



# Productive Performance Tools for Heterogeneous Parallel Computing

Allen D. Malony  
Department of Computer and Information Science  
University of Oregon

Shigeo Fukuda, *Lunch with a Helmet On*

# Heterogeneous Implications for Performance Tools

- ❑ Tools should support parallel computation models
- ❑ Current status quo is comfortable
  - Mostly homogeneous parallel systems and software
  - Shared-memory multithreading – OpenMP
  - Distributed-memory message passing – MPI
- ❑ Parallel computational models are relatively stable (simple)
  - Corresponding performance models are relatively tractable
  - Parallel performance tools are just keeping up
- ❑ Heterogeneity creates richer computational potential
  - Results in greater performance diversity and complexity
- ❑ Performance tools have to support richer computation models and broader (less constrained) performance perspectives

# Heterogeneous Performance Perspective

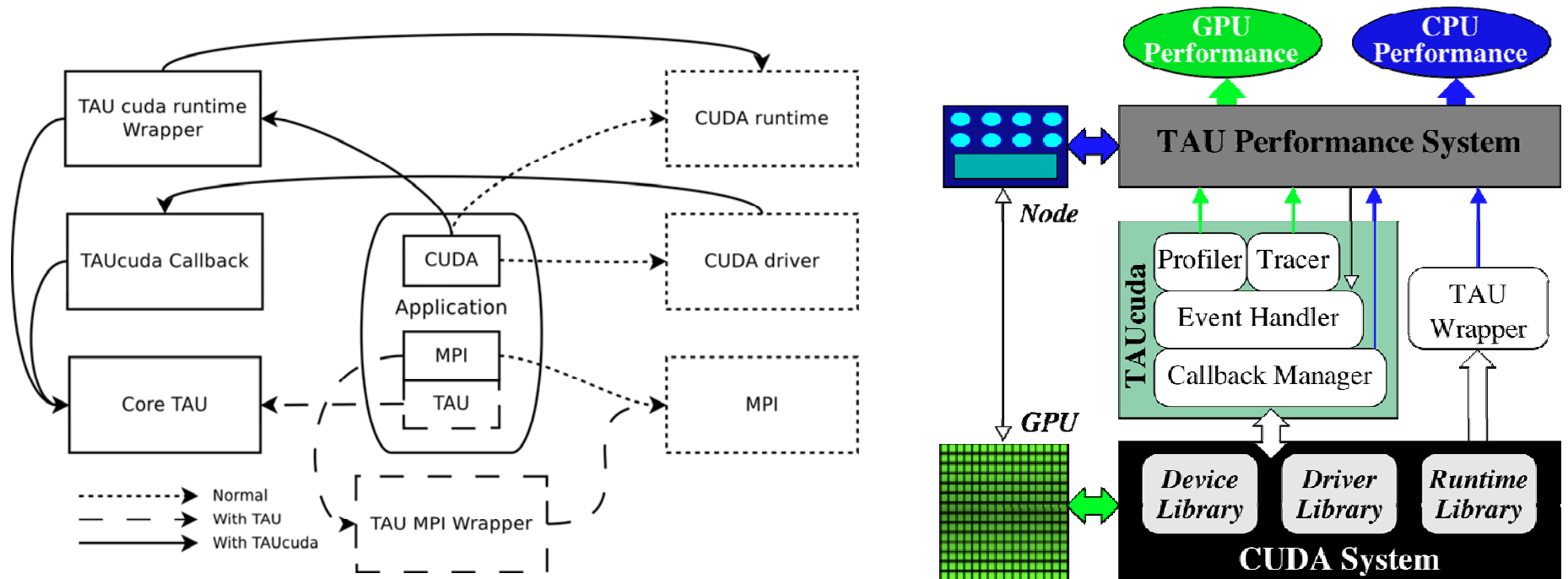
- ❑ Want to create performance views that capture heterogeneous concurrency and execution behavior
  - Reflect execution logic beyond standard actions
  - Capture performance semantics at multiple levels
- ❑ Heterogeneous applications have concurrent execution
  - Want to capture performance for all execution paths
- ❑ Consider “host” path and “external” paths
- ❑ What perspective does the host have of the external entity?
  - Determines the semantics of the measurement data
- ❑ Existing parallel performance tools are CPU(host)-centric
  - Event-based sampling (not appropriate for accelerators)
  - Probe-based measurement

# Heterogeneous Performance Complexity Issues

- ❑ Asynchronous execution (concurrency)
- ❑ Memory transfer and memory sharing
- ❑ Interactions between heterogeneous components
- ❑ Interference between heterogeneous components
- ❑ Different programming languages/libraries/tools
- ❑ Availability of performance data
- ❑ ...

# TAUcuda Performance Measurement

- ❑ CUDA performance measurement
- ❑ Integrated with TAU performance system
- ❑ Built on experimental Linux CUDA driver (R190.86)
  - Captures CUDA device (*cuXXX*) events
  - Captures CUDA runtime (*cudaYYY*) events



# TAUcuda Experimentation Environment

- ❑ University of Oregon
  - Linux workstation
    - Dual quad core Intel Xeon
    - GTX 280
  - GPU cluster (Mist)
    - Four dual quad core Intel Xeon server nodes
    - Two S1070 Tesla servers (4 Tesla GPUs per S1070)
- ❑ Argonne National Laboratory
  - 100 dual quad core NVIDIA Quadro Plex S4
  - 200 Quadro FX5600 (2 per S4)
- ❑ University of Illinois at Urbana-Champaign
  - GPU cluster (AC cluster)
    - 32 nodes with one S1070 (4 GPUs per node)

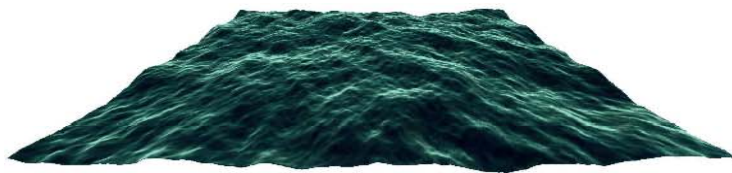


# CUDA SDK OceanFFT

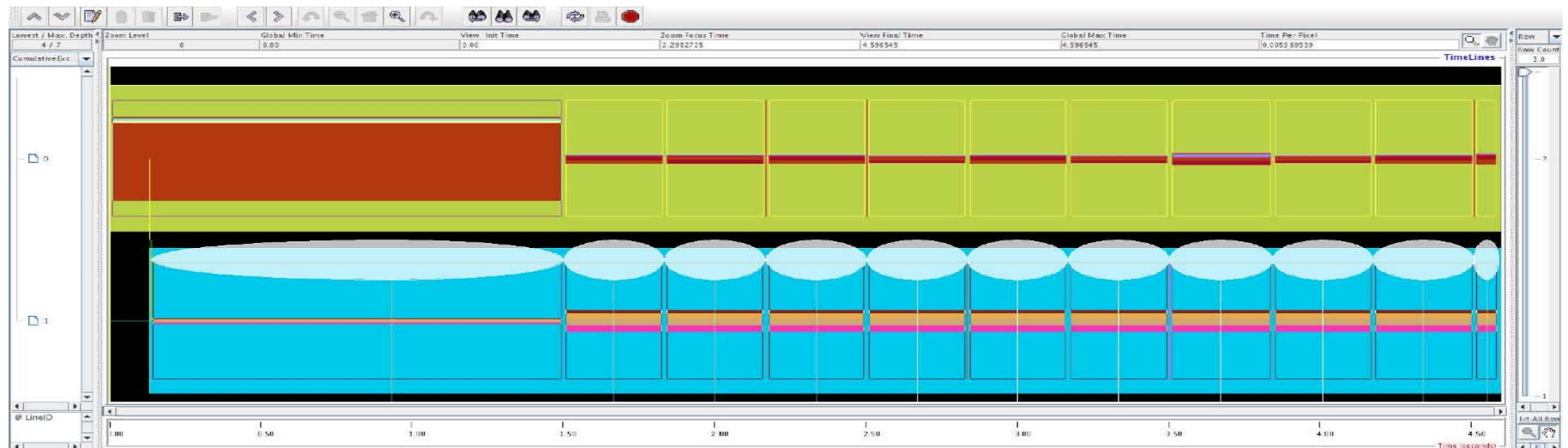
Metric: TIME  
Value: Exclusive  
Units: milliseconds

43503

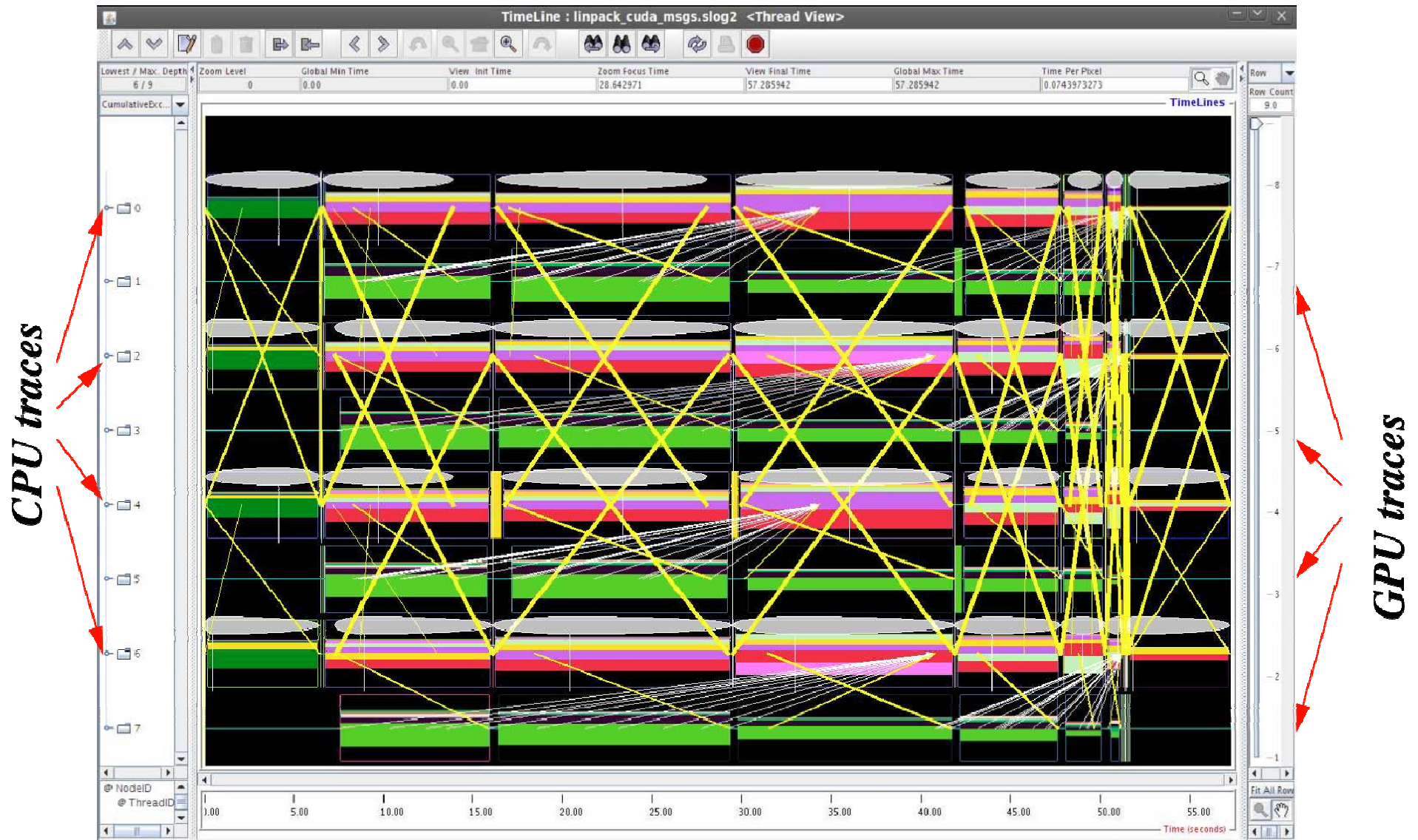
*kernels*



2192.1	.TAU application
2167.8	_Z23SP_c2c_radix4_sp_kernelIffiPvS_i11tfStride_st#cudaLaunch#.TAU application
1767.4	_Z20calculateSlopeKernelPfP6float2jj#cudaLaunch#.TAU application
1749.5	cuLaunchGridAsync
1725.1	_Z23SP_c2r_radix4_sp_kernelIffiPvS_i11tfStride_st#cudaLaunch#.TAU application
1289.4	cudaError_t cudaLaunch(const char *) C
376.484	_Z22generateSpectrumKernelP6float2S0_jjff#cudaLaunch#.TAU application
308.368	cuGLMapBufferObjectAsync
56.166	cuGLUnmapBufferObjectAsync
55.636	cuParamSetSize
54.374	cuFuncSetSharedSize
46.977	cuFuncSetBlockShape
26.414	cuParamSetv
15.729	cudaError_t cudaConfigureCall(dim3, dim3, size_t, cudaStream_t) C
15.271	cuGLCtxCreate
11.654	cudaError_t cudaSetupArgument(const void *, size_t, size_t) C
10.427	cuModuleLoadFatBinary
0.516	cudaError_t cudaGetLastError(void) C
0.504	cudaError_t cudaFree(void *) C
0.251	cudaError_t cudaGetDeviceCount(int *) C
0.143	cuGLRegisterBufferObject
0.081	cuMemAlloc
	cuMemcpyHtoD

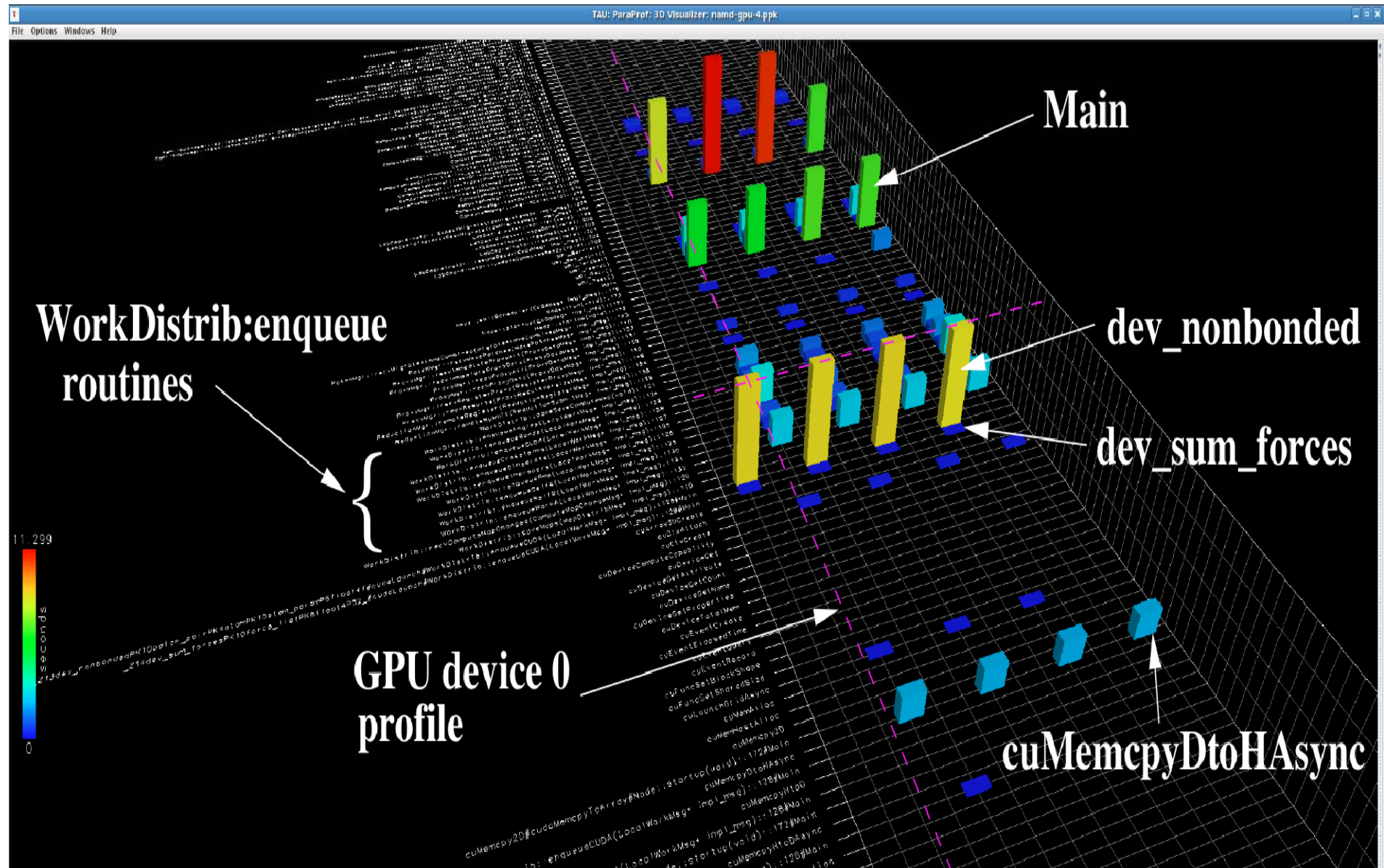


# CUDA Linpack (4 process, trace)





# NAMD Performance Profile



# Call for “Extreme” Performance Engineering

- ❑ Strategy to respond to technology changes and disruptions
- ❑ Strategy to carry forward performance expertise and knowledge
- ❑ Built on robust, integrated performance measurement infrastructure
- ❑ Model-oriented with knowledge-based reasoning
  - Community-driven knowledge engineering
  - Automated data / decision analytics
- ❑ Requires interactions with all SW stack components



# Model Oriented Global Optimization (MOGO)



- Empirical performance data evaluated with respect to performance expectations at various levels of abstraction

