Analyzing the Performance of IWAVE on a Cluster using HPCToolkit

John Mellor-Crummey and **Laksono Adhianto**
Department of Computer Science
Rice University
{johnmc,laksono}@rice.edu

SciDAC Scientific Discovery through Advanced Computing

RICE

Challenges for HPC Practitioners

- Execution environments and applications are rapidly evolving
  - architecture
    - rapidly changing multicore microprocessor designs
    - increasing scale of parallel systems
    - growing use of accelerators, e.g. GPGPU
  - applications
    - MPI everywhere to threaded implementations
    - adding additional scientific capabilities to existing applications
    - maintaining multiple variants or configurations for particular problems
- Steep increase in application development effort to attain performance, evolvability, and portability
- Application developers need to
  - assess weaknesses in algorithms and their implementations
  - improve scalability and performance within and across nodes
  - adapt to changes in emerging architectures
  - overhaul algorithms & data structures as needed

Challenges for HPC Practitioners

- **Execution environments and applications are rapidly evolving**
  - architecture
    - rapidly changing multicore microprocessor designs
    - increasing scale of parallel systems
    - growing use of accelerators, e.g. GPGPU
  - applications
    - MPI everywhere to threaded implementations
    - adding additional scientific capabilities to existing applications
    - maintaining multiple variants or configurations for particular problems
- **Steep increase in application development effort to attain performance, evolvability, and portability**
- **Application developers need to**
  - assess weaknesses in algorithms and their implementations
  - improve scalability and performance within and across nodes
  - adapt to changes in emerging architectures
  - overhaul algorithms & data structures as needed

Performance tools can play an important role as a guide

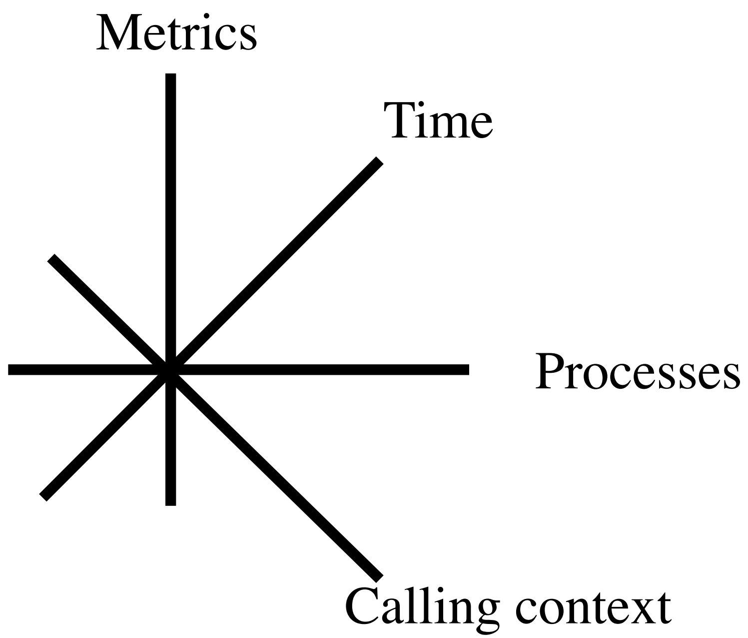
Motivation

- In December 2011, a member of CRAY Chapel team was able to achieve about 20x speedup
  - — multi-threaded program compiled for a single locale
  - — less than a day's work
- In January 2011, Rice Coarray Fortran team detected performance bottleneck in their HPCC FFT benchmark
  - — majority of the time was spent in executing communication to perform a bit-reversal permutation
  - — changing the algorithm and using coarse-grain all-to-all communication reduced the cost to only about 6%
- In December 2011, HPCToolkit team identified several performance bottlenecks in a DOD procurement benchmark
  - — inefficient use of Posix I/O
  - — load imbalance when not power of 2 processors
- And so on ...

HPCToolkit: Why it's Cool

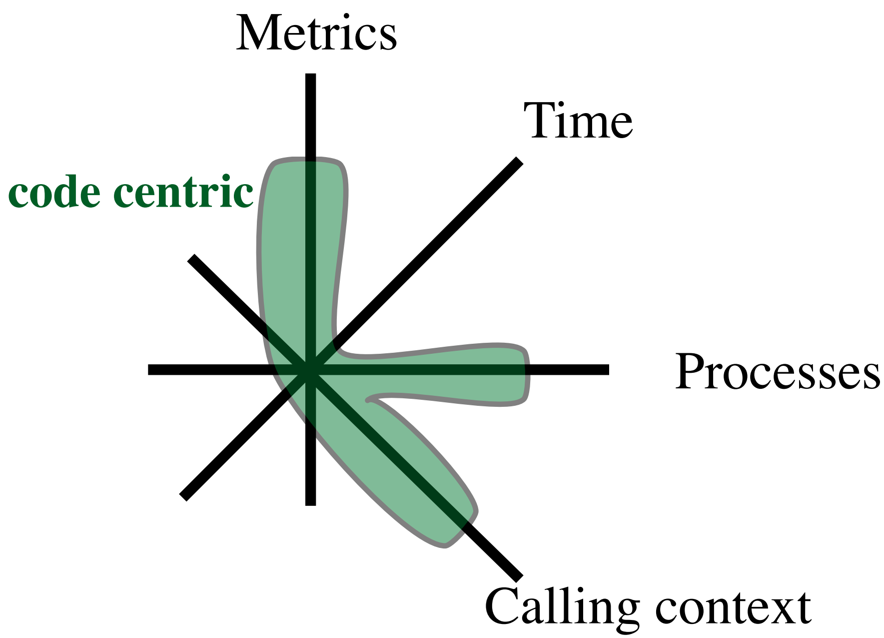
- **It runs (almost) anywhere, anytime by anyone**
  - — language independent (C, C++, Fortran, ...)
  - — programming model independent (MPI, OpenMP, UPC, ...)
  - — operating systems independent (any Linux flavor)
  - — architecture independent (x86_64, ppc64, MIPS)
  - — compiler independent (Intel, PGI, GNU, Pathscale, ...)
  - — runtime independent (MPICH, OpenMPI, GASNet, ...)
- **Usable on production executions**
  - — low overhead: sampling rather than instrumentation
  - — large number of processors
- **It's easy to use**
  - — no need to rebuild code
  - — work for fully optimized code
- **Effective performance analysis**
  - — fine-grain attribution (lines, loops, procedures, call chains, ...)
  - — correlate measurements with code for actionable results

Understanding Performance Measurements



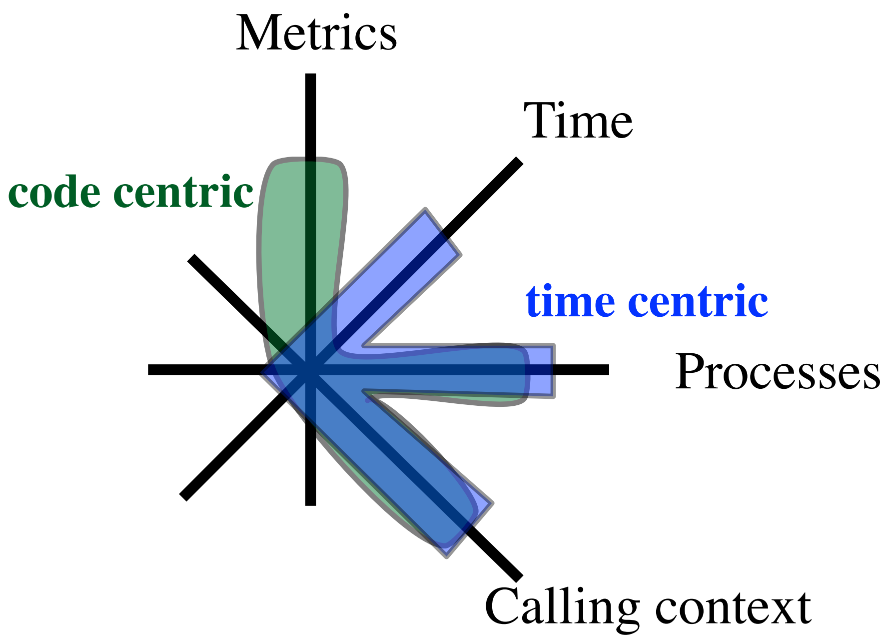
- **Four dimensions of performance data in HPCToolkit**
  - **metrics:** **wallclock, L2 cache miss, cycles, flops, ...**
  - **calling context:** **main -> a -> b -> ...**
  - **processes or ranks:** **0, 1, ..., P**
  - **time:** **from the beginning of measurement to the end**
- **Warning: finding performance bottlenecks can be challenging**

Understanding Performance Measurements



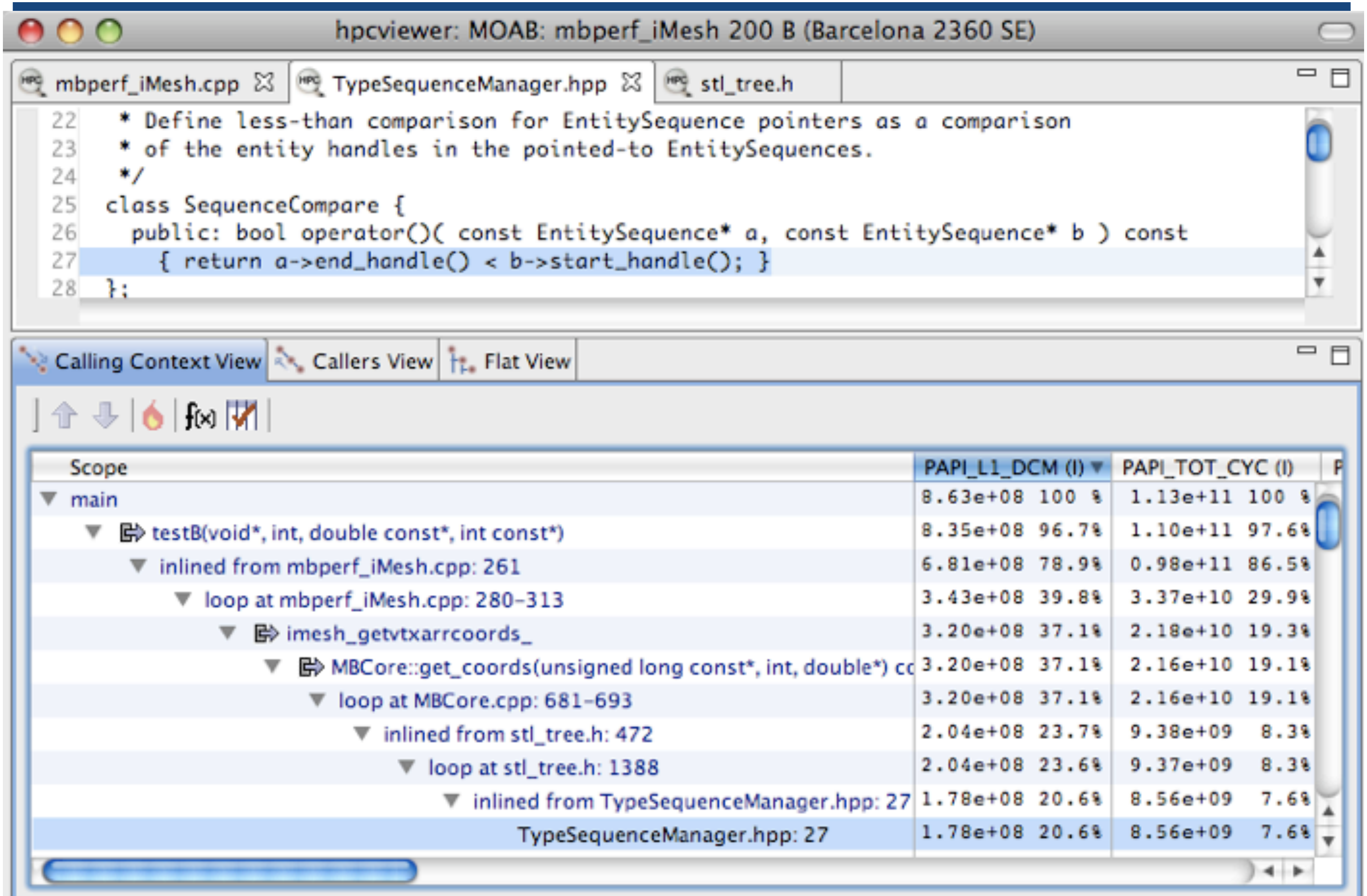
- **Four dimensions of performance data in HPCToolkit**
  - **metrics:** wallclock, L2 cache miss, cycles, flops, ...
  - **calling context:** main -> a -> b -> ...
  - **processes or ranks:** 0, 1, ..., P
  - **time:** from the beginning of measurement to the end
- **Warning: finding performance bottlenecks can be challenging**

Understanding Performance Measurements



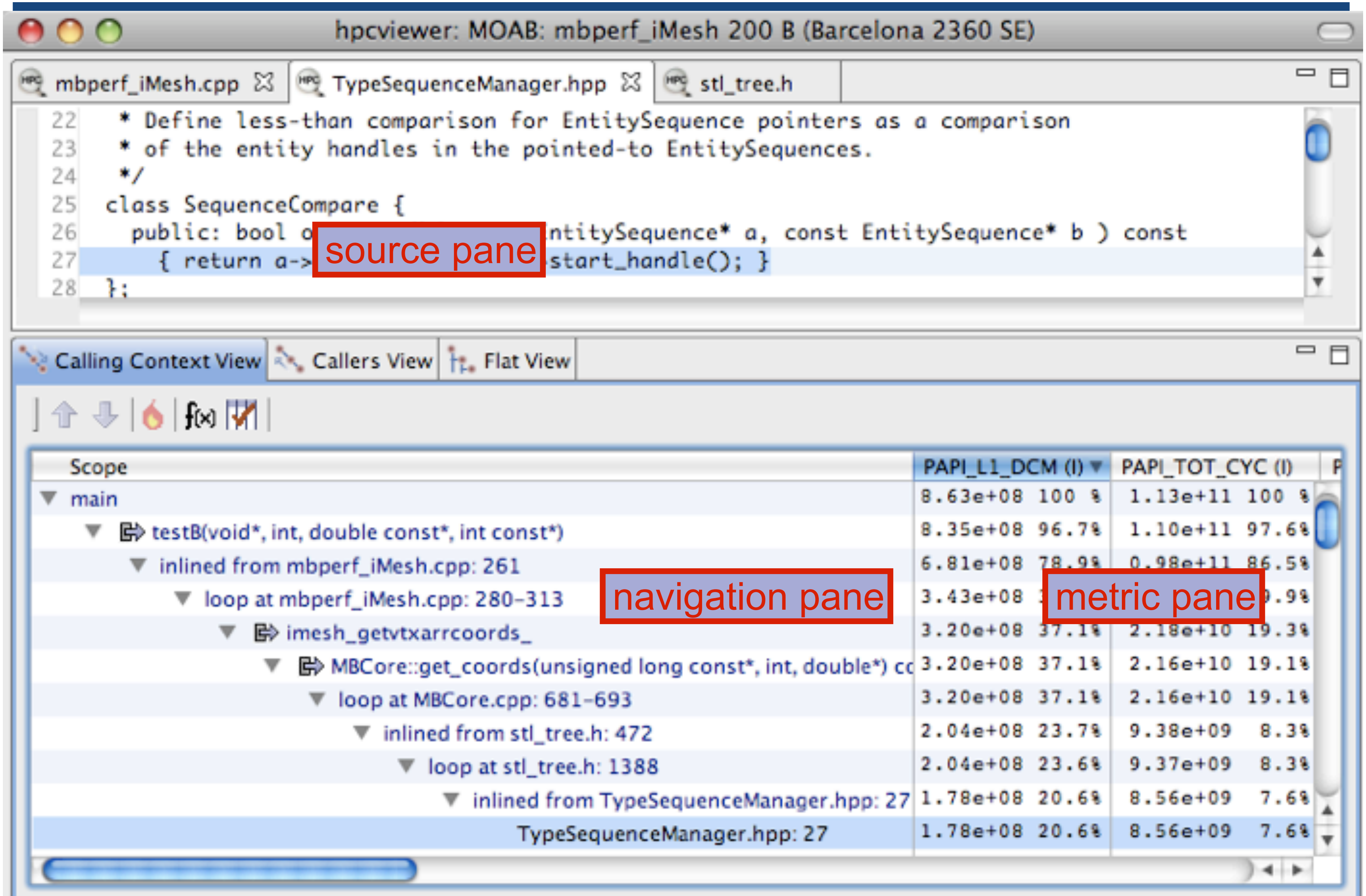
- **Four dimensions of performance data in HPCToolkit**
  - **metrics:** **wallclock, L2 cache miss, cycles, flops, ...**
  - **calling context:** **main -> a -> b -> ...**
  - **processes or ranks:** **0, 1, ..., P**
  - **time:** **from the beginning of measurement to the end**
- **Warning: finding performance bottlenecks can be challenging**

Code-centric Analysis with hpcviewer

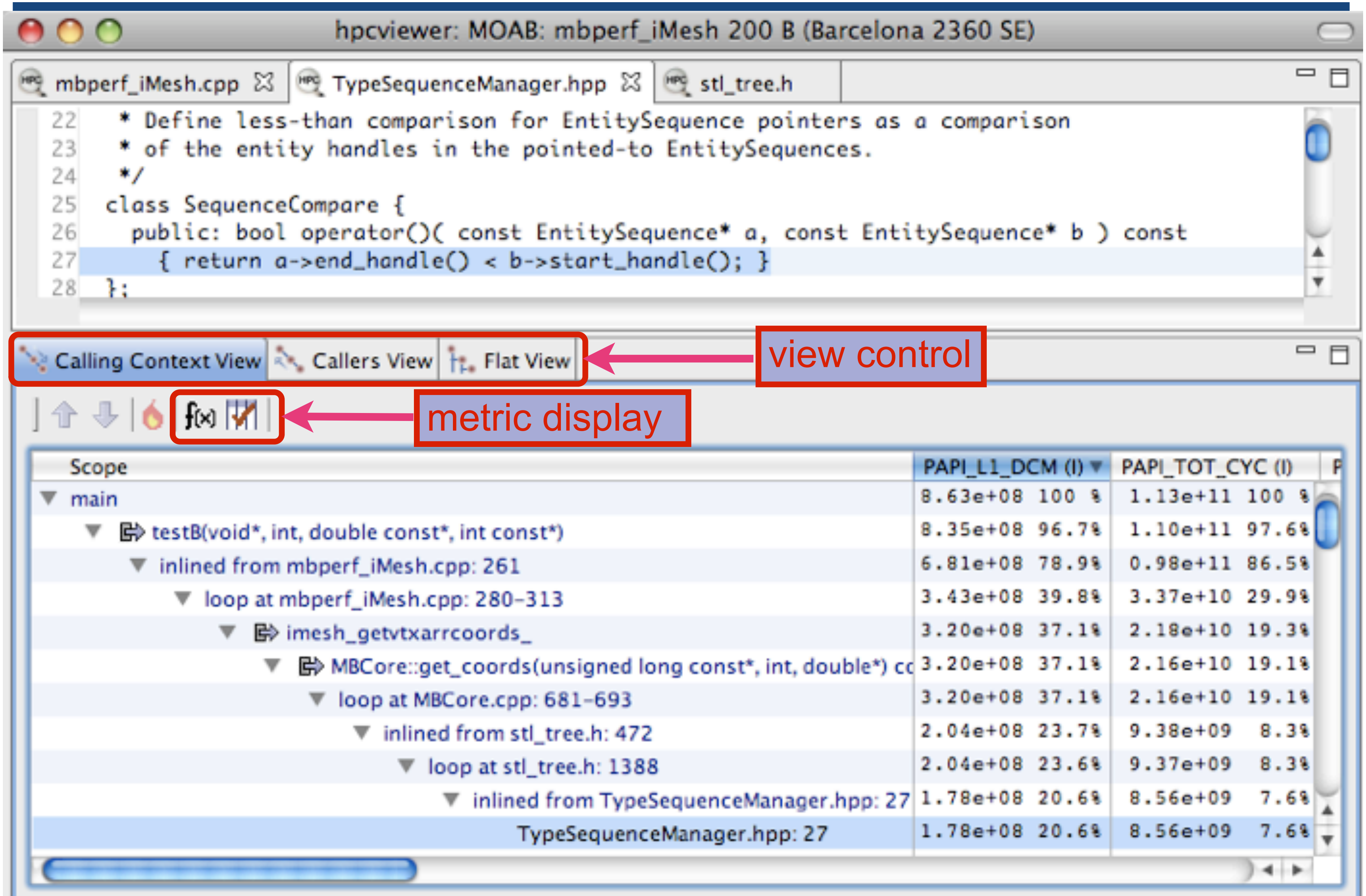


Thursday, March 29, 2012

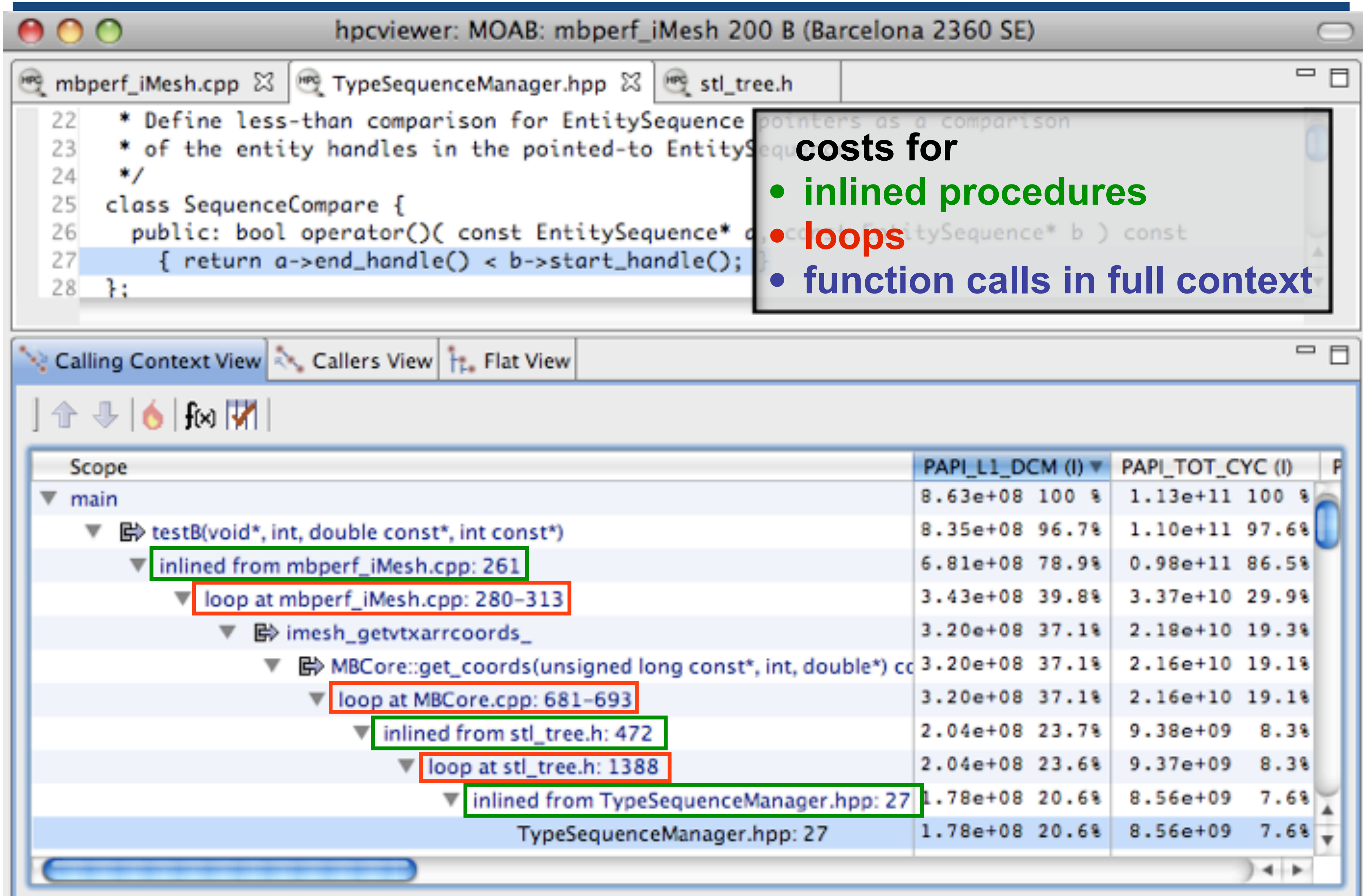
Code-centric Analysis with `hpcviewer`



Code-centric Analysis with `hpcviewer`



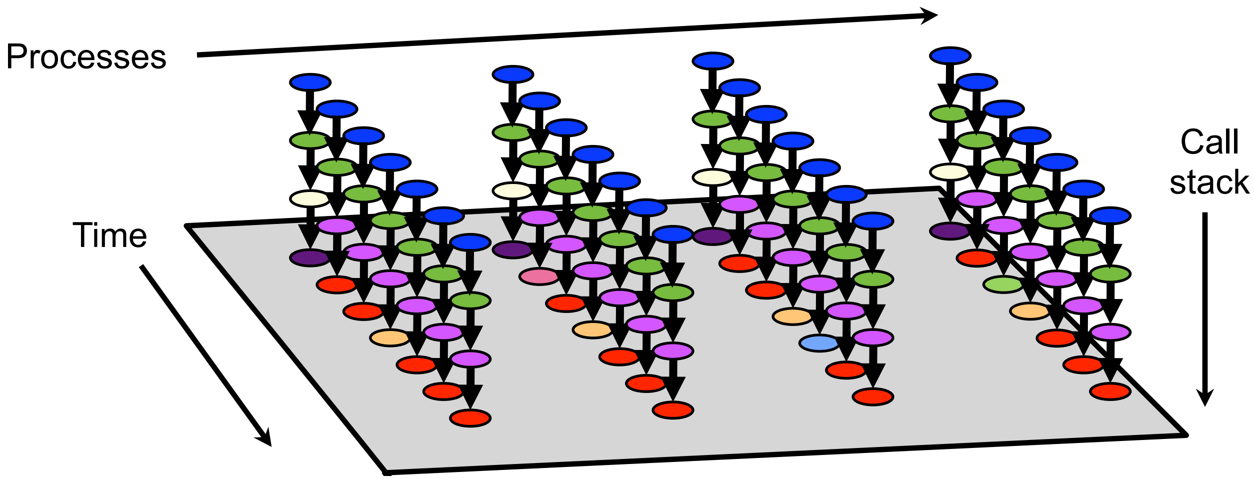
Code-centric Analysis with `hpcviewer`



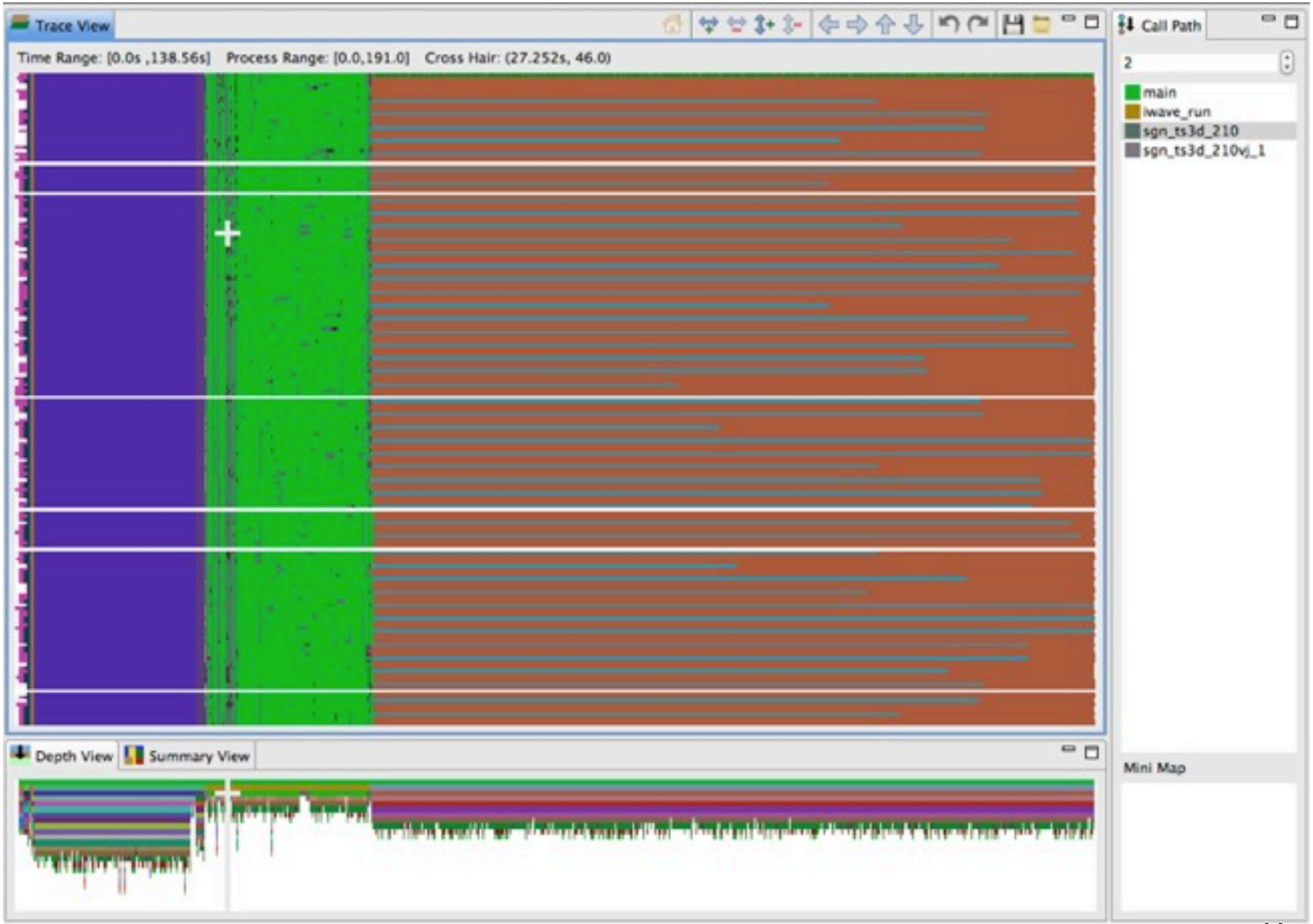
Understanding Executions over Time

- **Profiling compresses out the temporal dimension**
  - —**temporal patterns, e.g. serialization, are invisible in profiles**
- **What can we do? Trace call path samples**
  - —**sketch:**
    - **N times per second, take a call path sample of each thread**
    - **organize the samples for each thread along a time line**
    - **view how the execution evolves left to right**
    - **what do we view?**

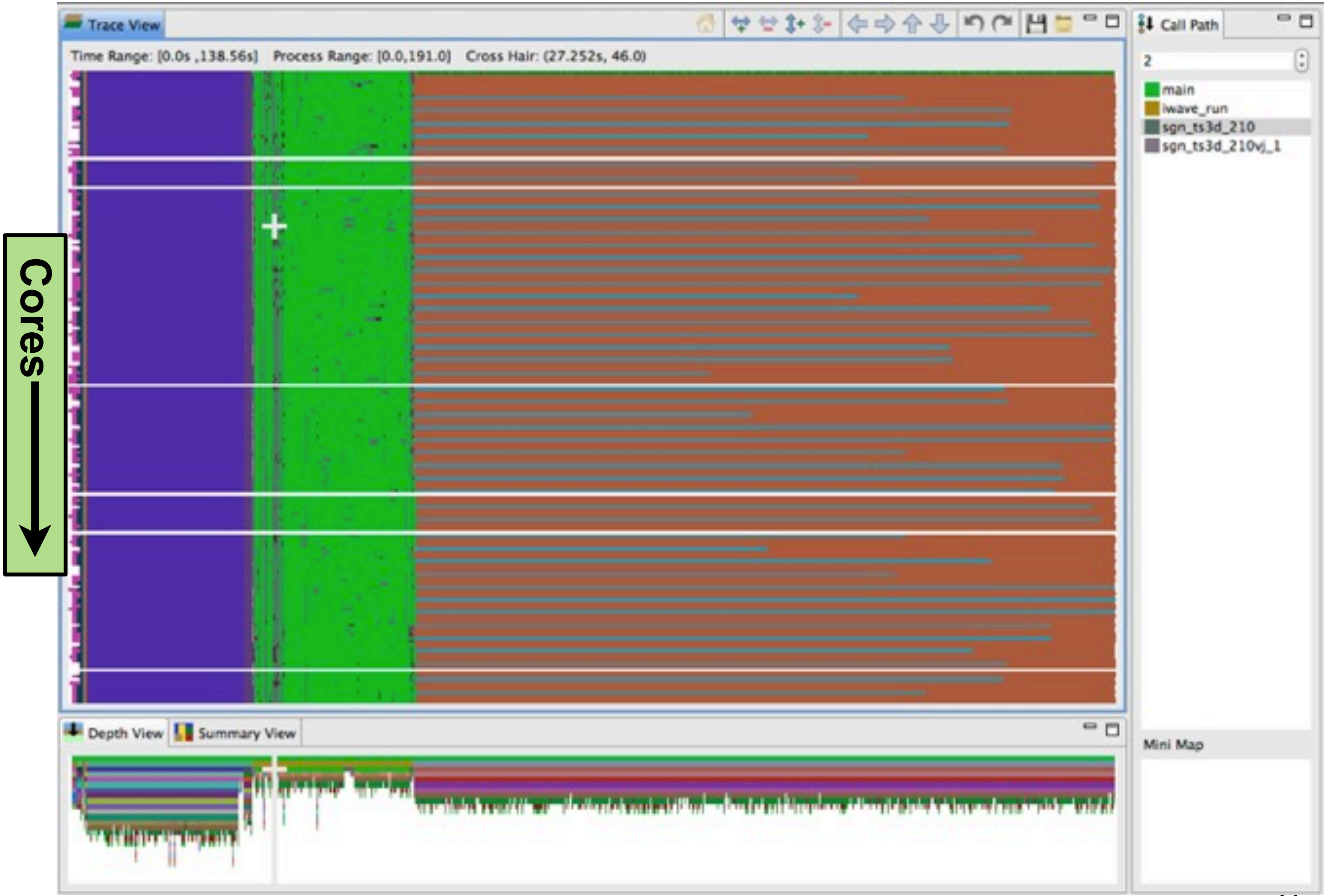
      assign each procedure a color; view a depth slice of an execution



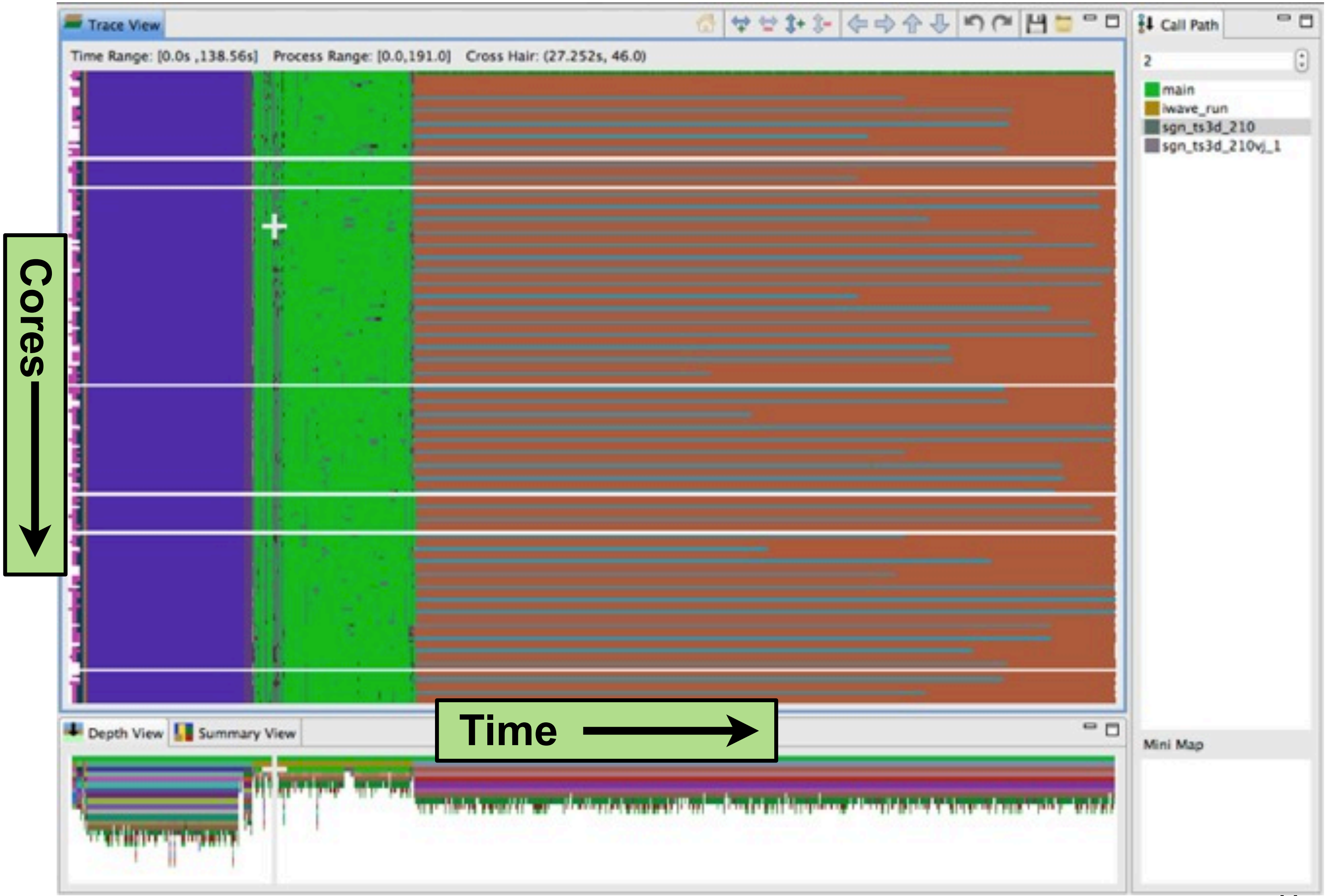
HPCToolkit's Time-centric View



HPCToolkit’s Time-centric View

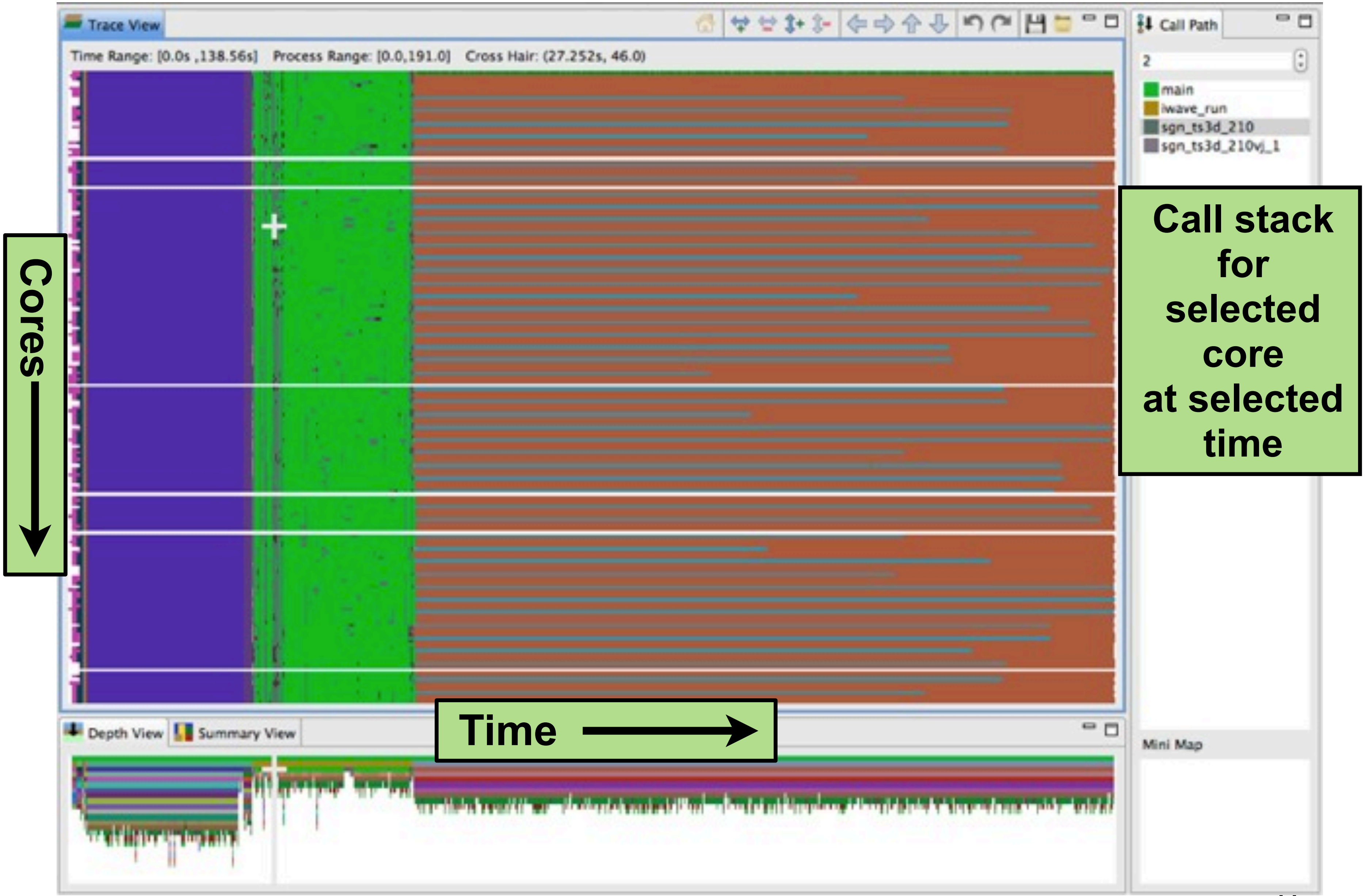


HPCToolkit's Time-centric View

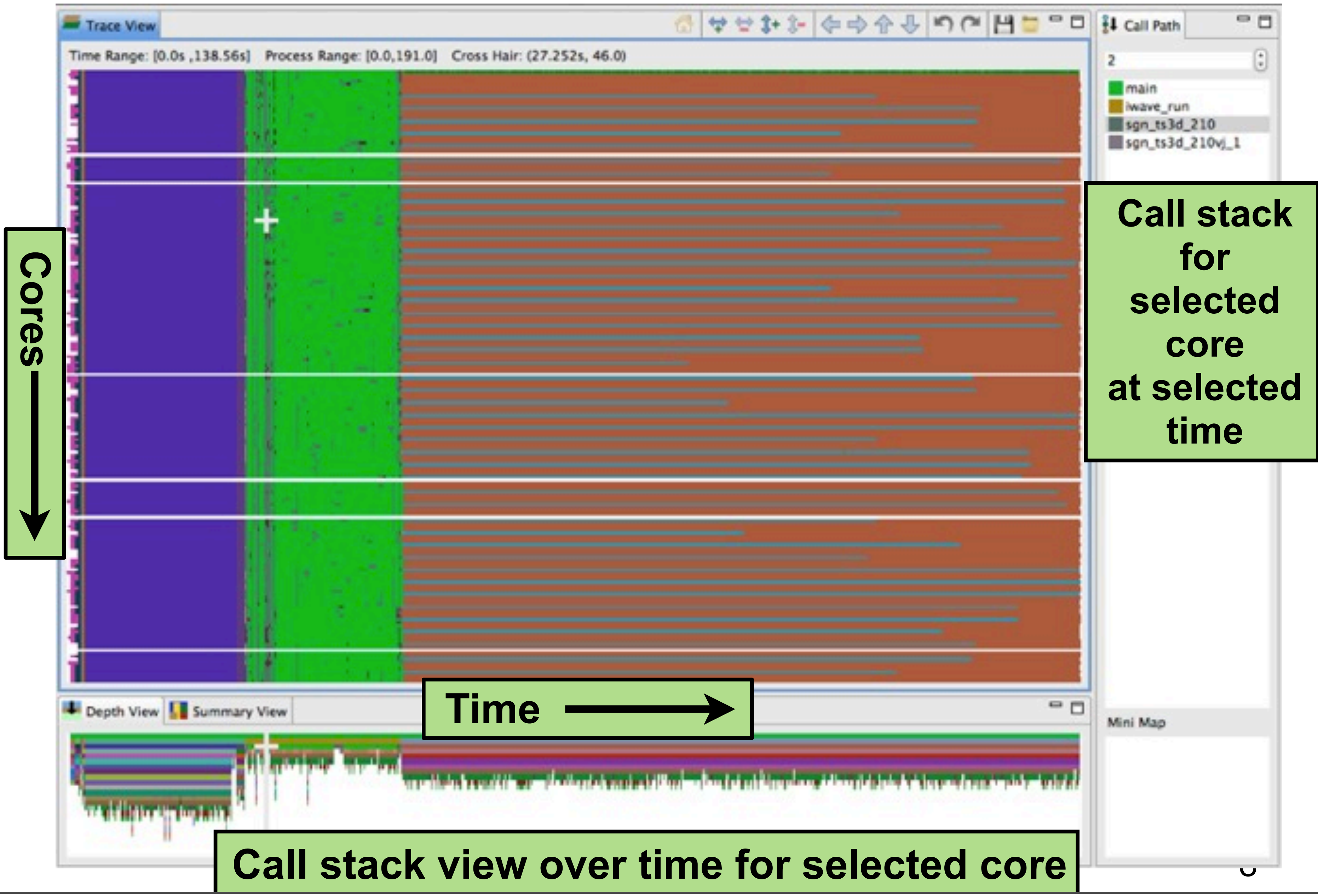


Thursday, March 29, 2012

HPCToolkit's Time-centric View



HPCToolkit's Time-centric View



IWAVE - Rice Inversion Project (Symes, PI)

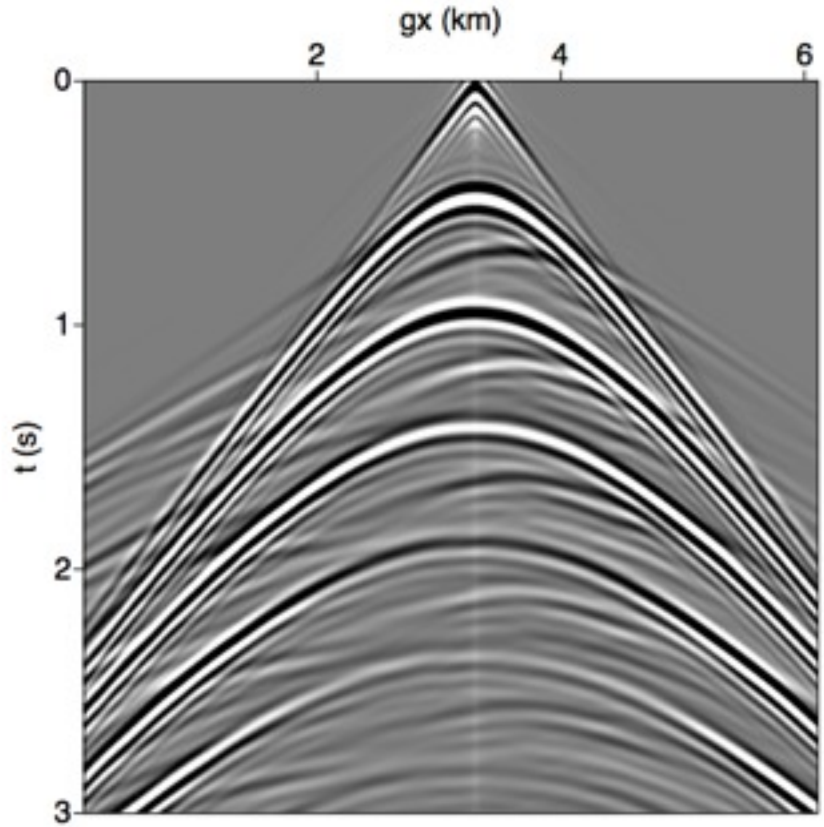
- **Framework for finite difference simulation**
  - common services - memory, communication, I/O, job control
  - prescribed interfaces - problem description, numerical schemes
- **Applications written to the framework**
  - staggered grid acoustics with PML
  - staggered grid isotropic elasticity with PML
- **Portable - ISO C99, MPI, OpenMP**
- **Modeling engine for migration and inversion**

IWAVE on a Cluster

- Experimental Platforms
  - DaVinci
    - node: two 2.83 GHz Intel Westmere (6-core) processors, 48GB RAM
    - interconnect: 4x QDR Inifinband (40Gb/s)
    - GPFS fast scratch
  - STIC
    - node: two 2.4 GHz Intel Nehalem (4-core), 12GB RAM
    - interconnect: 2 4x DDR Infiniband links per node (20Gb/s each)
    - 11TB Panassas scratch
- IWAVE configuration studied
  - asg package
    - staggered grid finite difference (pressure, velocity) acoustic modeling
  - 3D finite difference configuration
  - compiled with icc, version 12.0.0
    - -O3 -std=c99 -g
  - SEAM 20M GRID, FOR SHOT S020433

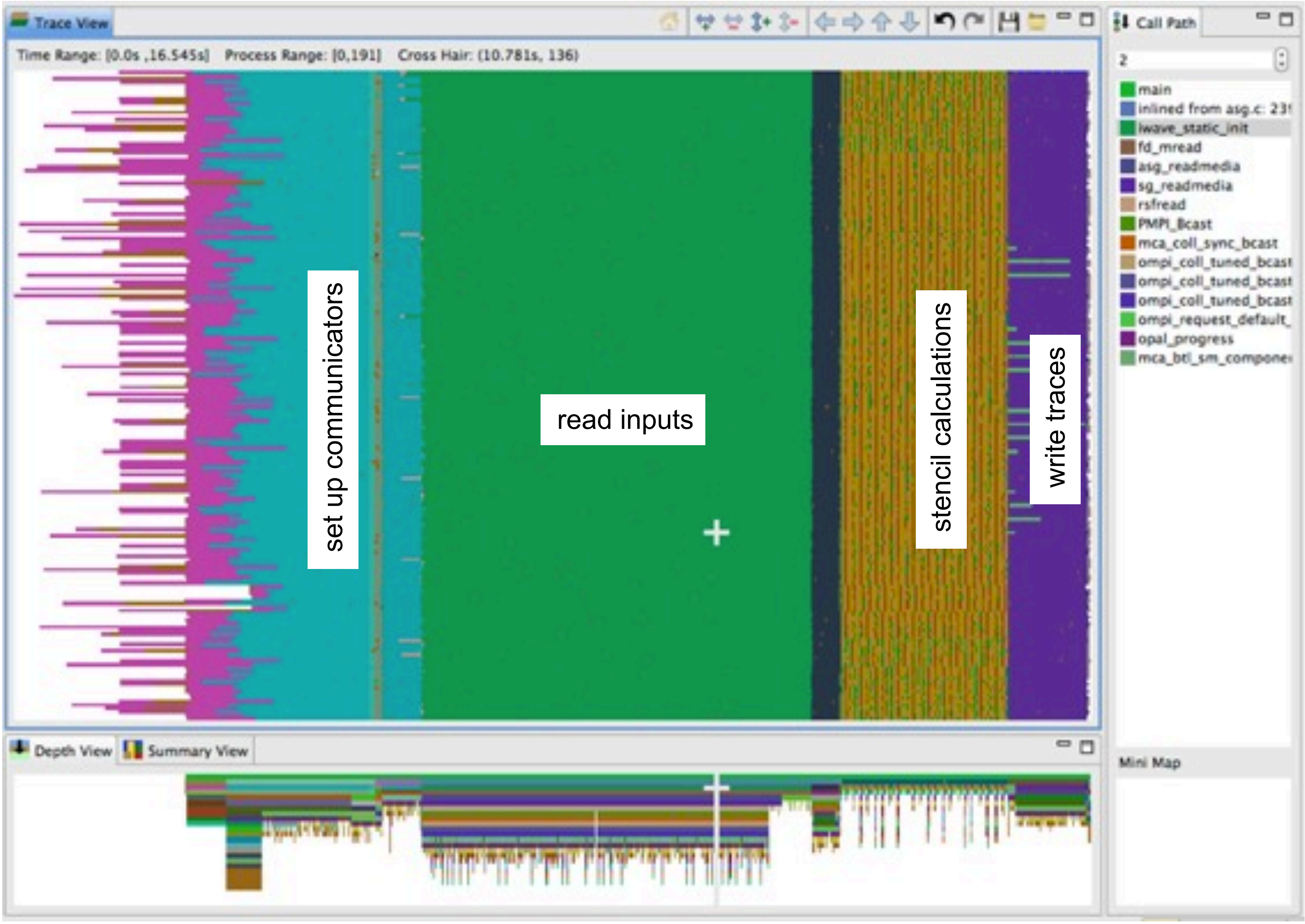
IWAVE Execution

- 8 x 6 x 4 MPI decomposition
- Model info
  - SEAMHALF20M
    - density info 808MB
    - velocity info 808MB
- IWAVE run
  - read velocity model
    - broadcast to all processors
  - read density model
    - broadcast to all processors
  - perform stencil calculations to compute pressures and velocities
  - write traces to disk

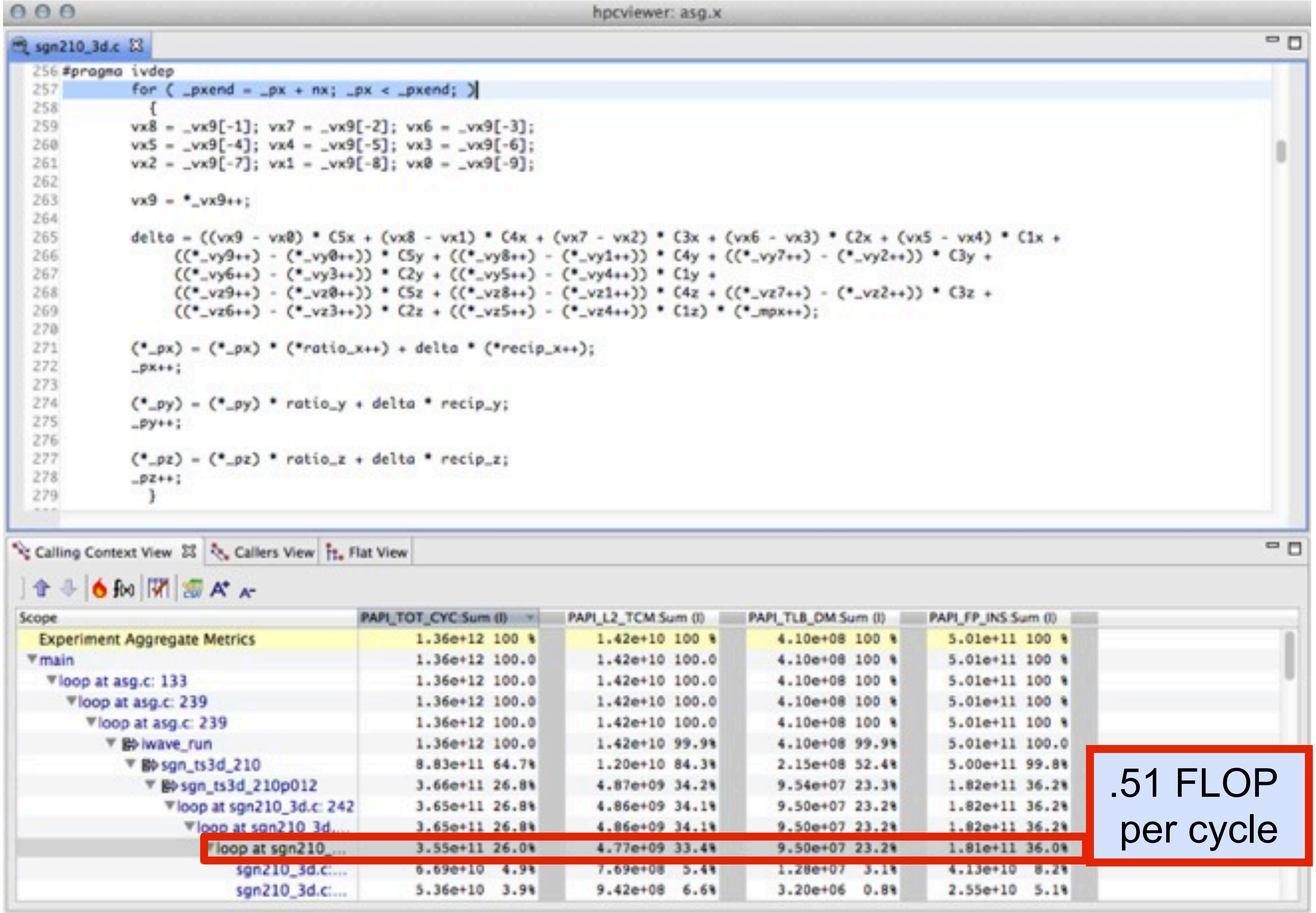


Time-centric view of IWAVE

- **MPI decomposition 8 x 6 x 4**
- **32 nodes, 6 cores each (192 processor cores), OpenMPI**

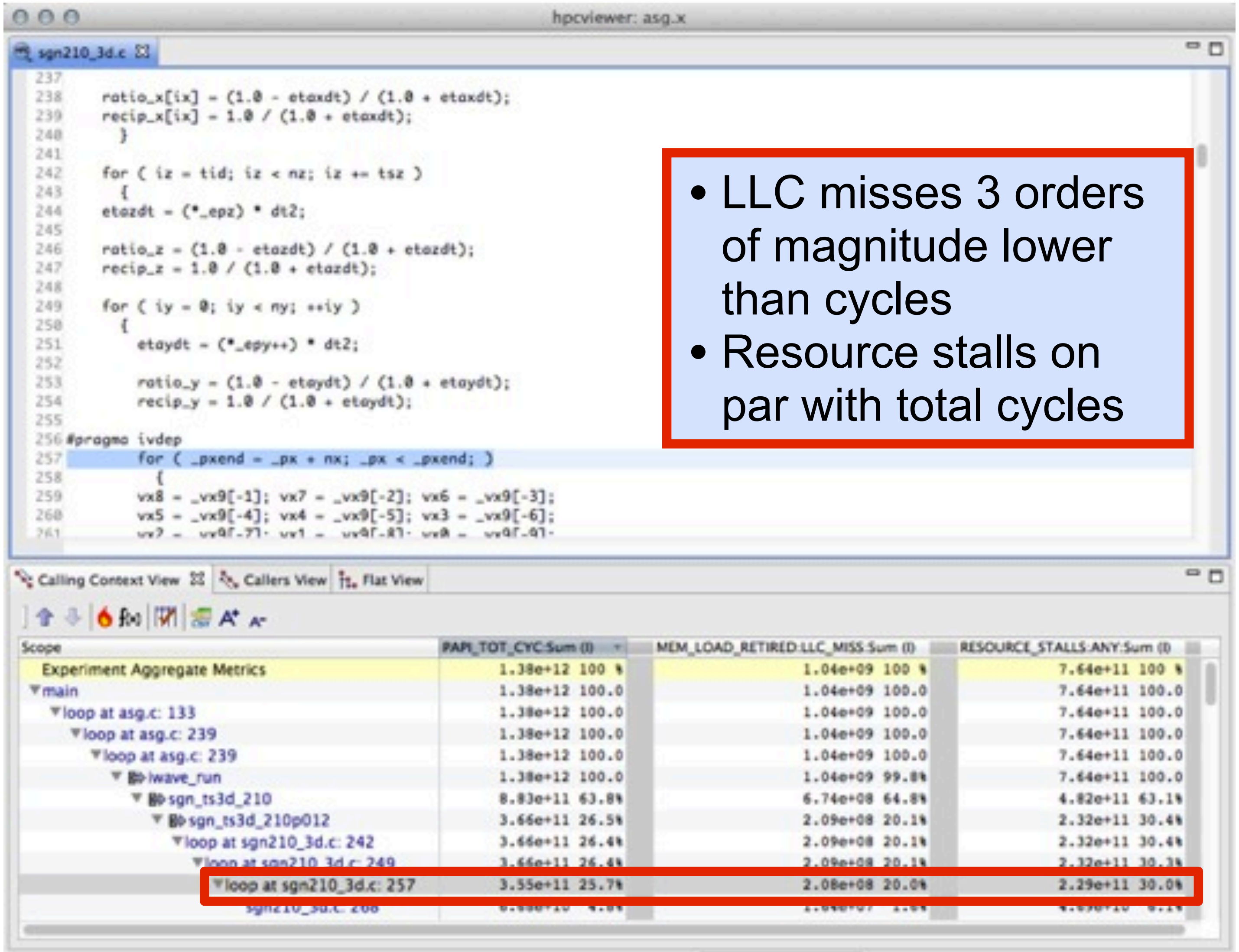


IWAVE Stencil - Overall Performance



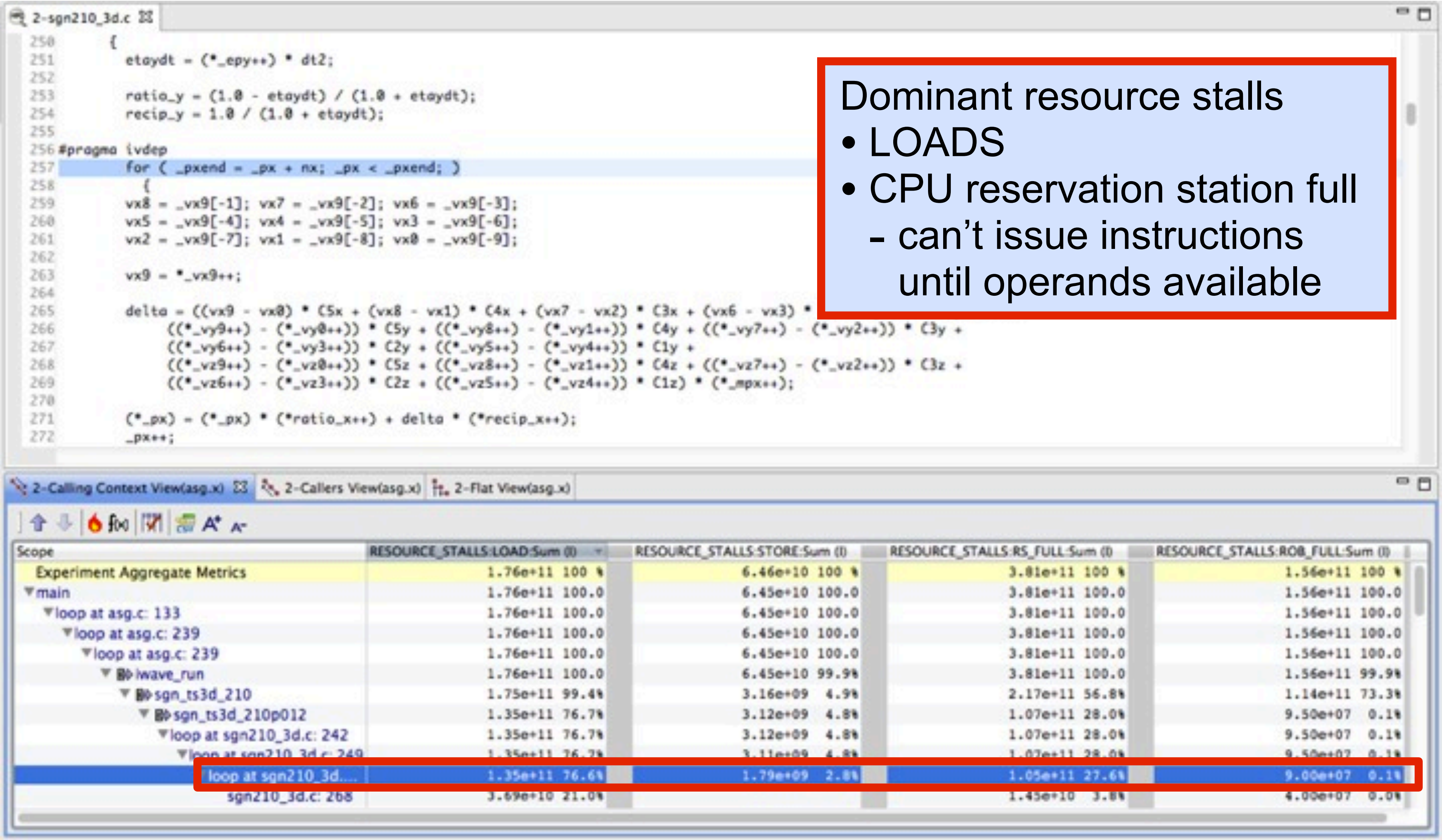
IWAVE Stencil - Why Low Performance?

- **Look at LLC misses to see demand fetch from memory**
- **Survey resource stalls from any source**

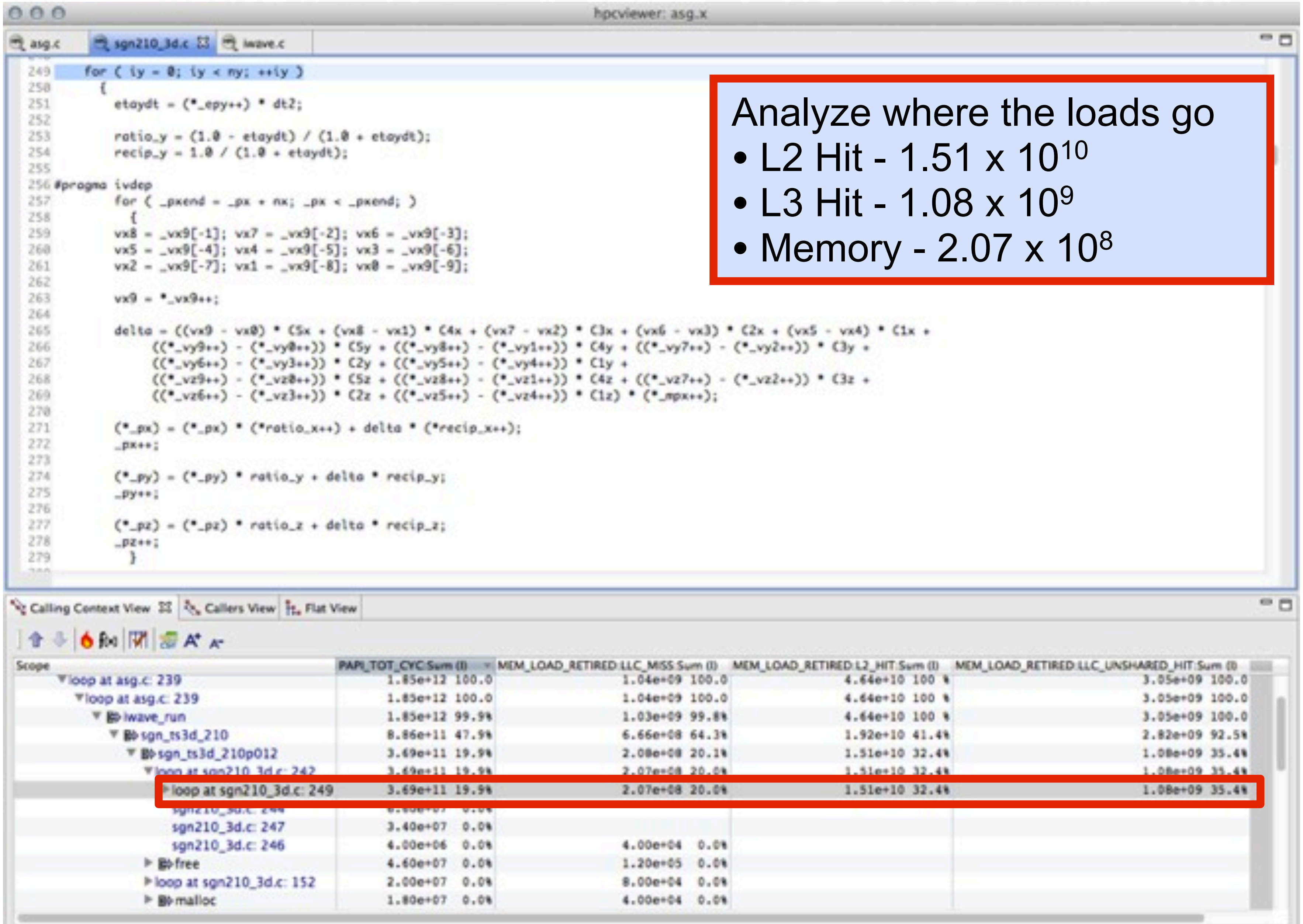


- LLC misses 3 orders of magnitude lower than cycles
- Resource stalls on par with total cycles

IWAVE - Looking at Resource Stall Causes



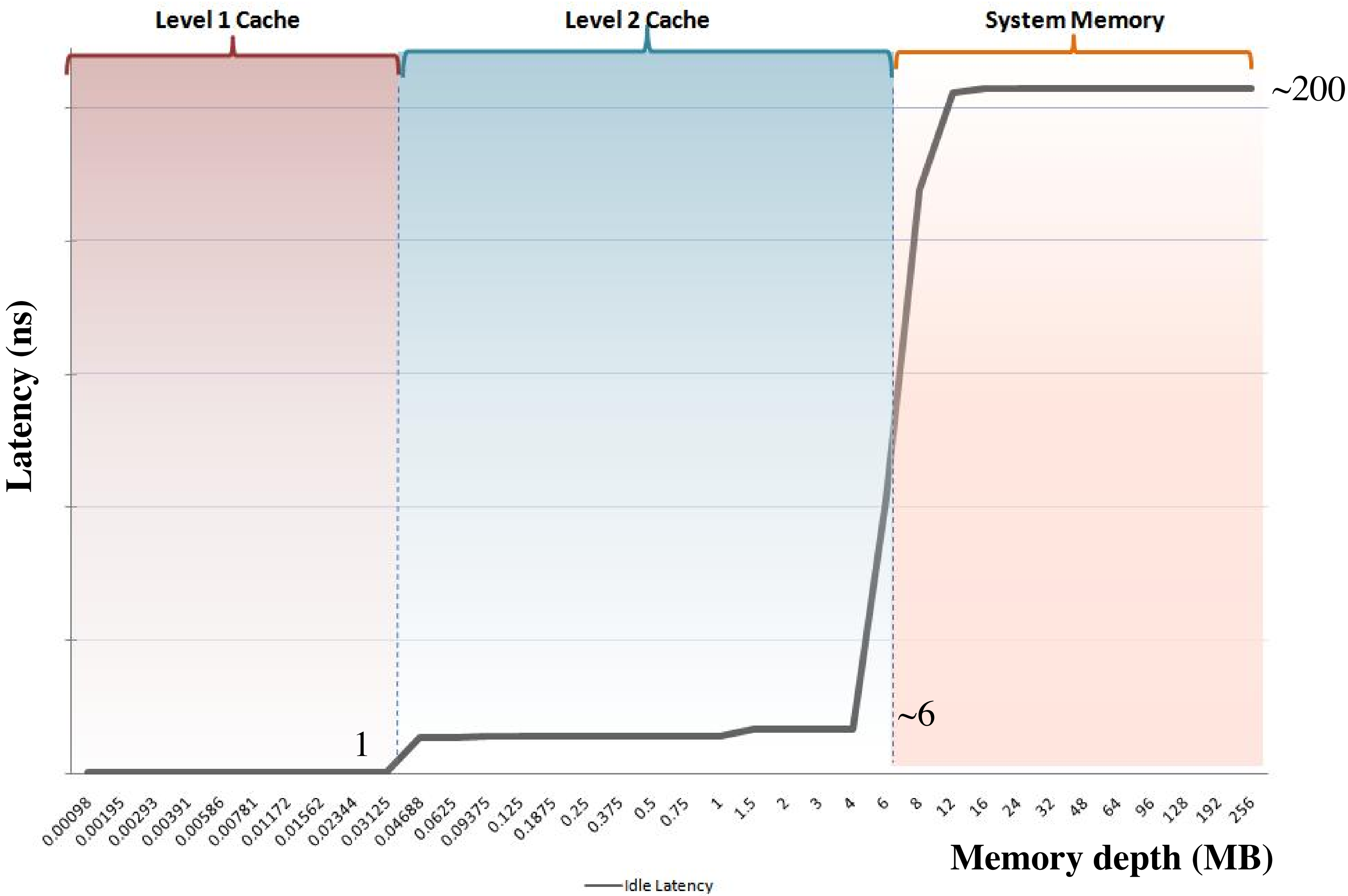
IWAVE - Looking at Memory System Usage



Analyze where the loads go

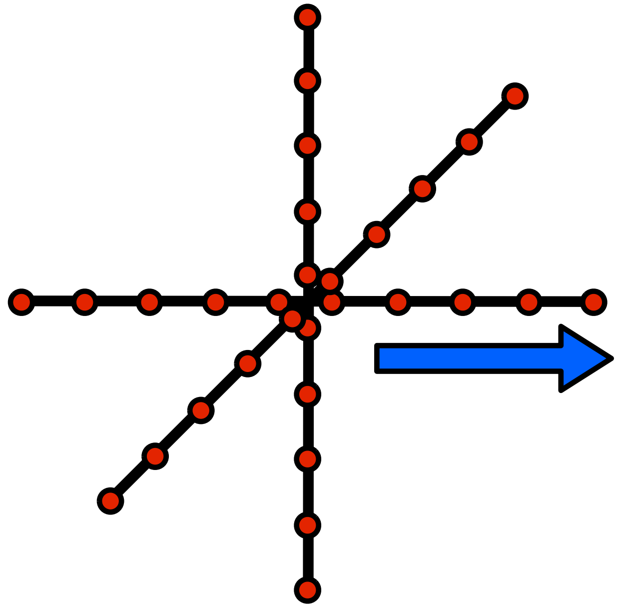
- L2 Hit - 1.51×10^{10}
- L3 Hit - 1.08×10^{9}
- Memory - 2.07×10^{8}

Memory Latency on Intel 5100 MCH



Principal Stencil Pattern

- **Execution under study**
  - — **sgn_ts3d_210p012**
    - – **10 points along each coordinate axis**
    - – **sweep through memory along the X coordinate dimension**

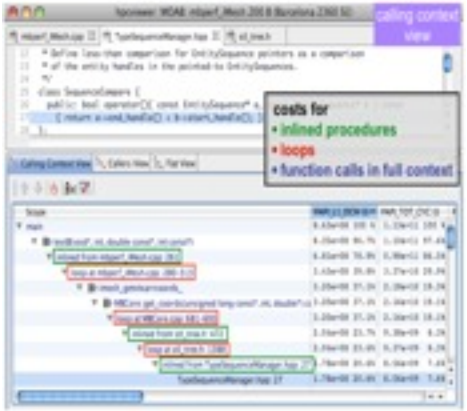


IWAVE Tuning Recommendations

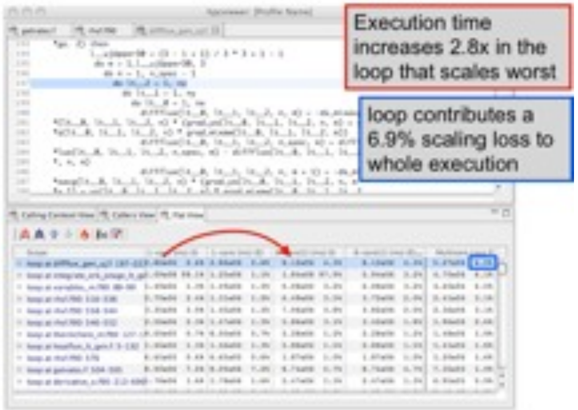
- Computation vs. communication
  - — communication for the example studied is ~27% of iwave_run
    - – compute on more data per core for higher parallel efficiency
  - — no communication/computation overlap
- I/O
  - — IWAVE uses serial Posix I/O for its input
  - — using HDF5 and parallel I/O would be a higher performance choice
- Stencil calculations
  - — IWAVE's stencil calculations are latency bound
    - – spend most of their time waiting for data from L2 cache
  - — need to make better use of the memory hierarchy

    - – unrolling once in Y and Z coordinate dimensions will reuse data values immediately
      - currently, temporal reuse along Y and Z axis is long distance
      - unrolling in Y and Z: immediately reuse 9 of every 10 values loaded
    - – pointer-based data access inhibits compiler-based tiling
      - tiling along Y and Z will be important for good cache reuse with large data

HPCToolkit Capabilities at a Glance

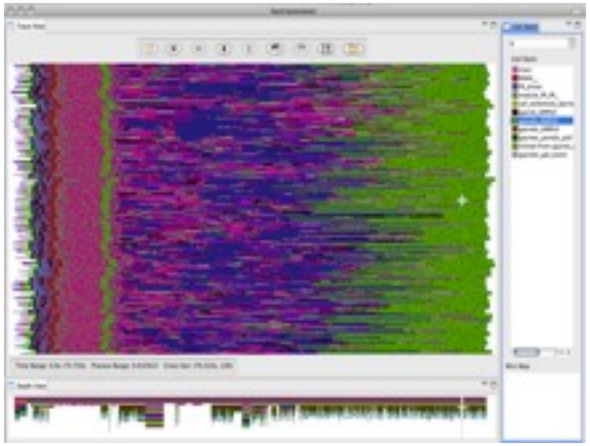


Attribute Costs to Code

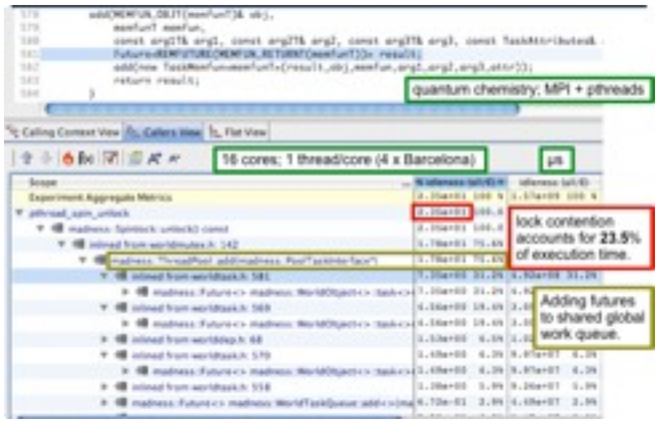


Pinpoint & Quantify Scaling Bottlenecks

Assess Imbalance and Variability



Analyze Behavior over Time



Shift Blame from Symptoms to Causes

Associate Costs with Data

RICE

hpctoolkit.org

HPCToolkit Status

- Operational today on
  - 64- and 32-bit x86 systems running Linux (including Cray XT/E/K)
  - IBM Blue Gene/P
  - IBM Power7 systems running Linux
- Emerging capabilities
  - IBM Blue Gene/Q
  - NVIDIA GPU
    - measurement and reporting using GPU hardware counters
  - data centric analysis
- Available as open source software at http://hpctoolkit.org

Ongoing Work

- Homogeneous nodes
  - measurement and analysis for massive numbers of threads
  - "blame shifting" to pinpoint and quantify causes of idleness in OpenMP programs
- Heterogeneous nodes
  - "blame shifting" to pinpoint and quantify causes of CPU and GPU idleness in hybrid programs
  - derived metrics for GPU
- Bandwidth monitoring of communication and I/O
- Future enhancements
  - support for Intel MIC
  - provide higher-level prescriptive feedback

References

- **HPCToolkit Project: http://hpctoolkit.org**
- **David Levinthal. Performance Analysis Guide for Intel® CoreTM i7 Processor and Intel® XeonTM 5500 processors, Version 1.0, http://software.intel.com/sites/products/collateral/hpc/vtune/performance_analysis_guide.pdf**