APEX: HPX/Phylanx Y2 Plans

(Autonomic Performance Environment for eXascale)

Kevin Huck, Mohammad Alaul Haque Monil,

Allen Malony

khuck@cs.uoregon.edu

http://github.com/khuck/xpress-apex

UNIVERSITY OF OREGON

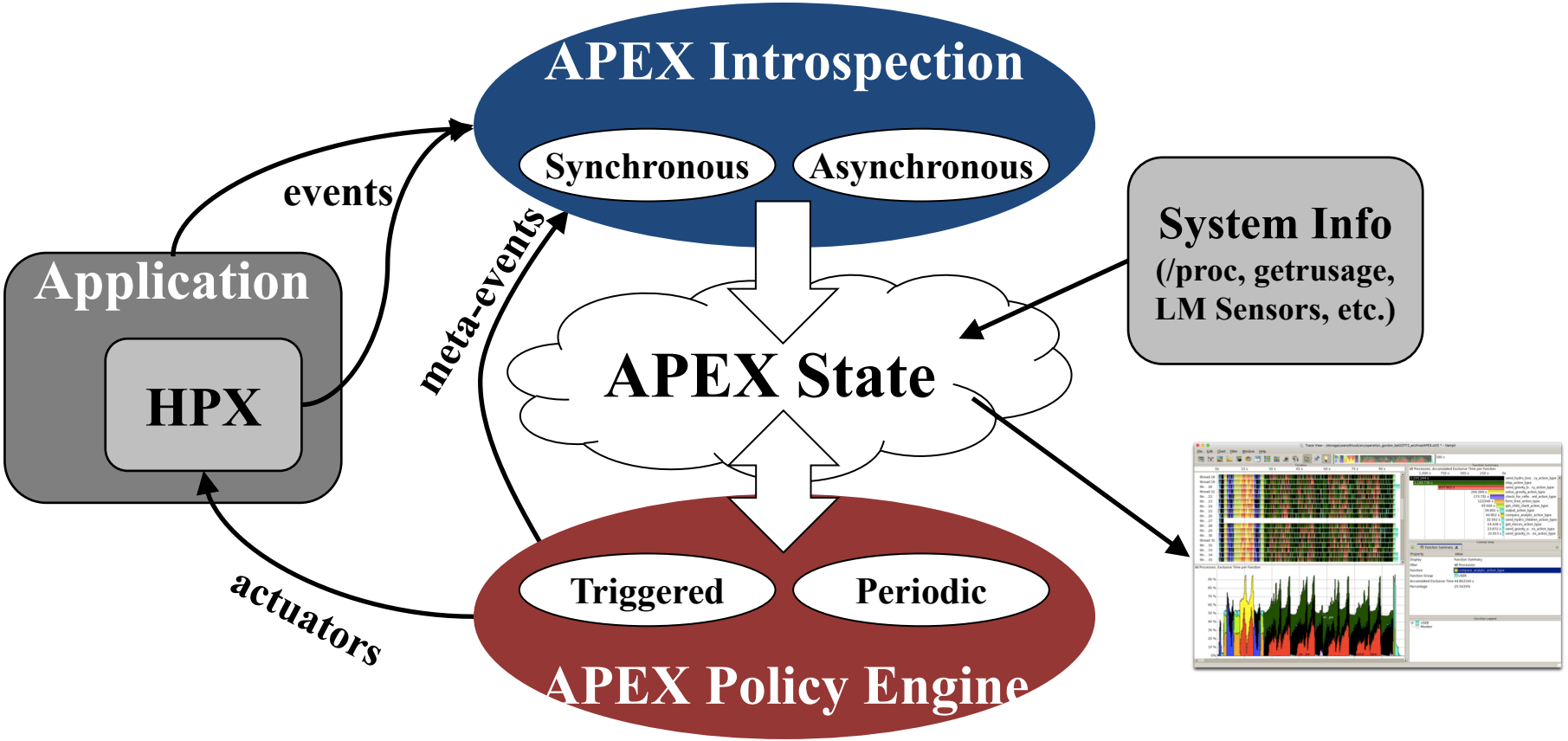
What is APEX?

- **Performance measurement library for distributed, asynchronous tasking models/runtimes**
  - i.e. HPX, but there are others
  - Lightweight measurement (tasks <1ms)
  - High concurrency (both OS threads and tasks in flight)
  - Distinction between OS and runtime (HPX) thread context
  - Lack of a traditional call stack
    - Task dependency chain instead
  - Runtime controlled task switching
- **Infrastructure for dynamic feedback and control of both the runtime and the application**

APEX Measurement

- **Top down** (application) and **bottom up** (OS/HW) performance mapping / feedback
  - Make node-wide resource utilization data and analysis, energy consumption, and health information available in real time
  - Associate performance state with policies for feedback control
- **APEX introspection**
  - OS: track system resources, utilization, job contention, overhead
  - Runtime (HPX): track threads, queues, concurrency, remote operations, parcels, memory management
  - Application timer / counter observation
- **Post-mortem performance analysis**
- **Integrated with HPX performance counters**

APEX Architecture



APEX Introspection

- **APEX collects data through "inspectors"**
  - *Synchronous* uses an event API and event "listeners"
    - Initialize, terminate, new thread – added to HPX runtime
    - Timer start, stop, yield*, resume* – integrated into HPX task scheduler
    - Custom events (meta-events)
  - *Asynchronous* do not rely on events, but occur periodically
    - Sampled values (counters from HPX)
- **APEX exploits access to performance data from lower stack components**
  - "Health" data through other interfaces (/proc/stat, /proc/cpuinfo, /proc/meminfo, /proc/net/dev, /proc/self/status, lm_sensors, power*, PAPI hardware counters, etc.)

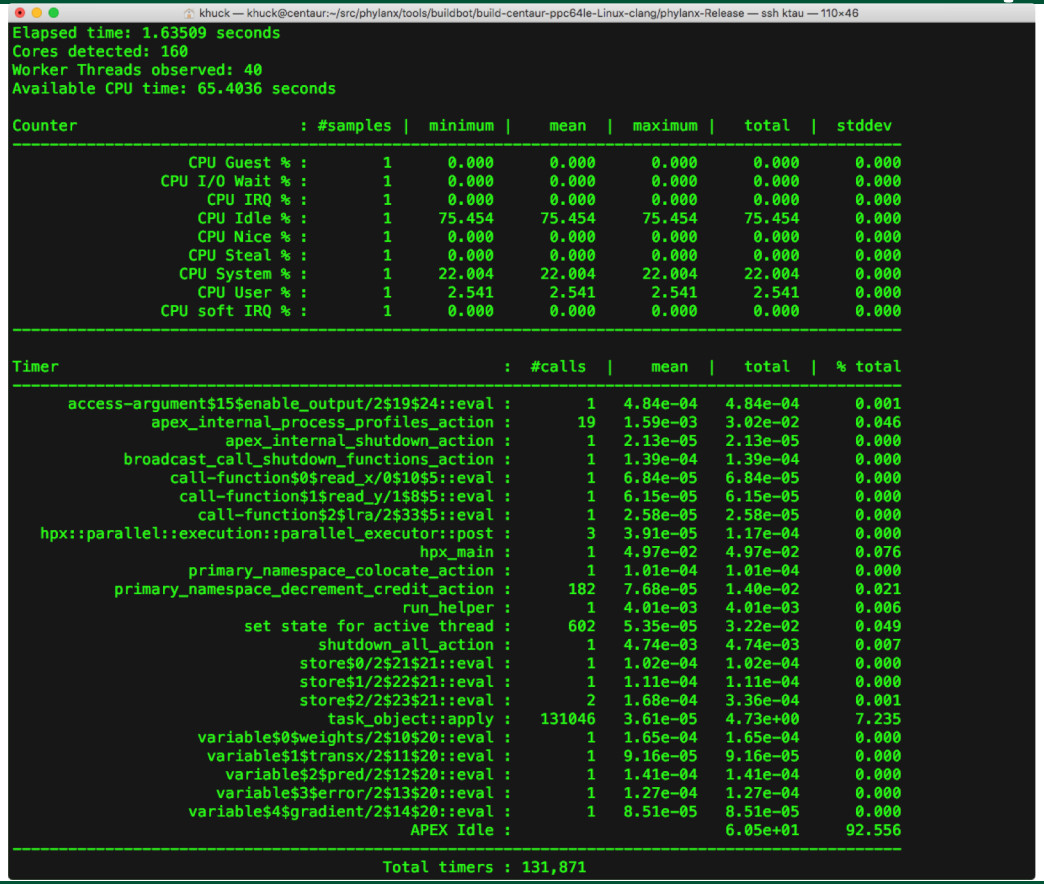
APEX Event Listeners

- **Profiling listener**
  - New task event: capture parent task relationship
  - Start event: input name/address, get timestamp, return profiler handle
  - Stop event: get timestamp, put profiler object in a queue for back-end processing, return
  - Sample event: put the name & value in the queue
  - Asynchronous consumer thread: process profiler objects and samples to build statistical profile (in HPX, processed/scheduled as a thread/task)
- **TAU Listener (postmortem analysis)**
  - Synchronously passes all measurement events to TAU to build an offline profile/trace
- **OTF2 Listener (postmortem analysis)**
  - Synchronously passes all measurement events to libotf2 for trace analysis
- **Concurrency listener (postmortem analysis)**
  - Start event: push timer ID on stack
  - Stop event: pop timer ID off stack
  - Asynchronous consumer thread: periodically log current timer for each thread, output report at termination

APEX Policy Listener

- ***Policies*** are rules that decide on outcomes based on observed state
  - *Triggered* policies are invoked by introspection API events
  - *Periodic* policies are run periodically on asynchronous thread
- Polices are registered with the Policy Engine
  - Applications, runtimes, and/or OS register callback functions
- Callback functions define the policy rules
  - "If x < y then…" – any arbitrary logic
- Enables runtime adaptation using introspection data
  - Feedback and control mechanism
  - Engages actuators across stack layers
  - Could also be used to involve online auto-tuning/online-search support
    - Active Harmony http://www.dyninst.org/harmony
    - Hill climbing, etc.
    - Control theory (e.g. concurrency)

Screen Output

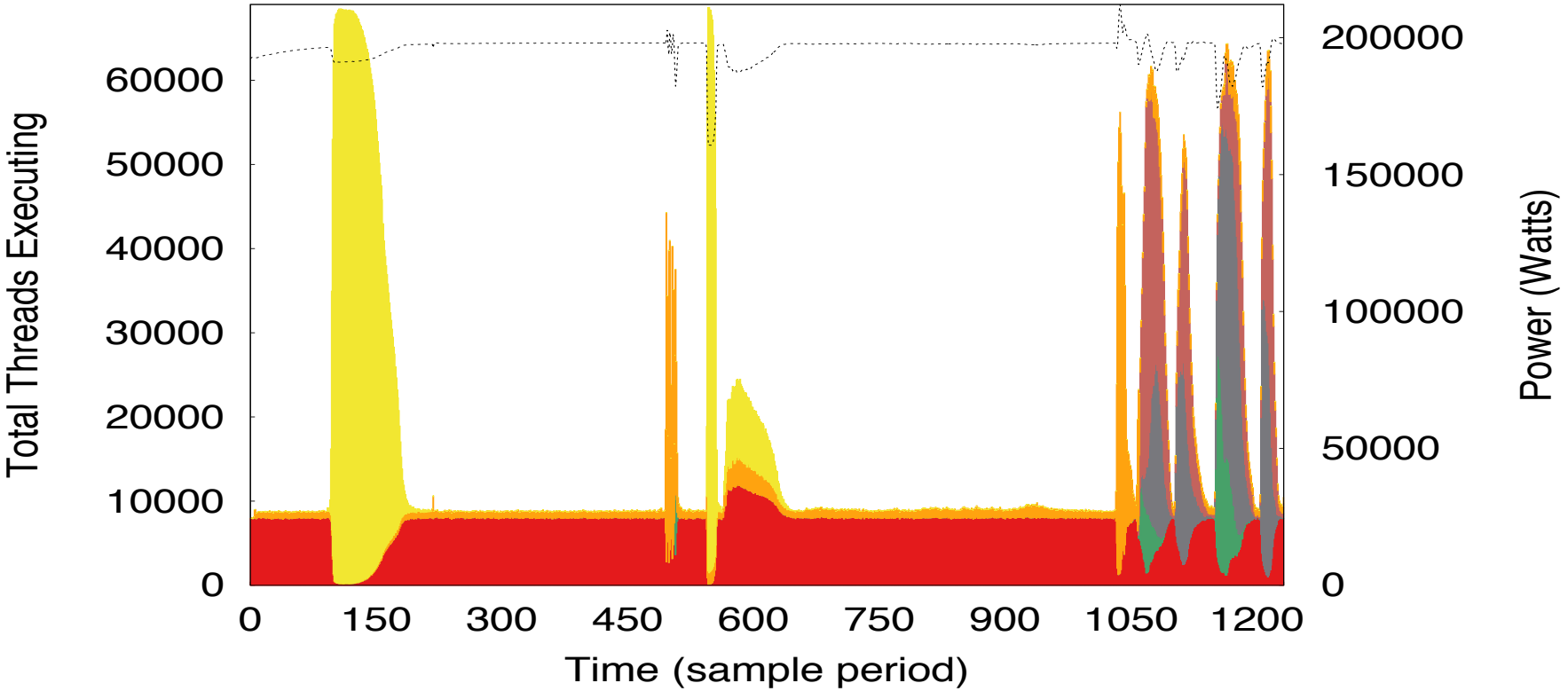


```
khuck — khuck@centaur:~/src/phylanx/tools/buildbot/build-centaur-ppc64le-Linux-clang/phylanx-Release — ssh ktau — 110×46
Elapsed time: 1.63509 seconds
Cores detected: 160
Worker Threads observed: 40
Available CPU time: 65.4036 seconds

Counter                      : #samples |  minimum |     mean |  maximum |    total |  stddev
-------------------------------------------------------------------------------------------
            CPU Guest % :          1       0.000      0.000      0.000      0.000      0.000
          CPU I/O Wait % :          1       0.000      0.000      0.000      0.000      0.000
              CPU IRQ % :          1       0.000      0.000      0.000      0.000      0.000
             CPU Idle % :          1      75.454     75.454     75.454     75.454      0.000
             CPU Nice % :          1       0.000      0.000      0.000      0.000      0.000
            CPU Steal % :          1       0.000      0.000      0.000      0.000      0.000
           CPU System % :          1      22.004     22.004     22.004     22.004      0.000
             CPU User % :          1       2.541      2.541      2.541      2.541      0.000
         CPU soft IRQ % :          1       0.000      0.000      0.000      0.000      0.000
-------------------------------------------------------------------------------------------

Timer                                                : #calls  |    mean  |   total  | % total
-------------------------------------------------------------------------------------------
      access-argument$15$enable_output/2$19$24::eval :        1   4.84e-04   4.84e-04      0.001
                apex_internal_process_profiles_action :       19   1.59e-03   3.02e-02      0.046
                        apex_internal_shutdown_action :        1   2.13e-05   2.13e-05      0.000
           broadcast_call_shutdown_functions_action :        1   1.39e-04   1.39e-04      0.000
                call-function$0$read_x/0$10$5::eval :        1   6.84e-05   6.84e-05      0.000
                 call-function$1$read_y/1$8$5::eval :        1   6.15e-05   6.15e-05      0.000
                  call-function$2$lra/2$33$5::eval :        1   2.58e-05   2.58e-05      0.000
   hpx::parallel::execution::parallel_executor::post :        3   3.91e-05   1.17e-04      0.000
                                          hpx_main :        1   4.97e-02   4.97e-02      0.076
                 primary_namespace_colocate_action :        1   1.01e-04   1.01e-04      0.000
         primary_namespace_decrement_credit_action :      182   7.68e-05   1.40e-02      0.021
                                       run_helper :        1   4.01e-03   4.01e-03      0.006
                    set state for active thread :      602   5.35e-05   3.22e-02      0.049
                            shutdown_all_action :        1   4.74e-03   4.74e-03      0.007
                          store$0/2$21$21::eval :        1   1.02e-04   1.02e-04      0.000
                          store$1/2$22$21::eval :        1   1.11e-04   1.11e-04      0.000
                          store$2/2$23$21::eval :        2   1.68e-04   3.36e-04      0.001
                            task_object::apply :   131046   3.61e-05   4.73e+00      7.235
                 variable$0$weights/2$10$20::eval :        1   1.65e-04   1.65e-04      0.000
                  variable$1$transx/2$11$20::eval :        1   9.16e-05   9.16e-05      0.000
                   variable$2$pred/2$12$20::eval :        1   1.41e-04   1.41e-04      0.000
                  variable$3$error/2$13$20::eval :        1   1.27e-04   1.27e-04      0.000
                variable$4$gradient/2$14$20::eval :        1   8.51e-05   8.51e-05      0.000
                                       APEX Idle :                       6.05e+01     92.556
-------------------------------------------------------------------------------------------
                                  Total timers : 131,871
```

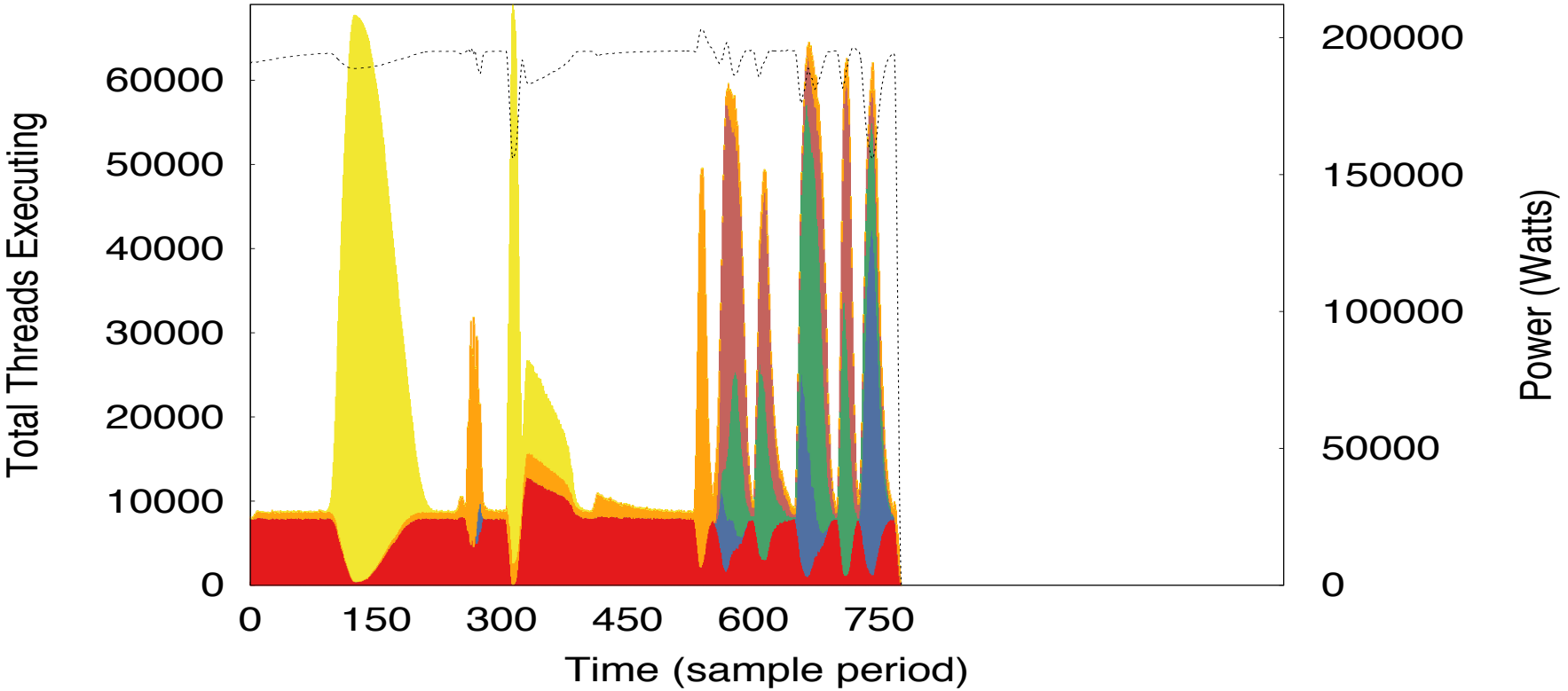
APEX, HPX counters

APEX timers

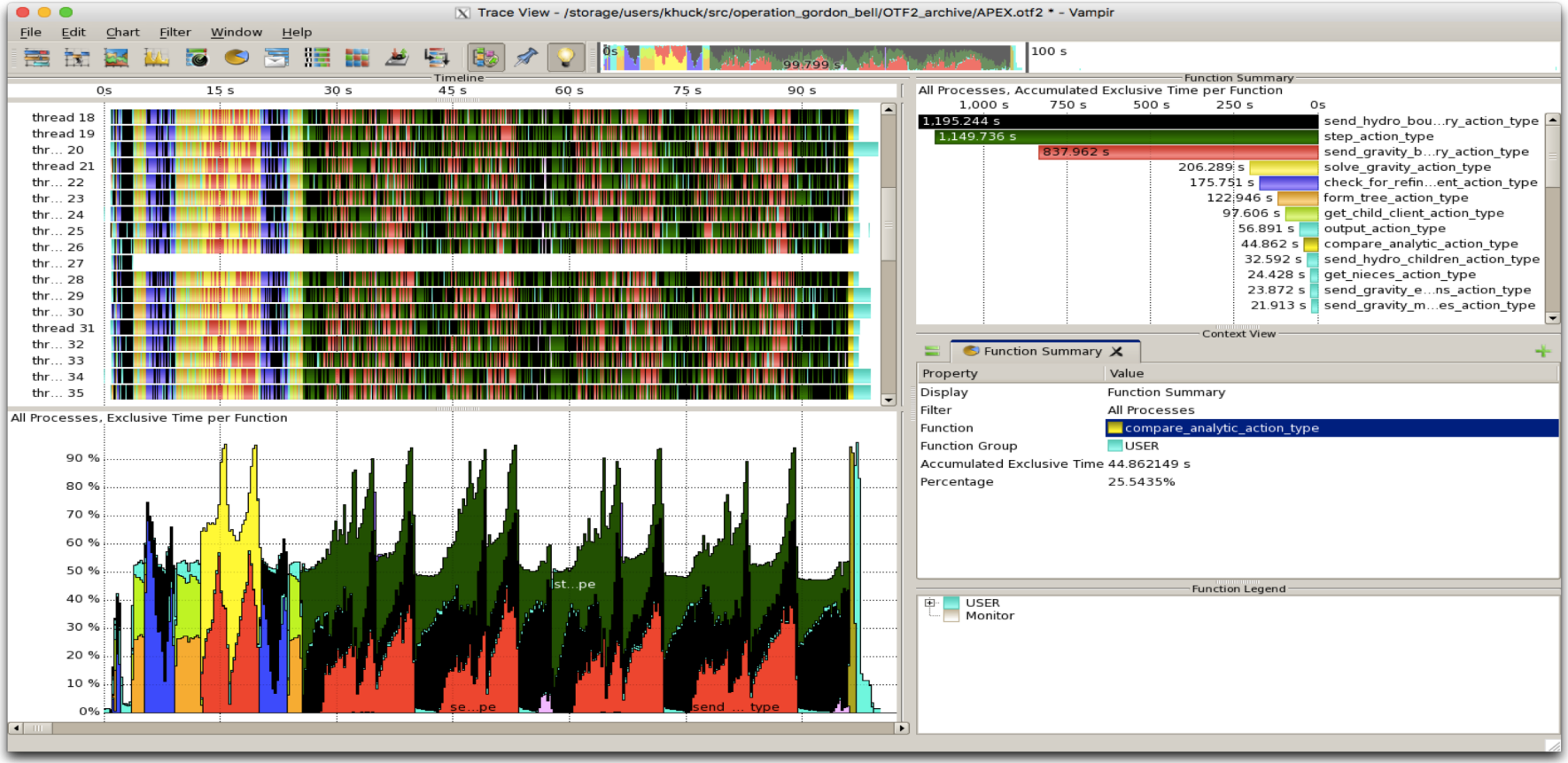
Concurrency View



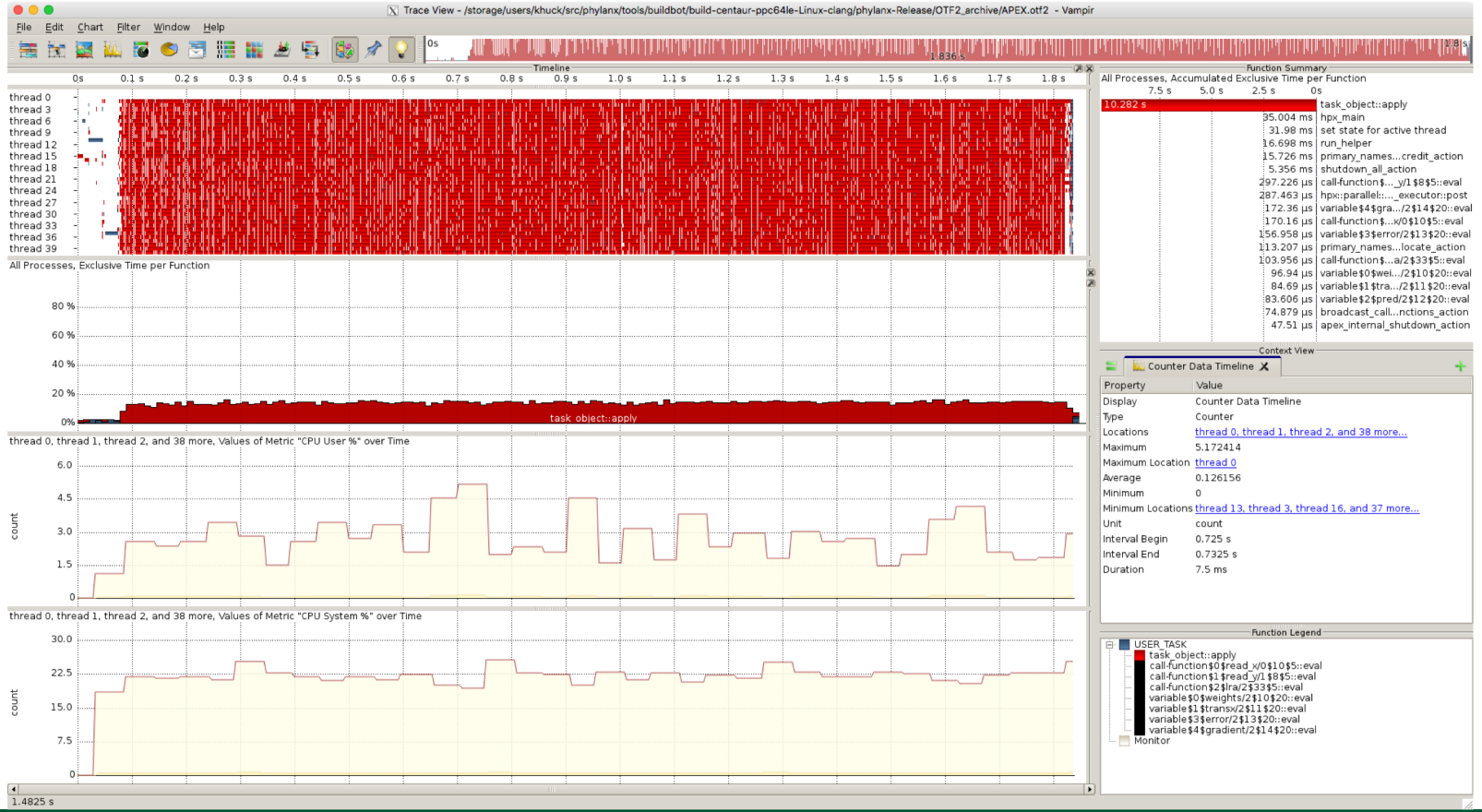
Concurrency View



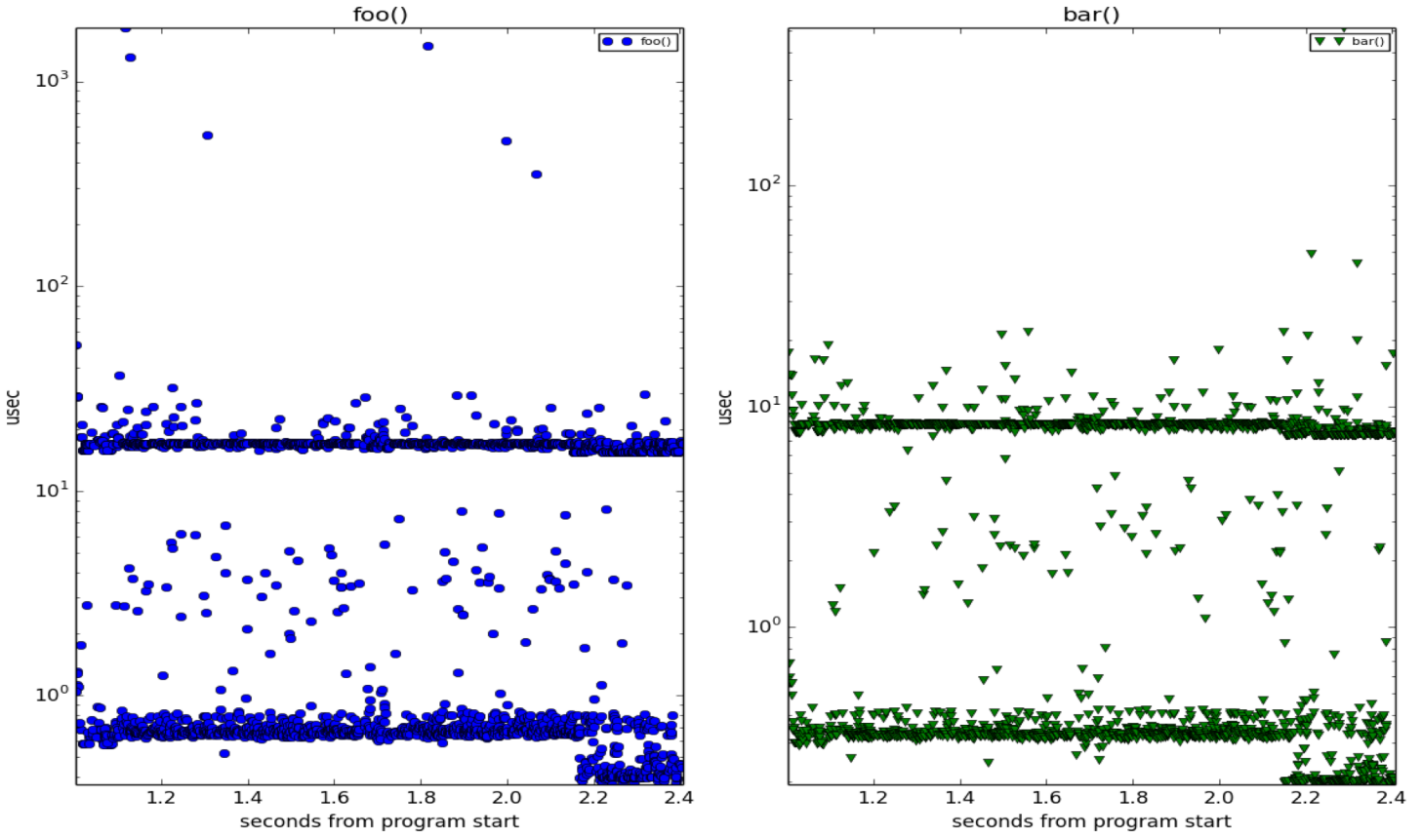
OTF2 View in Vampir



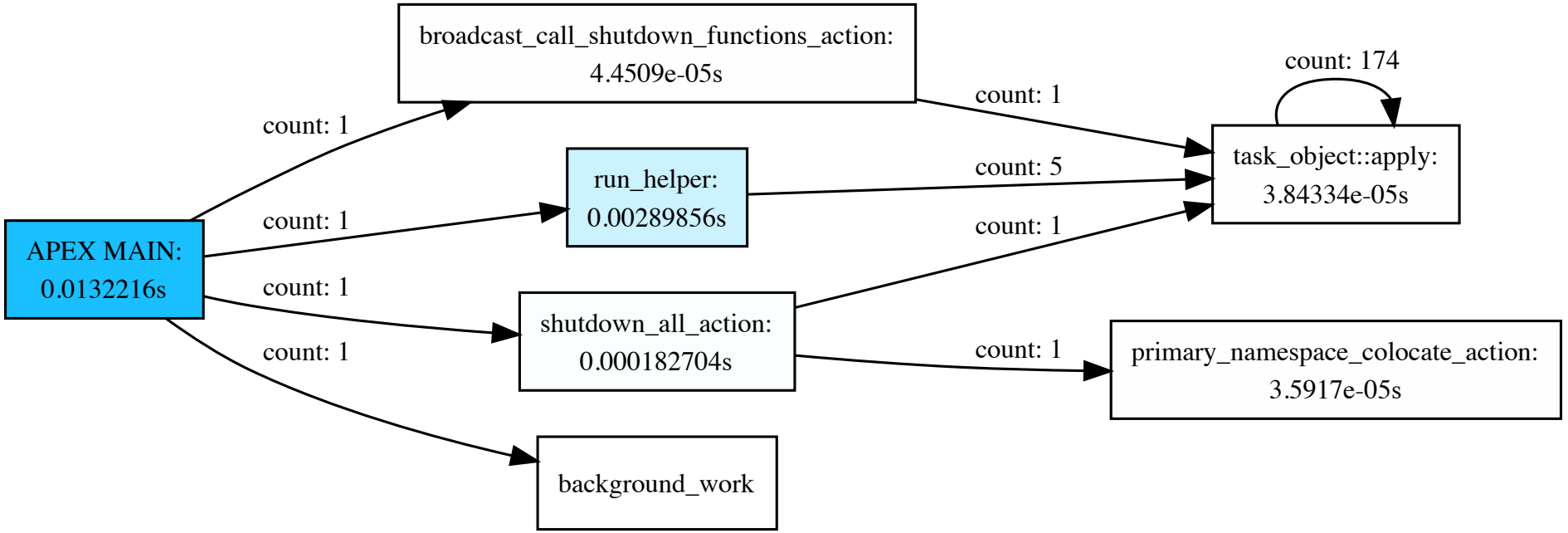
OTF2 View in Vampir - Phylanx



Task Scatterplot Analysis



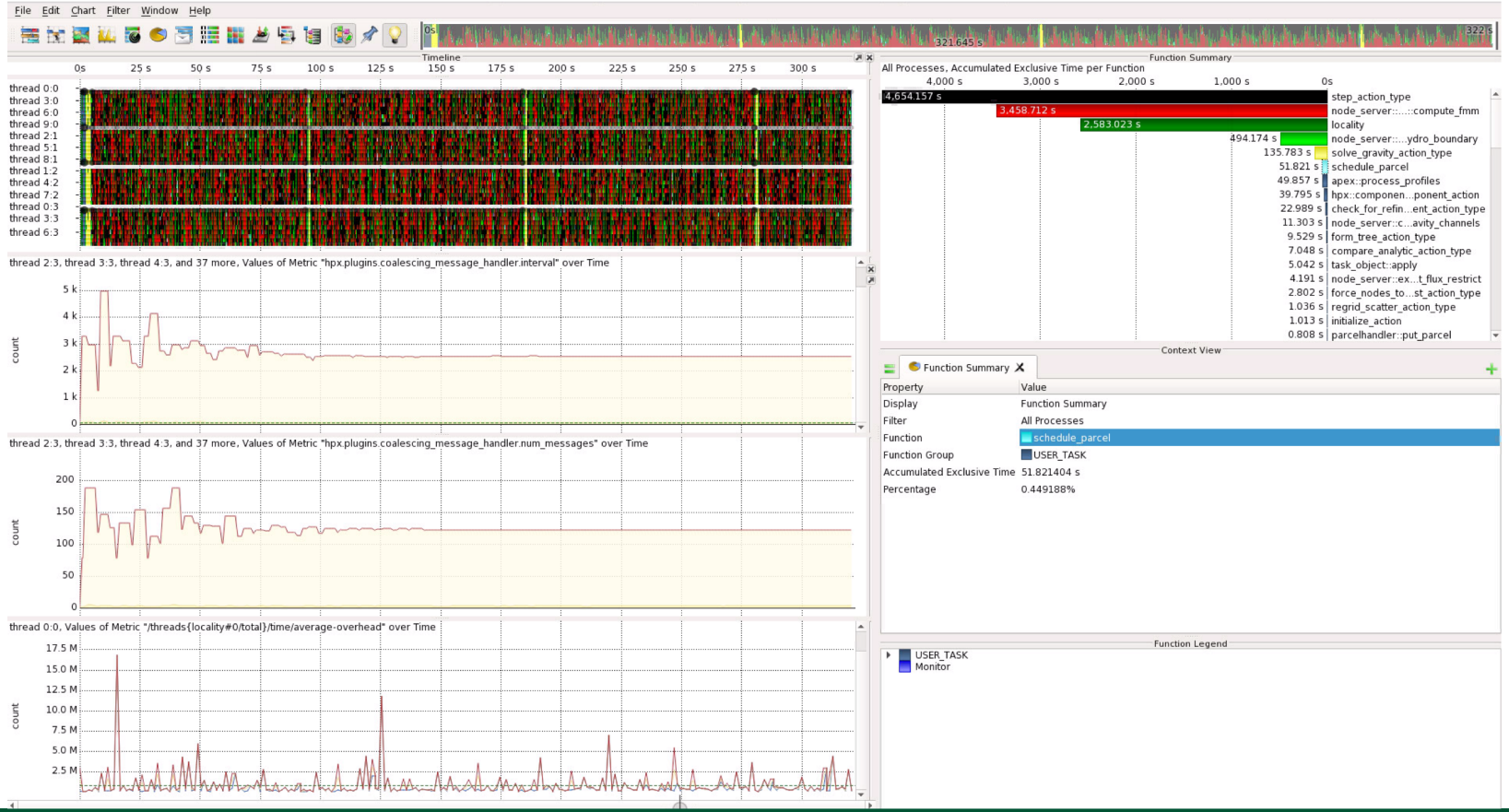
Taskgraph View



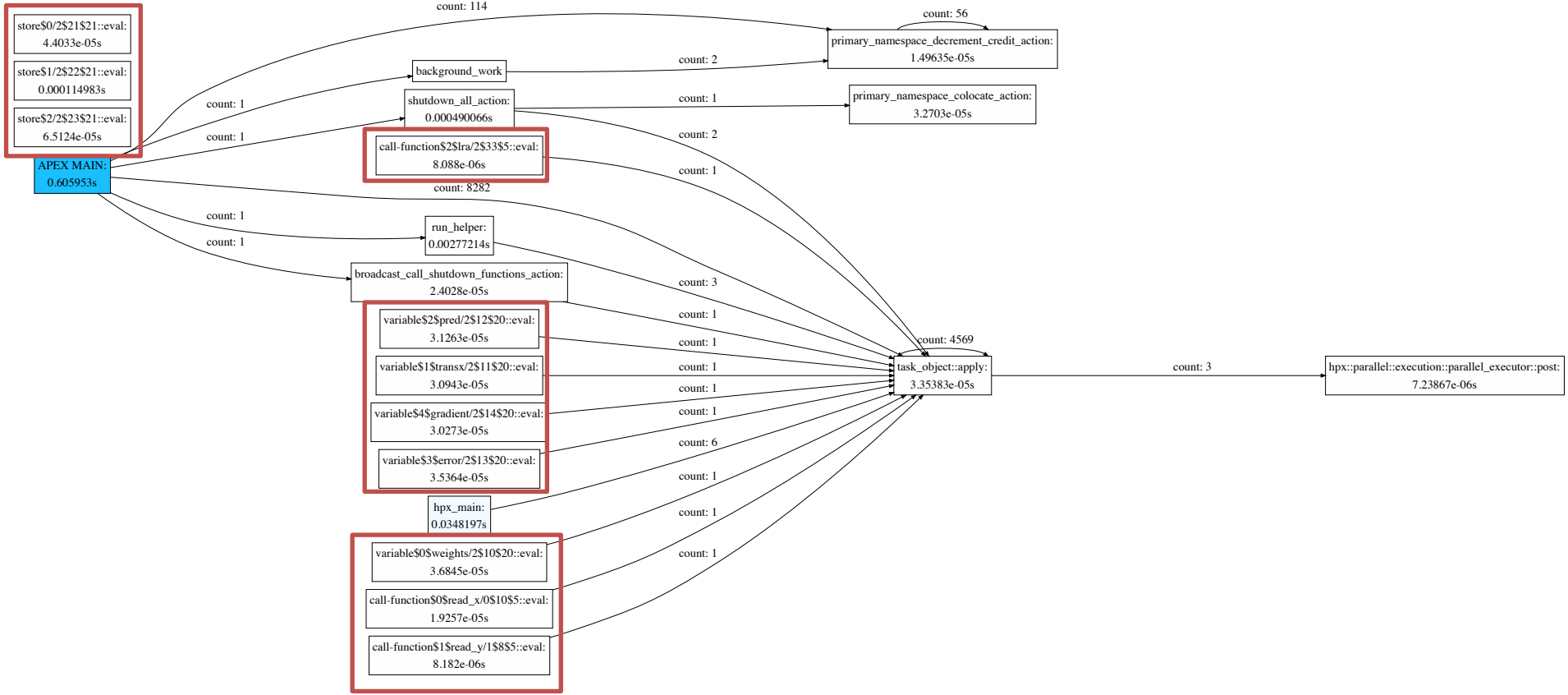
Next Steps

- Phylanx Policies
  - Direct actions vs. scheduled tasks
  - Algorithmic parameter optimization
- HPX Policies
  - Parcel coalescing
  - Auto-chunking / Parallel Algorithms
  - Global performance view
- Phylanx counter integration
  - Add as APEX counters
  - AST tree measurements
- Task dependency analysis
  - APEX taskgraph combined with Phylanx AST graph
  - OTF2 task dependency analysis
  - HPX “states”
- Updated documentation
  - Quickstart
  - Howto
  - FAQ
  - User/Reference guides
    - http://khuck.github.io/xpress-apex/

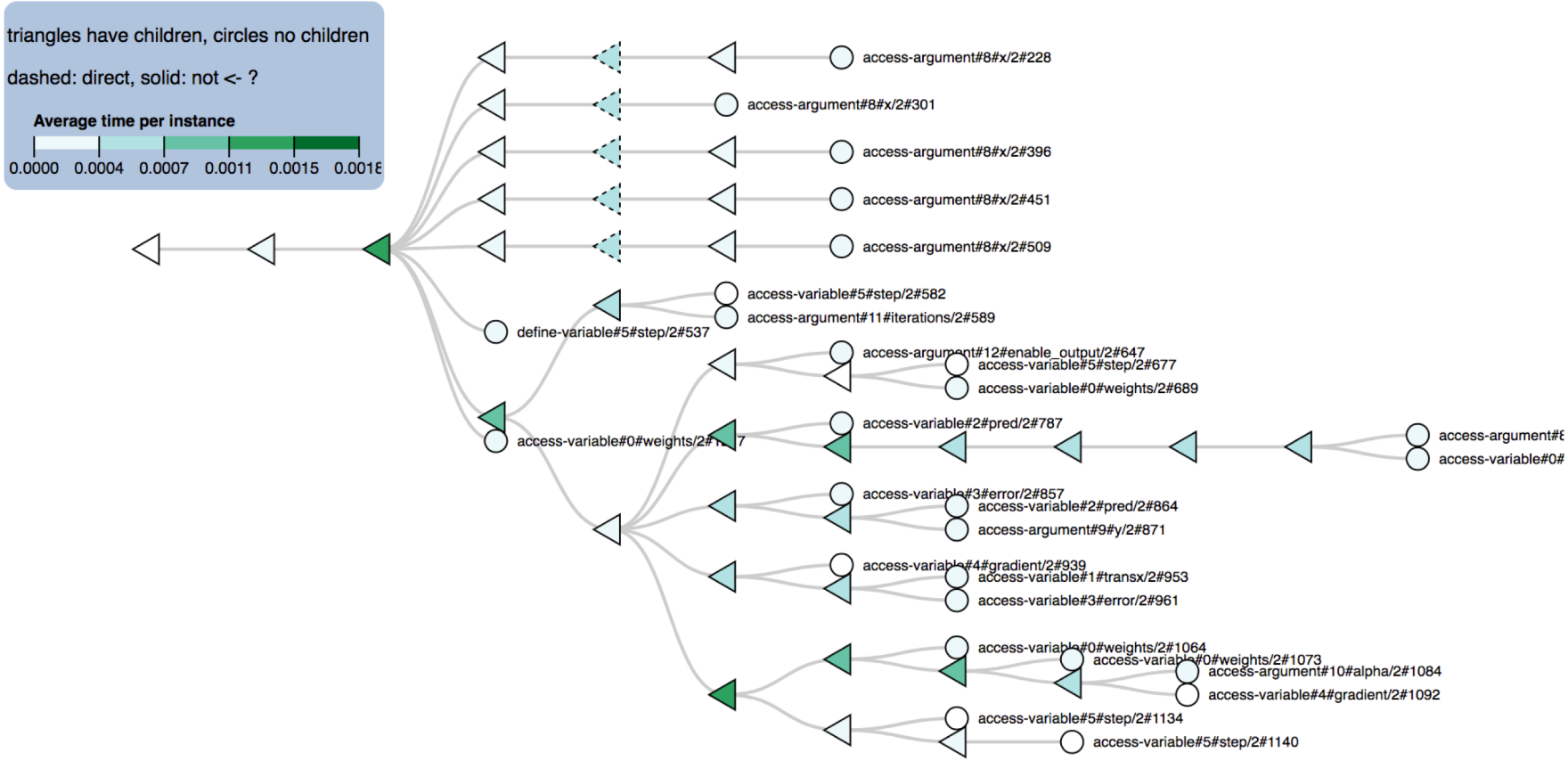
Policy Example: Parcel Coalescing



Task Graph Example : lra_csv



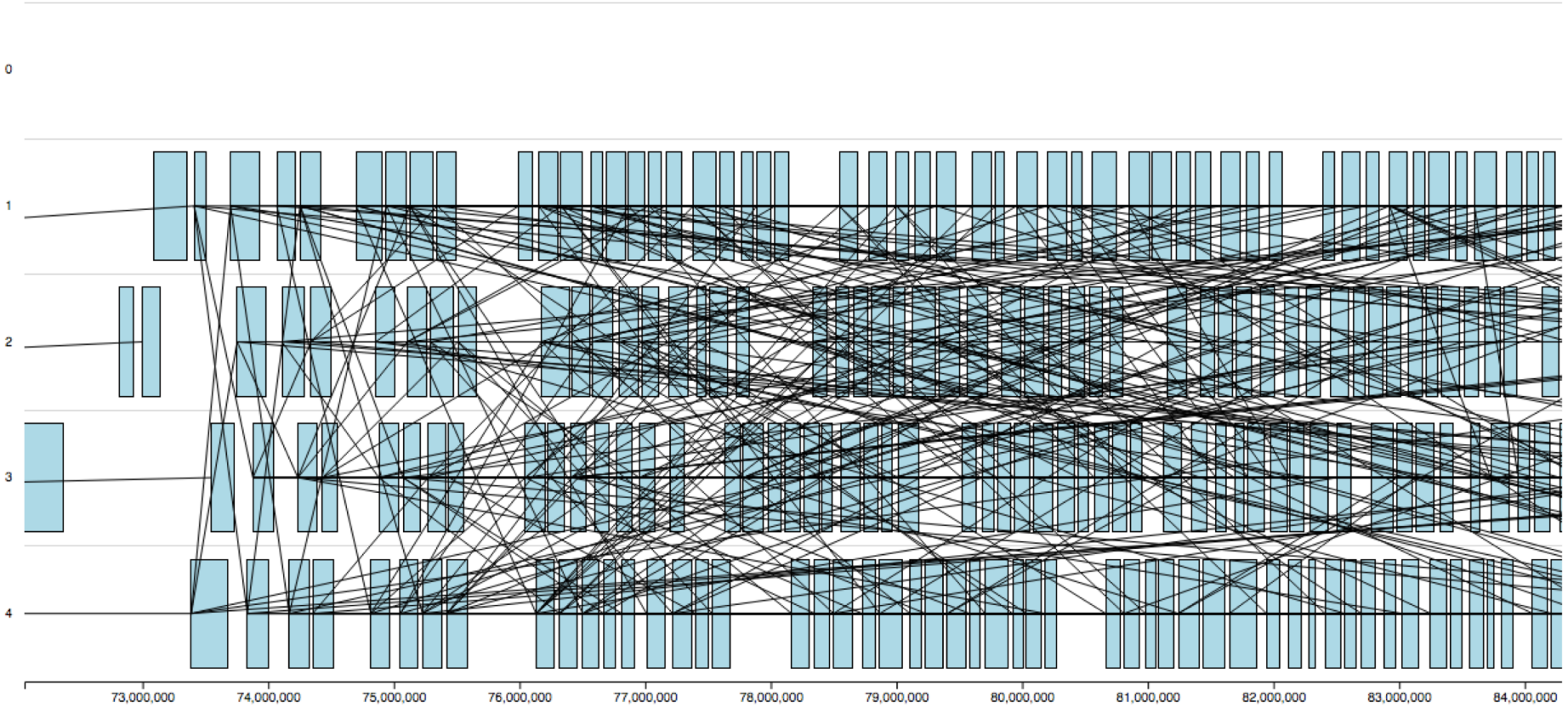
Phylanx AST (old data/graph)



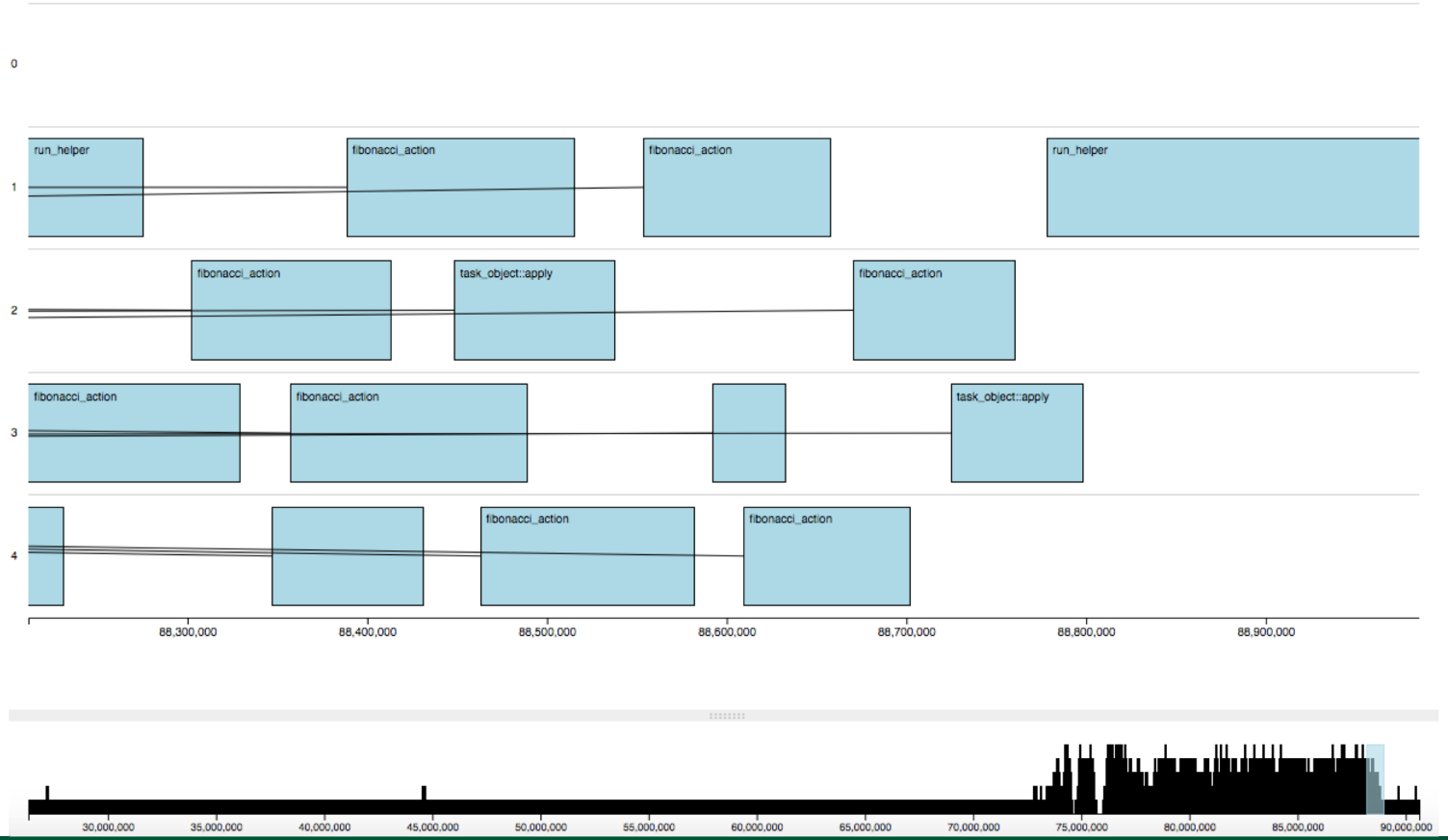
Task dependency analysis

- Previous work (see following slides) performed by extending OTF2 and building custom analysis and viewer
- Current OTF2 modifications: adding task GUID and parent GUID attributes to timer events captures dependencies
- Options:
  - Update existing viewer to work with conventional OTF2 with attributes
  - Add analysis algorithm to “viewer 2.0” (née Ravel)

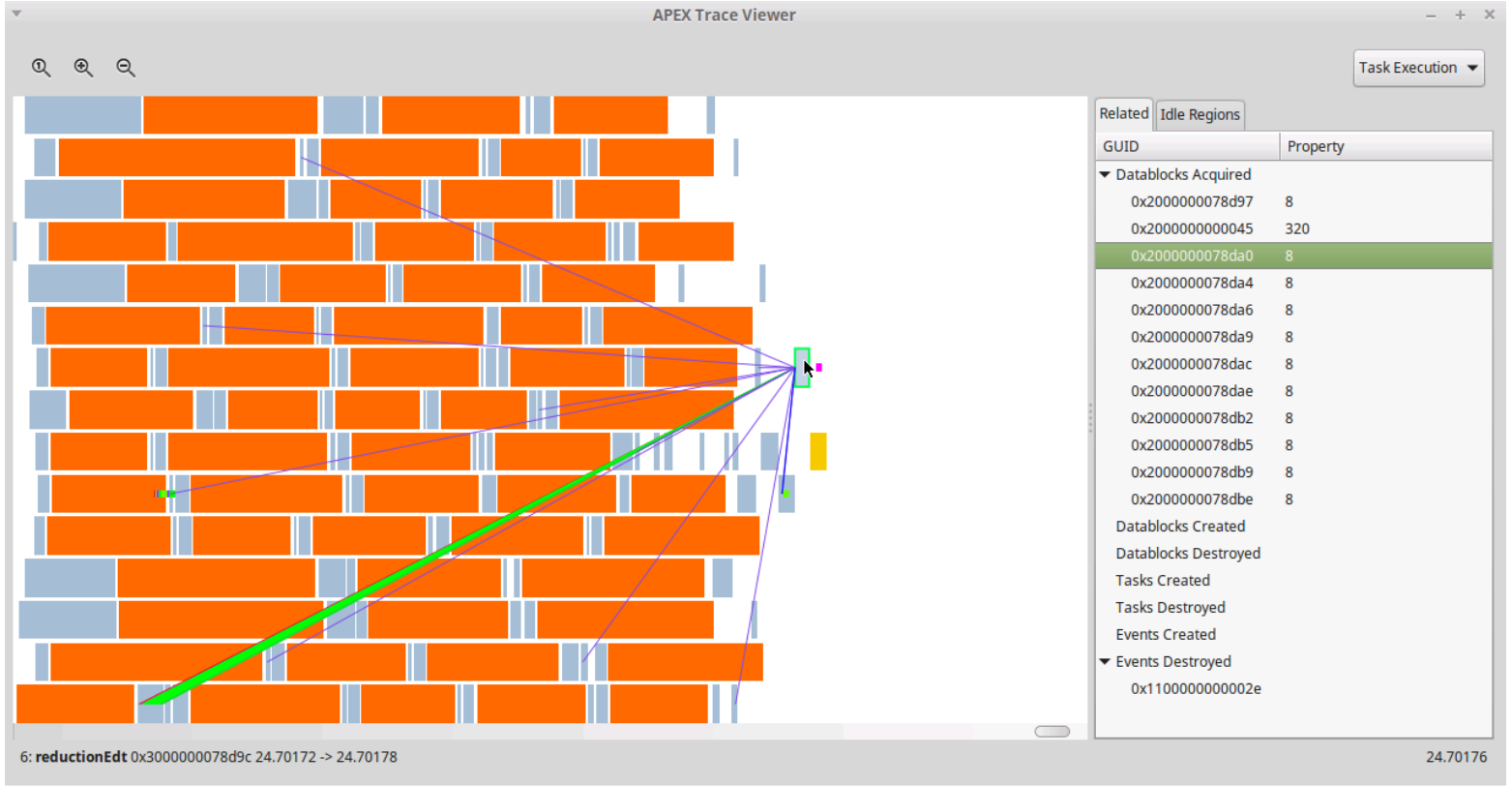
Task dependency in new viewer



Zoom to end of computation

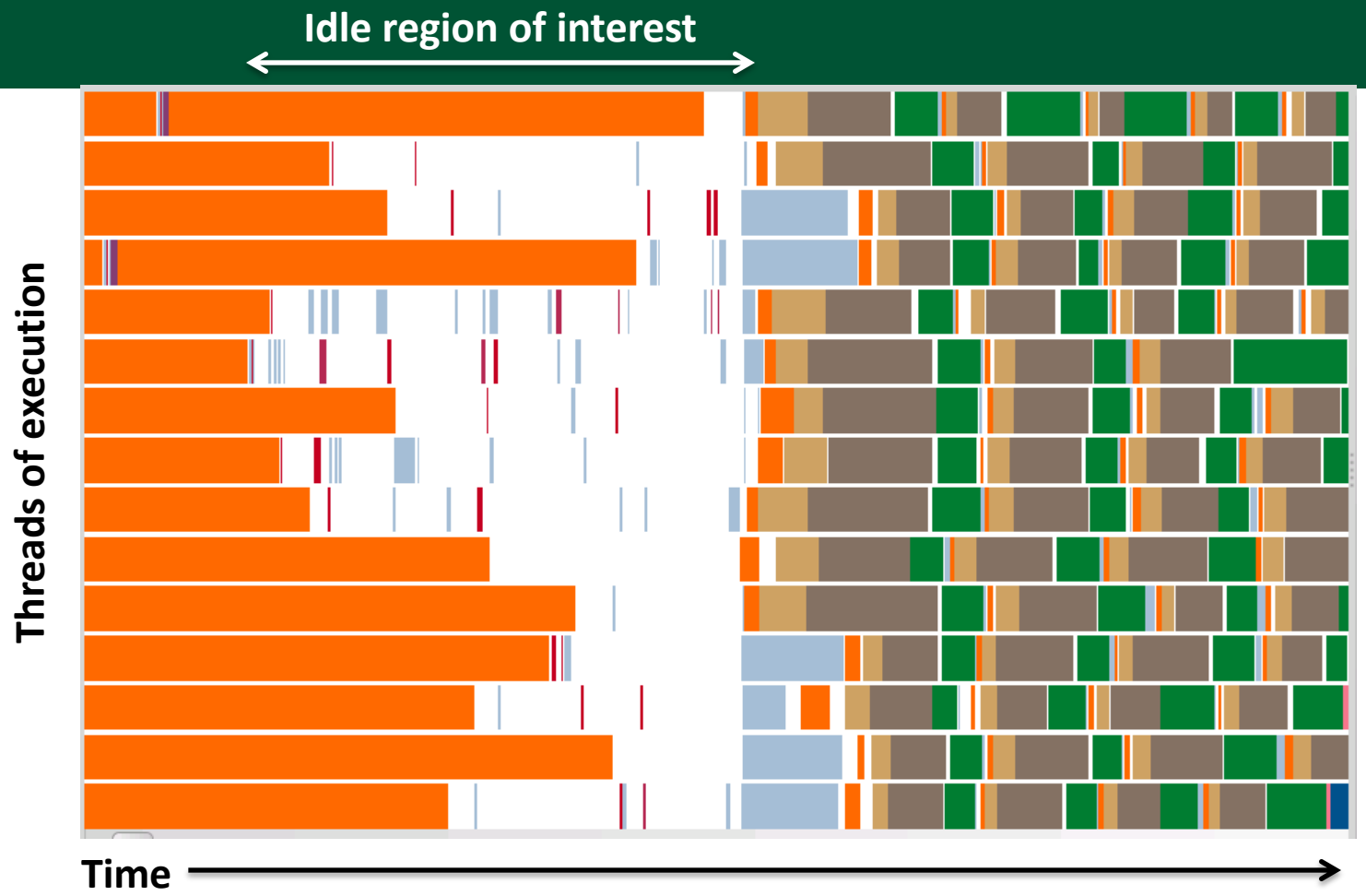


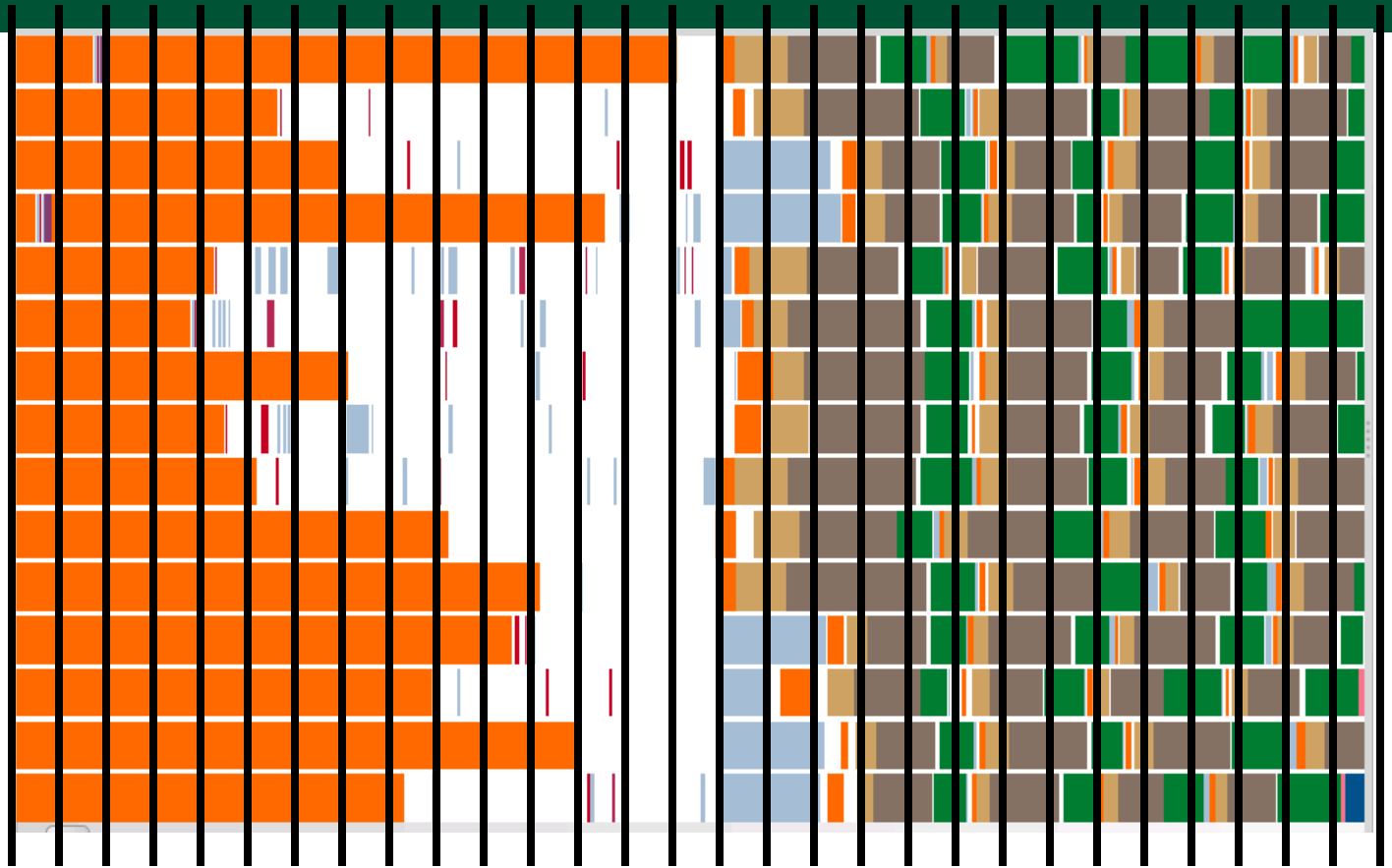
HPCG application from “another runtime”

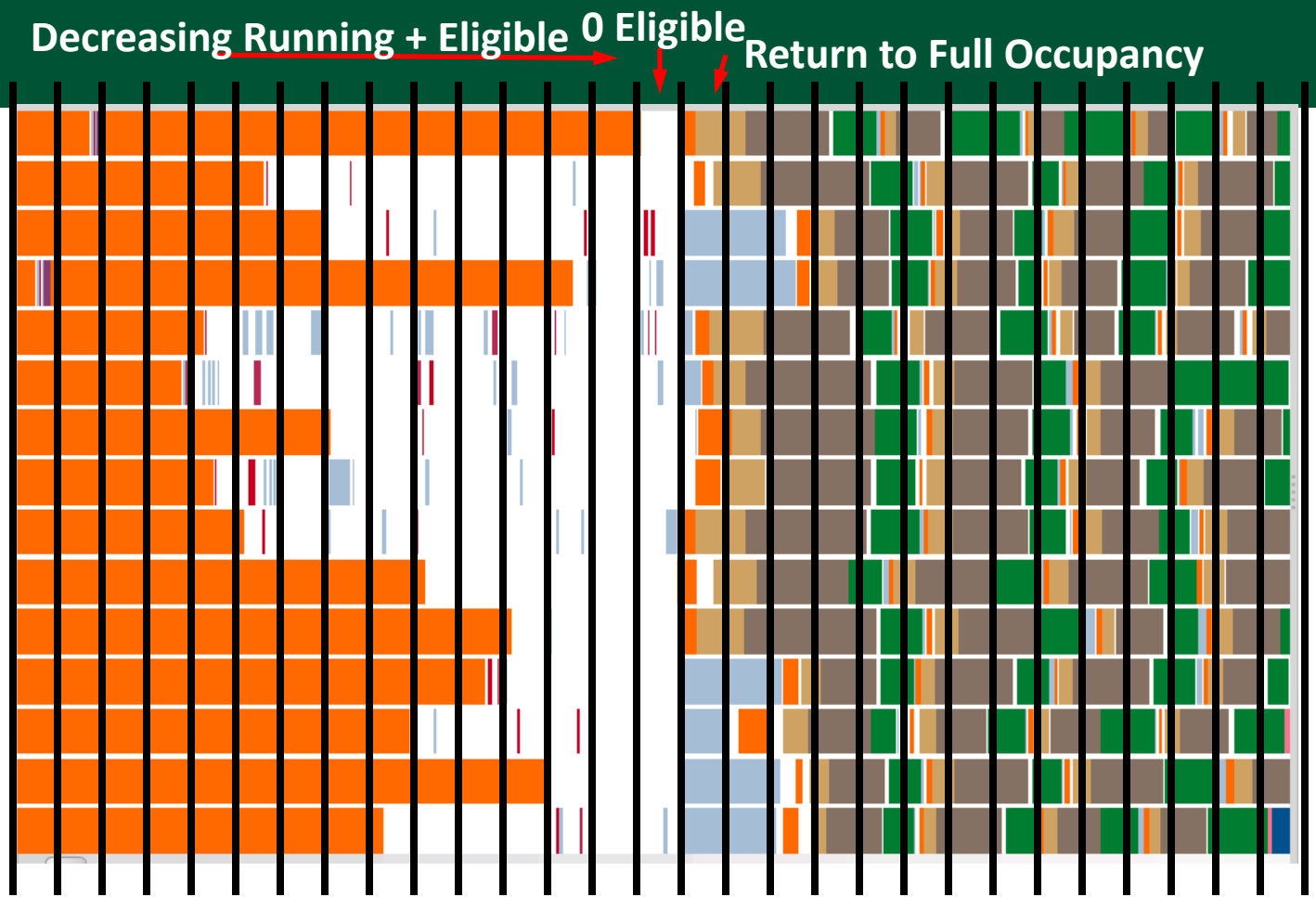


Automated Performance Diagnosis

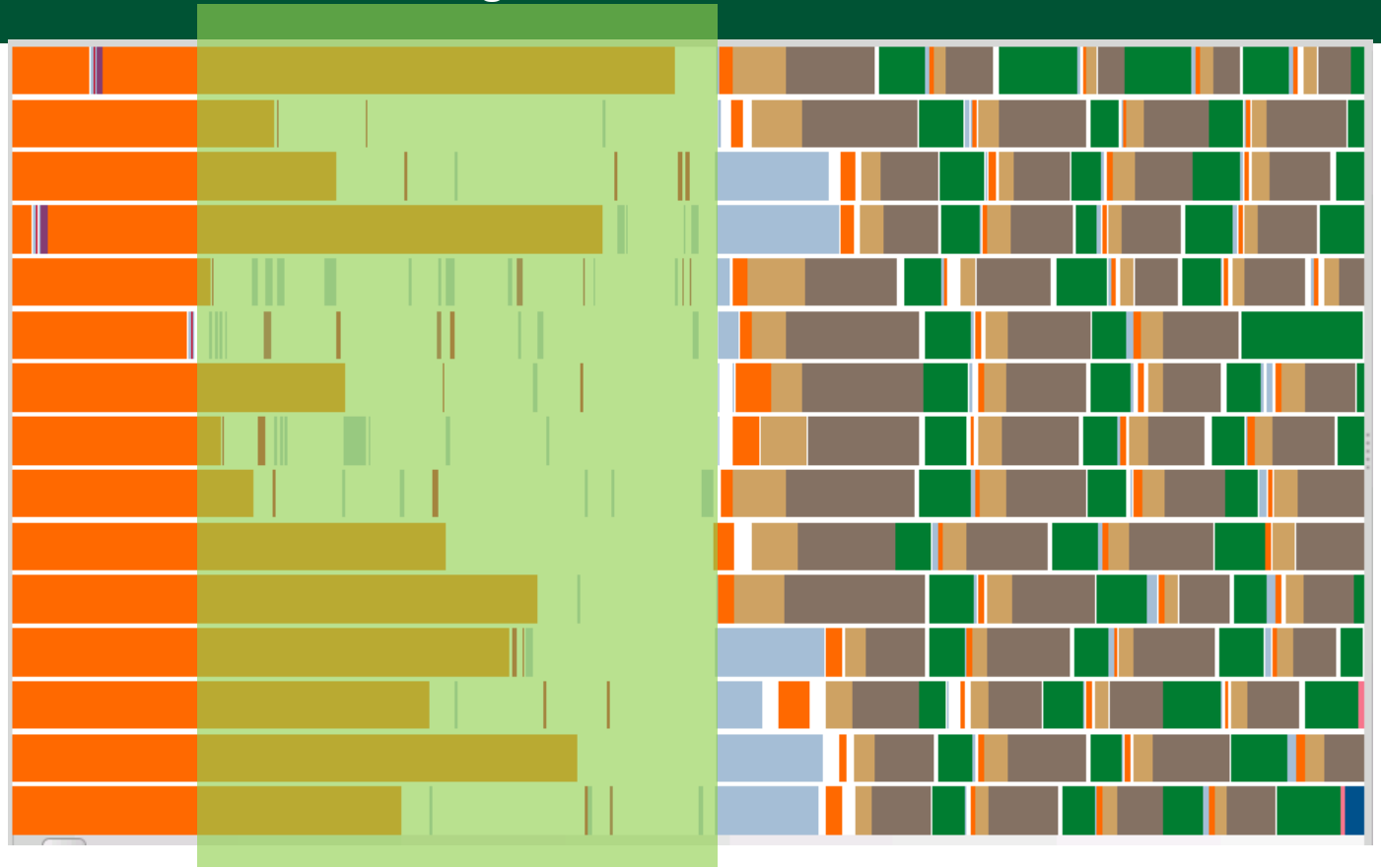
- Idle workers
  - Can we automatically identify why workers are idle?
- HPCG iteration boundaries
  - Center of reduction represents natural bottleneck
  - Reduction cannot complete until the final task of the iteration is complete
  - Next iteration cannot begin until reduction completes
- Generic diagnosis module
  - Find **idle regions**
  - Identify the **bottleneck** task (region-breaking task)
  - Follow dependencies back from effect to cause (task to blame)

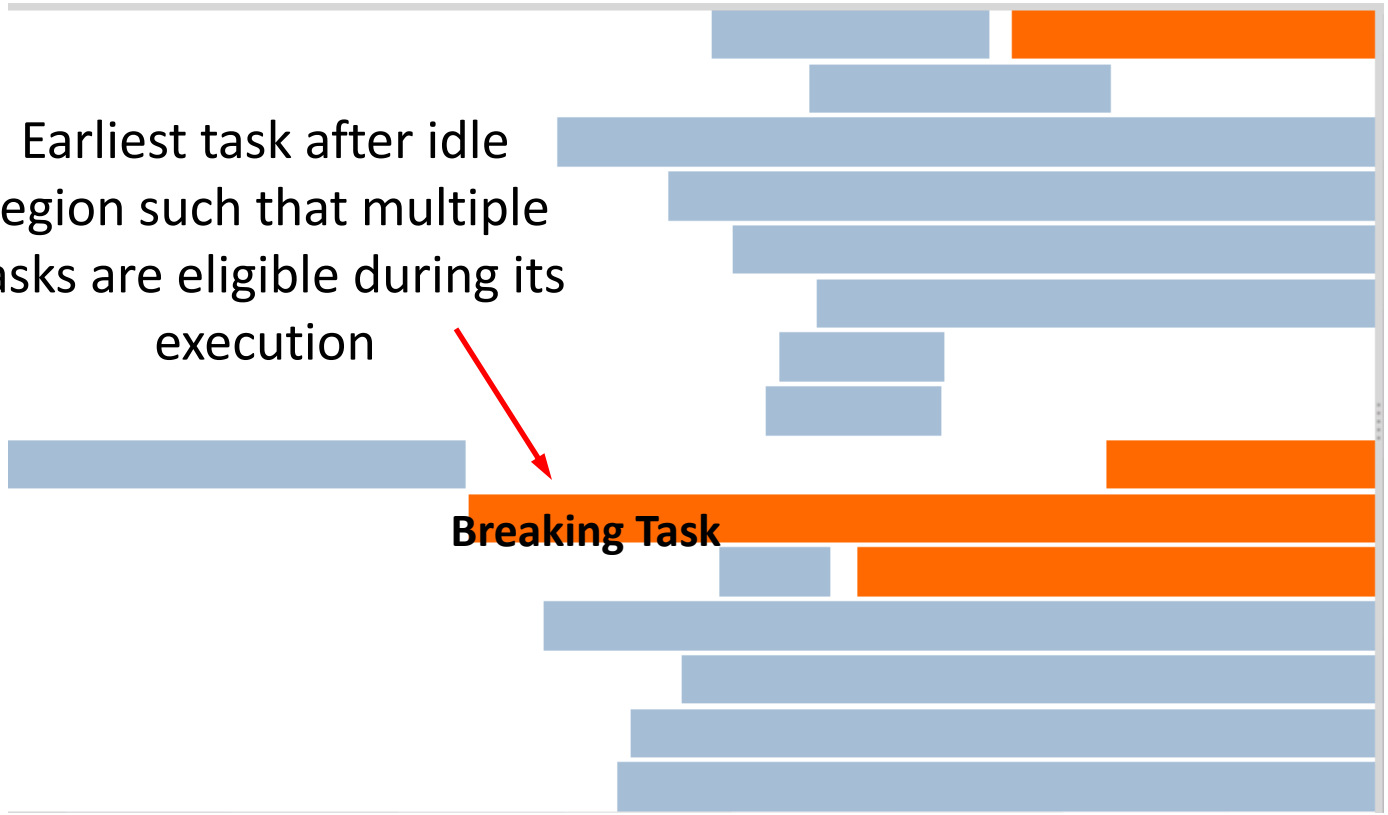

Idle region of interest
Threads of execution
Time

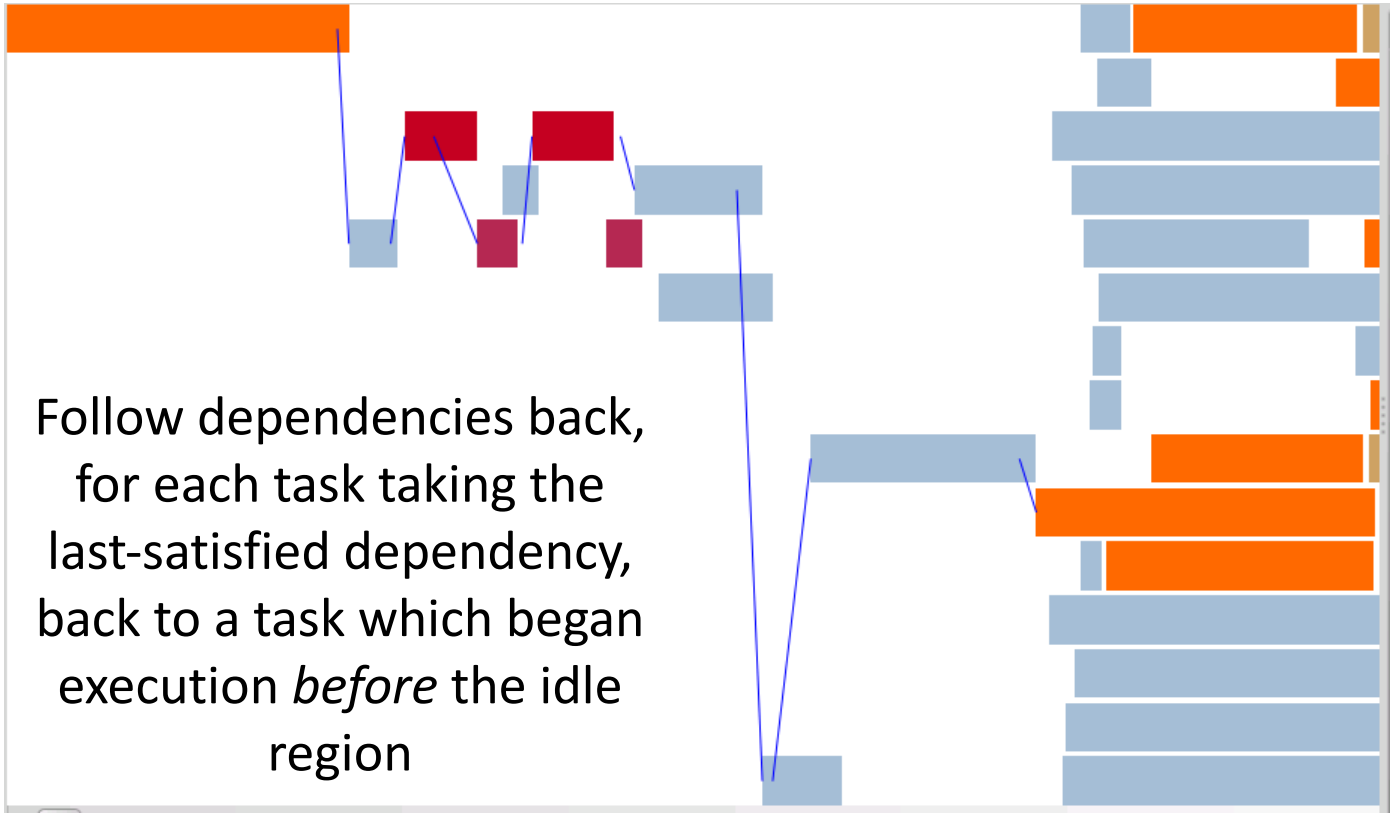


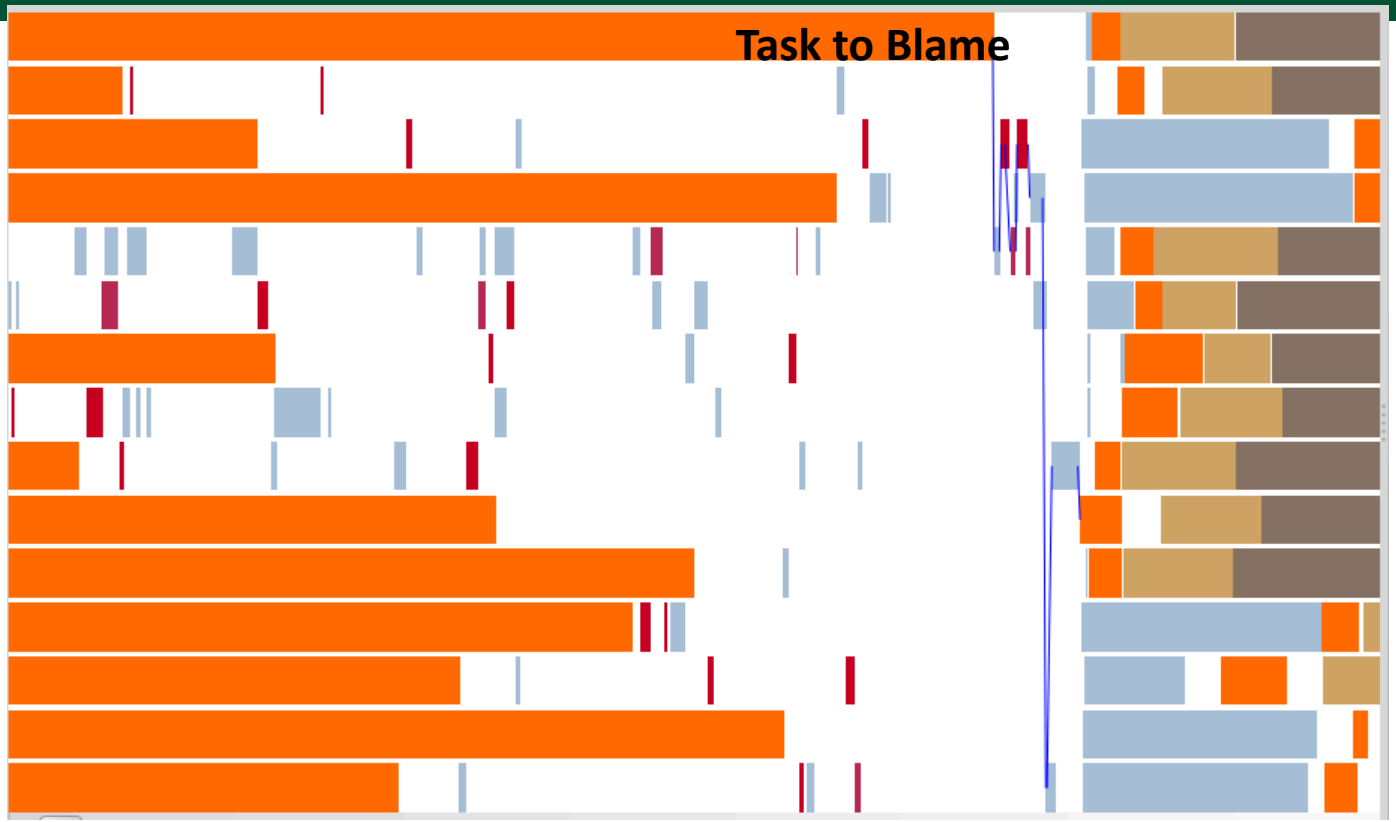

Decreasing Running + Eligible
0 Eligible
Return to Full Occupancy

Idle Region




Earliest task after idle region such that multiple tasks are eligible during its execution
Breaking Task


Follow dependencies back, for each task taking the last-satisfied dependency, back to a task which began execution *before* the idle region

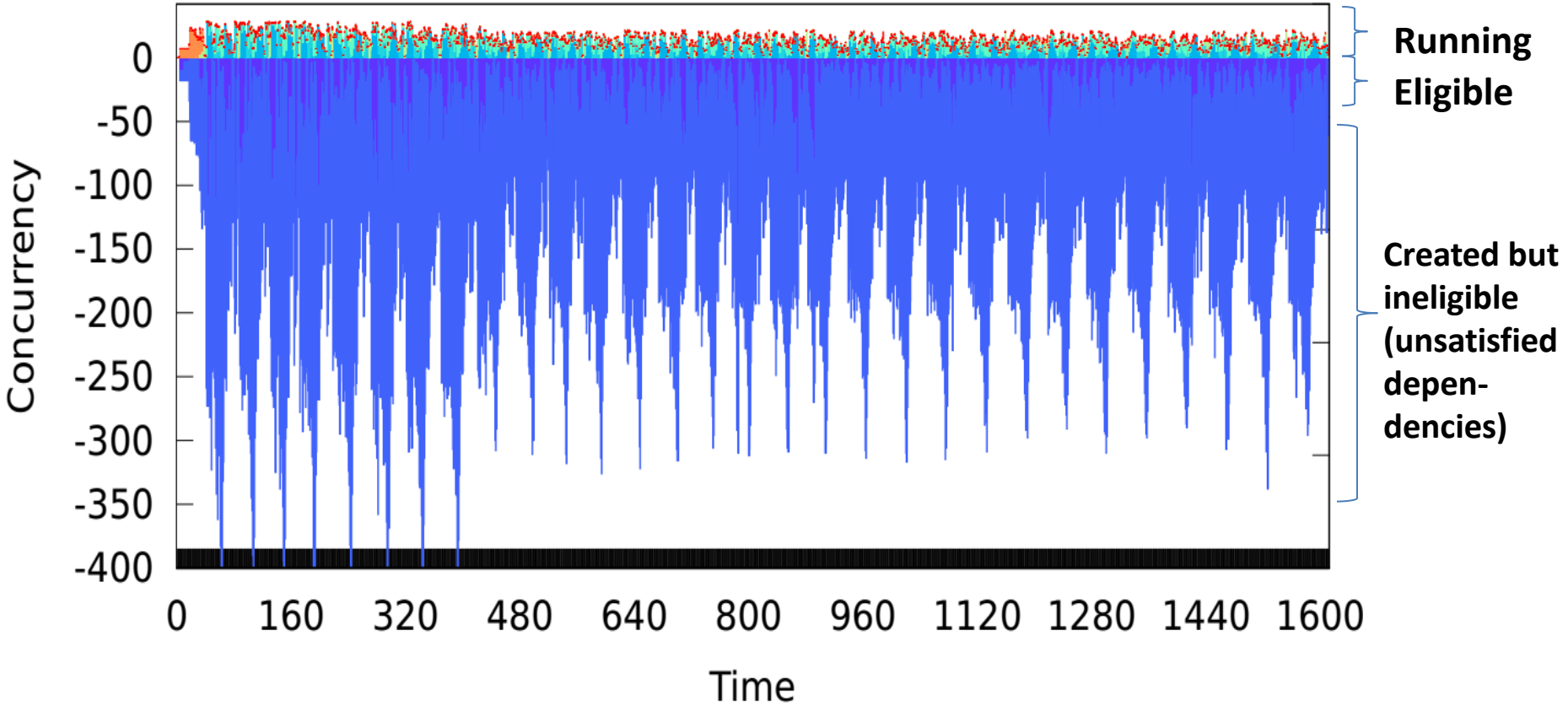


If the final satisfaction taking place during the task to blame had occurred earlier, the idle region *could* have been shorter.

“Eligible” to run (scheduled)

- Add events to HPX scheduler to identify thread states
  - Created
  - Waiting on dependency
  - Scheduled / ready-to-run
  - Executing
  - Completed

APEX concurrency view (modified)



Region of interest