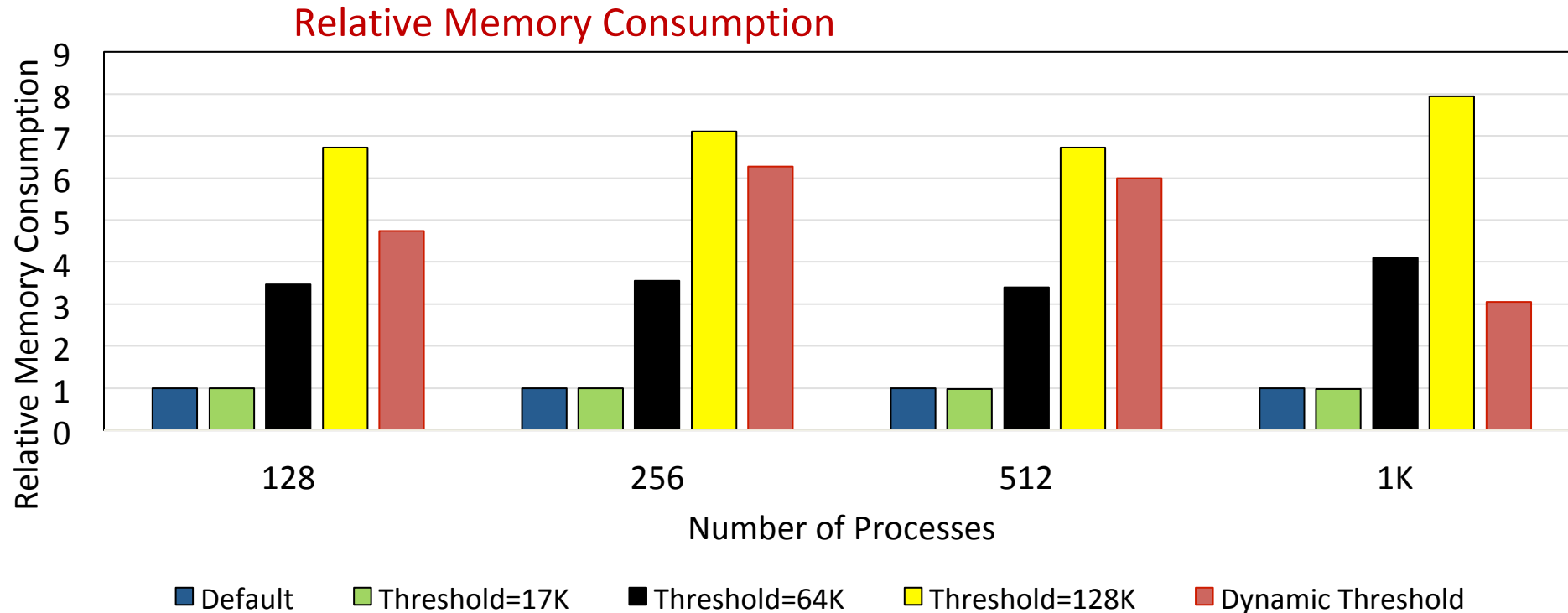


Performance of Amber at Different Process Counts



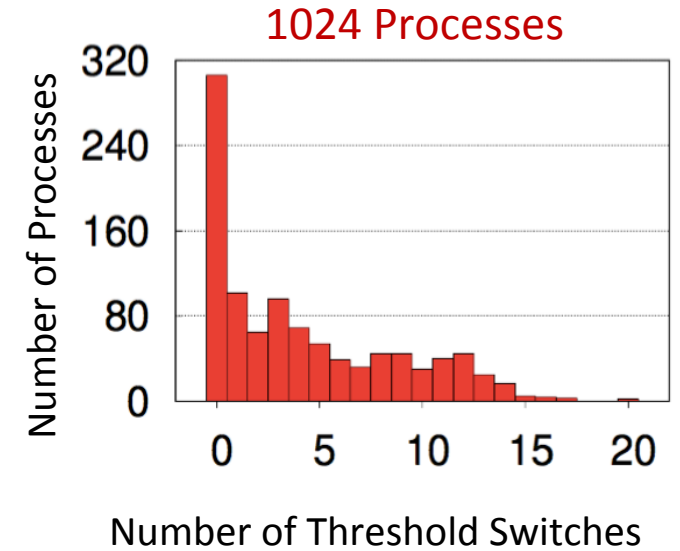
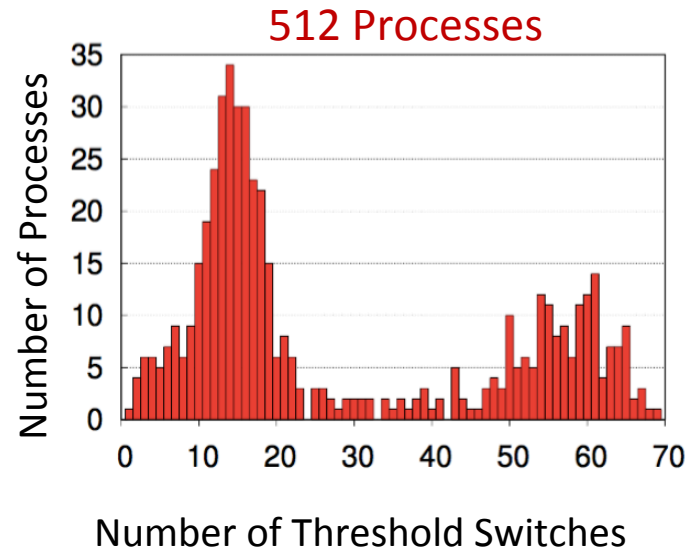
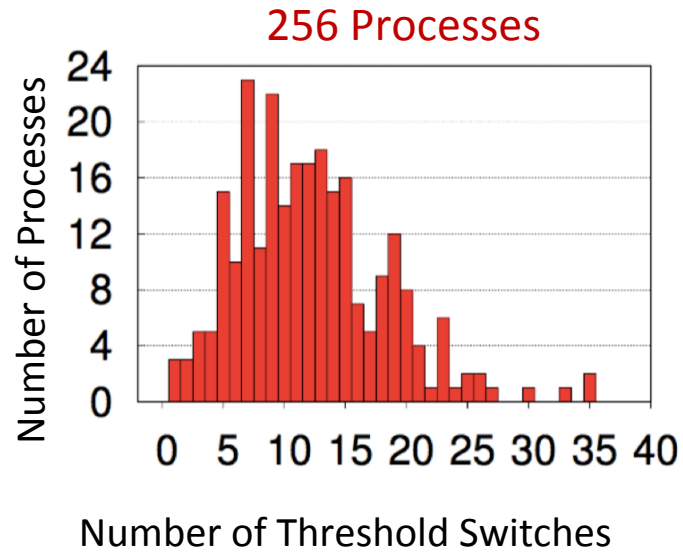
- Optimal values selected manually (Manual Tuning) changes based on job size and problem size
 - Cumbersome, Error prone, and Impractical
- Dynamic Threshold delivers performance on par with best manually tuned version for all job/problem size

Performance of Amber at Different Process Counts (Cont.)

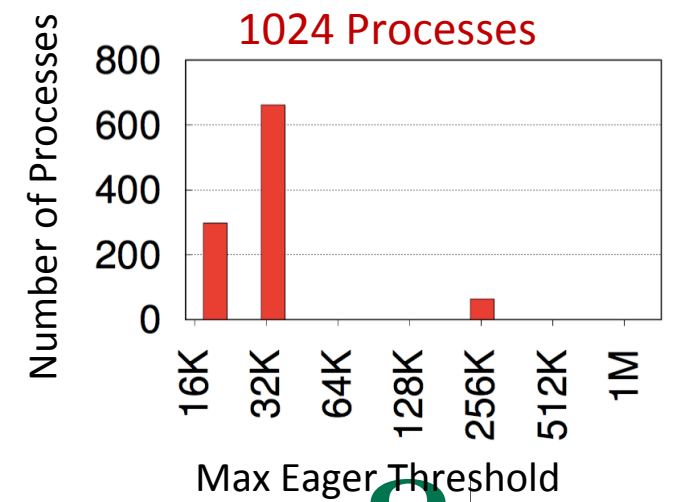
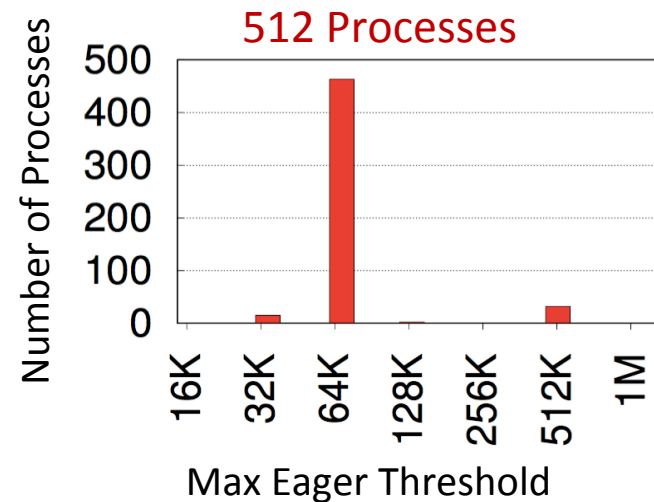
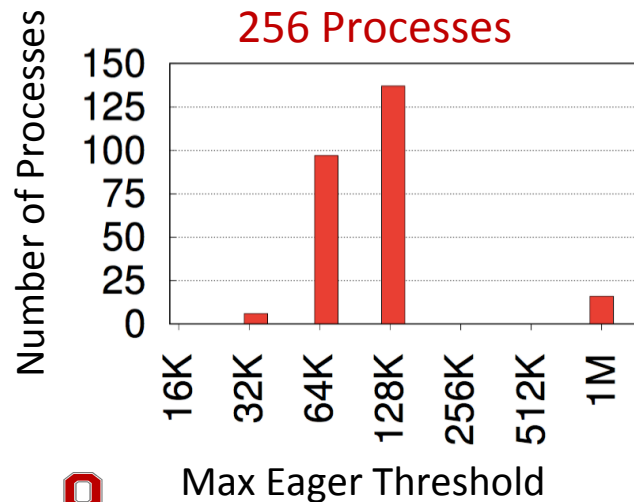


- Default design gives best memory scalability
 - Unable to deliver the best performance
- Dynamic Threshold able to keep memory footprint to what is absolutely needed to obtain performance benefits

Analyzing Dynamic Eager-Threshold Changes

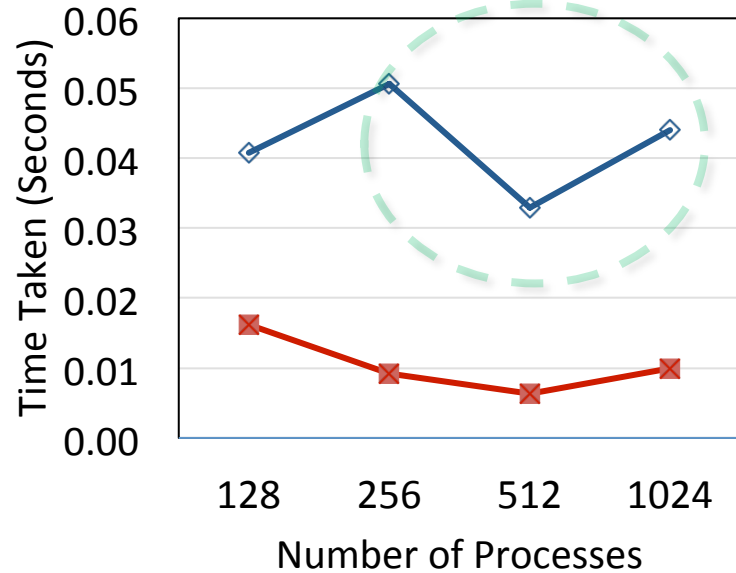


- Number of Eager switches correspond to larger communication requirements at the application level



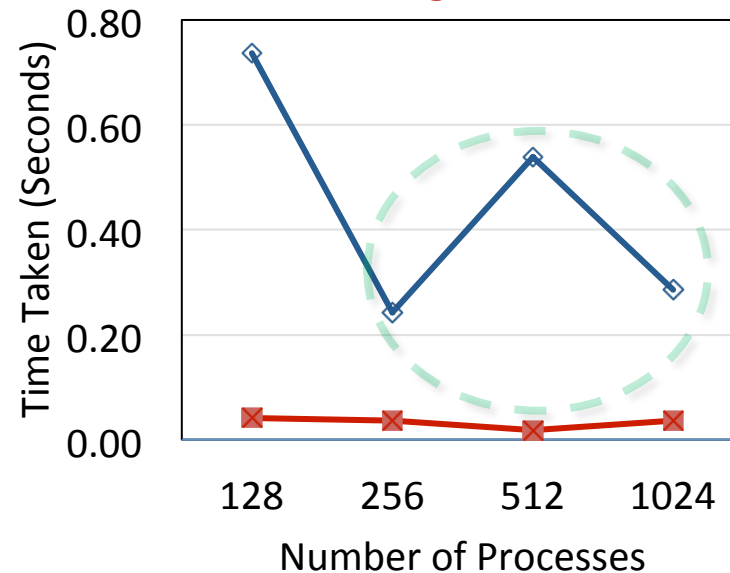
Time taken to Switch Thresholds and Allocate/Free Communication Buffers

Time to Switch Threshold Once



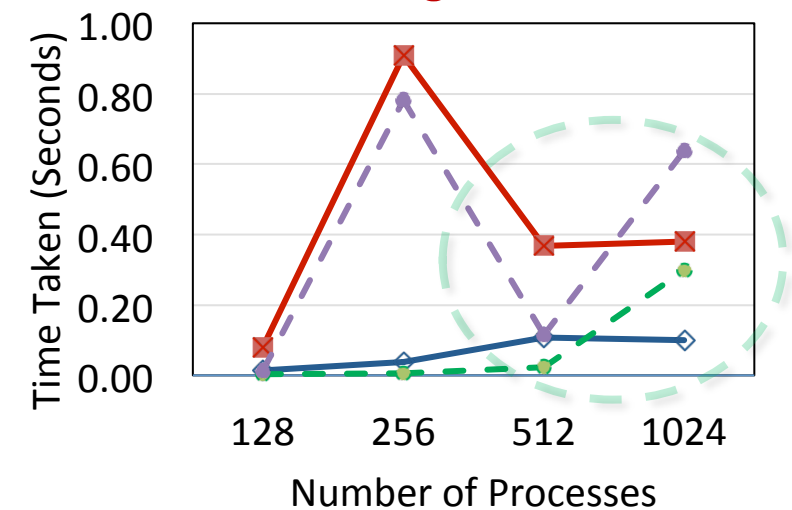
◆ Maximum ■ Average

Cumulative Time Spent in Switching Thresholds



◆ Maximum ■ Average

Time Spent in Allocating / Freeing Buffers



◆ Min-Alloc ■ Max-Alloc
● Min-Free ■ Max-Free

- Maximum overhead of establishing new connection is very low (~40 ms)
- Maximum cumulative time spent by each process for eager-threshold switching very low (< 0.5 s)
 - Less than 0.1% of overall execution time
- Time for dynamically allocating and freeing internal communication buffers also very low
 - Only a negligible percentage of the overall execution time

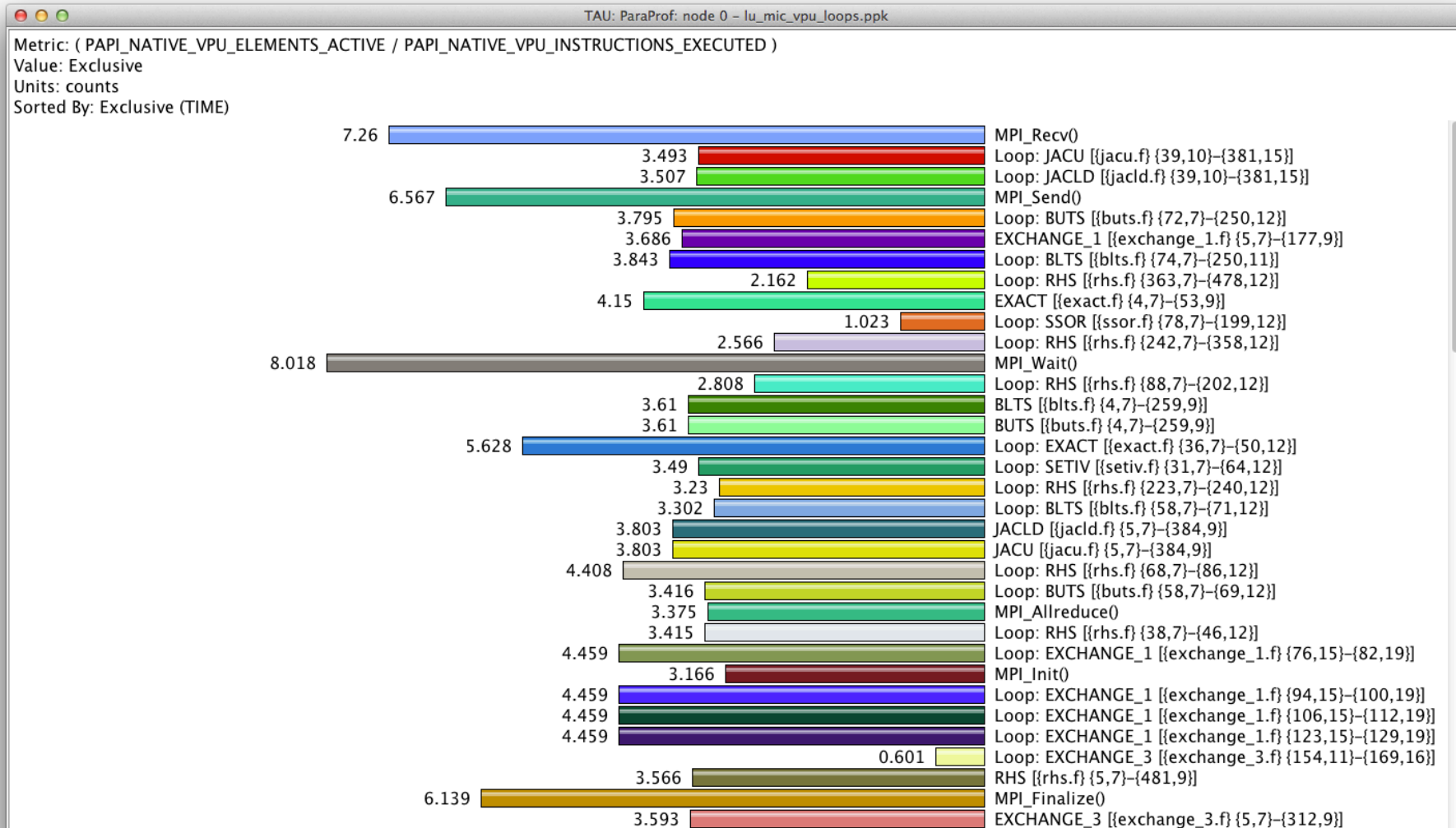
Other Runtime Environment Variables

| Environment Variable | Default | Description |
|----------------------------|---------|---|
| TAU_TRACE | 0 | Setting to 1 turns on tracing |
| TAU_CALLPATH | 0 | Setting to 1 turns on callpath profiling |
| TAU_TRACK_MEMORY_FOOTPRINT | 0 | Setting to 1 turns on tracking memory usage by sampling periodically the resident set size and high water mark of memory usage |
| TAU_SELECT_FILE | | Specify the path to runtime selective instrumentation file for filtering events using exclude and include lists of routines and/or files. |
| TAU_CALLPATH_DEPTH | 2 | Specifies depth of callpath. Setting to 0 generates no callpath or routine information, setting to 1 generates flat profile and context events have just parent information (e.g., Heap Entry: foo) |
| TAU_SAMPLING | 0 | Setting to 1 enables event-based sampling. |
| TAU_TRACK_SIGNALS | 0 | Setting to 1 generate debugging callstack info when a program crashes |
| TAU_COMM_MATRIX | 0 | Setting to 1 generates communication matrix display using context events |
| TAU_THROTTLE | 1 | Setting to 0 turns off throttling. Enabled by default to remove instrumentation in lightweight routines that are called frequently |
| TAU_THROTTLE_NUMCALLS | 100000 | Specifies the number of calls before testing for throttling |
| TAU_THROTTLE_PERCALL | 10 | Specifies value in microseconds. Throttle a routine if it is called over 100000 times and takes less than 10 usec of inclusive time per call |
| TAU_COMPENSATE | 0 | Setting to 1 enables runtime compensation of instrumentation overhead |
| TAU_PROFILE_FORMAT | Profile | Setting to "merged" generates a single file. "snapshot" generates xml format |
| TAU_METRICS | TIME | Setting to a comma separated list generates other metrics. (e.g., TIME,ENERGY,PAPI_FP_INS,PAPI_NATIVE_<event>:<subevent>) |

Runtime Environment Variables (contd.)

| Environment Variable | Default | Description |
|------------------------------------|--------------------------|--|
| TAU_TRACK_MEMORY_LEAKS | 0 | Tracks allocates that were not de-allocated (needs <code>-optMemDbg</code> or <code>tau_exec -memory</code>) |
| TAU_EBS_SOURCE | TIME | Allows using PAPI hardware counters for periodic interrupts for EBS (e.g., <code>TAU_EBS_SOURCE=PAPI_TOT_INS</code> when <code>TAU_SAMPLING=1</code>) |
| TAU_EBS_PERIOD | 100000 | Specifies the overflow count for interrupts |
| TAU_MEMDBG_ALLOC_MIN/MAX | 0 | Byte size minimum and maximum subject to bounds checking (used with <code>TAU_MEMDBG_PROTECT_*</code>) |
| TAU_MEMDBG_OVERHEAD | 0 | Specifies the number of bytes for TAU's memory overhead for memory debugging. |
| TAU_MEMDBG_PROTECT_BELOW/ ABOVE | 0 | Setting to 1 enables tracking runtime bounds checking below or above the array bounds (requires <code>-optMemDbg</code> while building or <code>tau_exec -memory</code>) |
| TAU_MEMDBG_ZERO_MALLOC | 0 | Setting to 1 enables tracking zero byte allocations as invalid memory allocations. |
| TAU_MEMDBG_PROTECT_FREE | 0 | Setting to 1 detects invalid accesses to deallocated memory that should not be referenced until it is reallocated (requires <code>-optMemDbg</code> or <code>tau_exec -memory</code>) |
| TAU_MEMDBG_ATTEMPT_CONTINUE | 0 | Setting to 1 allows TAU to record and continue execution when a memory error occurs at runtime. |
| TAU_MEMDBG_FILL_GAP | Undefined | Initial value for gap bytes |
| TAU_MEMDBG_ALINGMENT | <code>sizeof(int)</code> | Byte alignment for memory allocations |
| TAU_EVENT_THRESHOLD | 0.5 | Define a threshold value (e.g., .25 is 25%) to trigger marker events for min/max |

Evaluating Extent of Vectorization on MIC

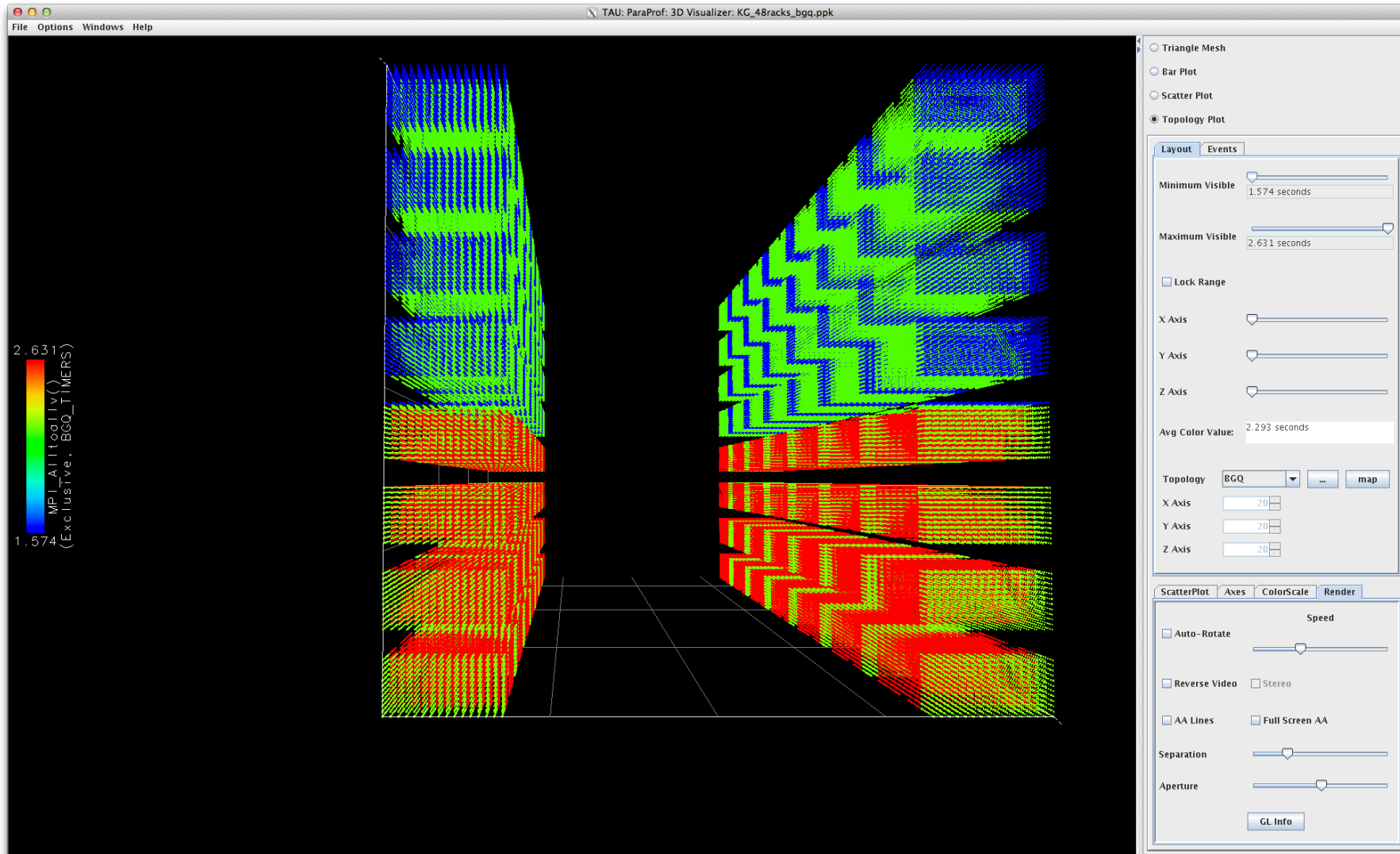


% export TAU_MAKEFILE=\$TAUROOT/mic_linux/lib/Makefile.tau-papi-mpi-pdt

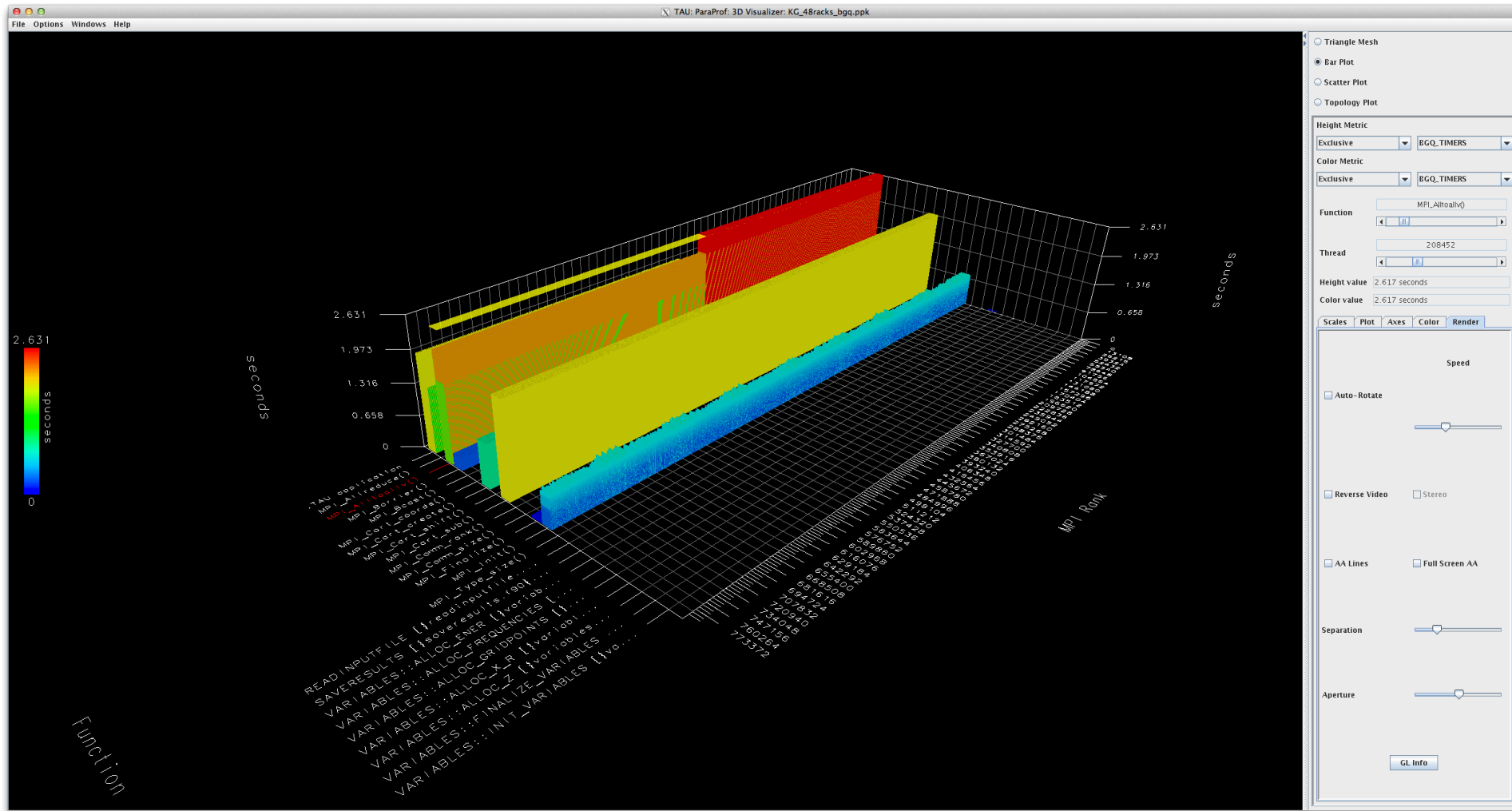
% export TAU_METRICS=TIME,

PAPI_NATIVE_VPU_ELEMENTS_ACTIVE,PAPI_NATIVE_VPU_INSTRUCTIONS_EXECUTED

ParaProf's Topology Display Window (BGQ)

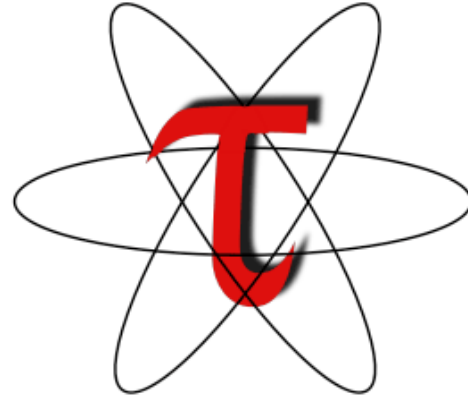


ParaProf's Scalable 3D Visualization (BGQ)



786,432 ranks

Download TAU from U. Oregon



<http://www.hpclinux.com> [OVA file]

<http://tau.uoregon.edu/tau.tgz>

for more information

Free download, open source, BSD license

PRL, University of Oregon, Eugene



www.uoregon.edu

Support Acknowledgments

US Department of Energy (DOE)

- ANL
- Office of Science contracts, ECP
- SciDAC, LBL contracts
- LLNL-LANL-SNL ASC/NNSA contract
- Battelle, PNNL and ORNL contract

CEA, France

Department of Defense (DoD)

- PETTT, HPCMP

National Science Foundation (NSF)

- SI2-SSI, Glassbox

Intel Corporation

NASA

Partners:

- University of Oregon
- The Ohio State University
- ParaTools, Inc.
- University of Tennessee, Knoxville
- T.U. Dresden, GWT
- Jülich Supercomputing Center



UNIVERSITY OF OREGON



THE OHIO STATE UNIVERSITY

THE UNIVERSITY OF TENNESSEE UT



ParaTools

