

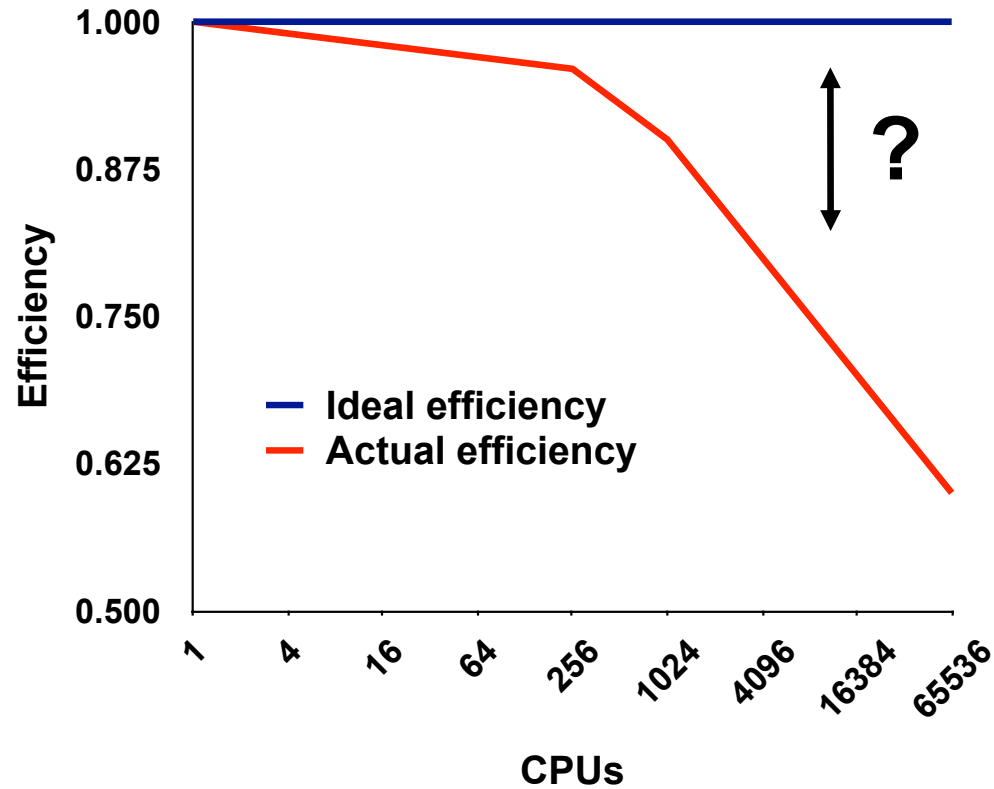
Differential Analysis with HPCToolkit

John Mellor-Crummey
Rice University

2025 Energy HPC Conference
February 28, 2025



The Problem of Scaling



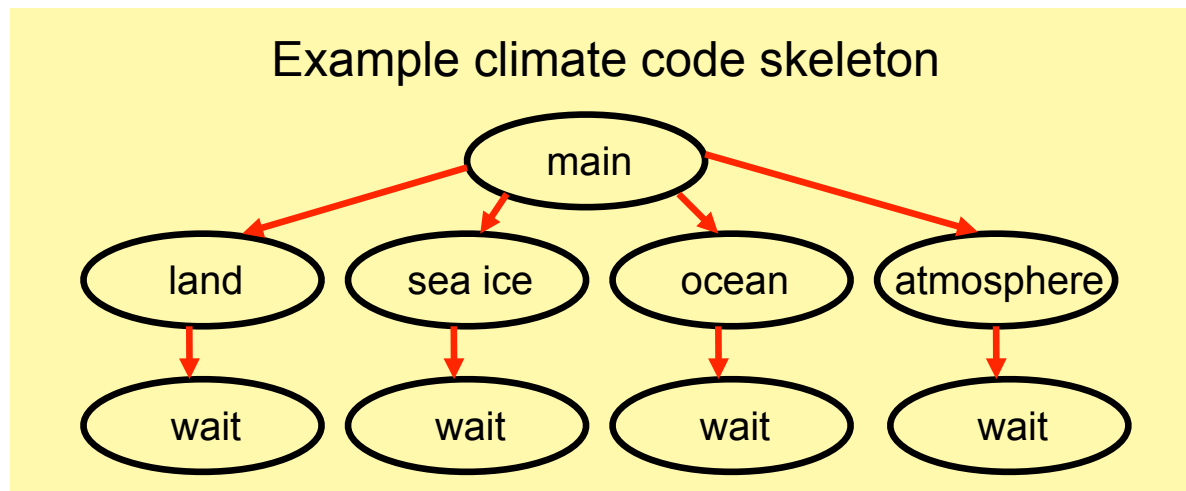
Note: higher is better

Goal: Automatic Scaling Analysis

- Pinpoint scalability bottlenecks
- Guide user to problems
- Quantify the magnitude of each problem
- Diagnose the nature of the problem

Challenges for Pinpointing Scalability Bottlenecks

- **Parallel applications**
 - modern software uses layers of libraries
 - performance is often context dependent

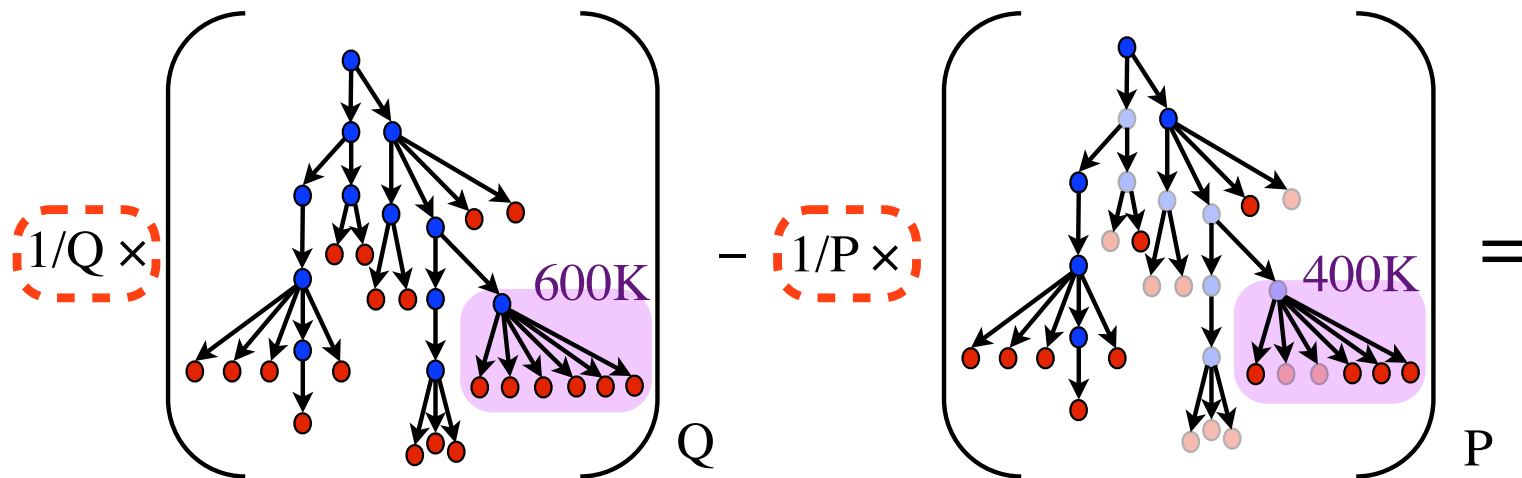


- **Monitoring**
 - bottleneck nature: computation, data movement, synchronization?
 - 2 pragmatic constraints
 - acceptable data volume
 - low perturbation for use in production runs

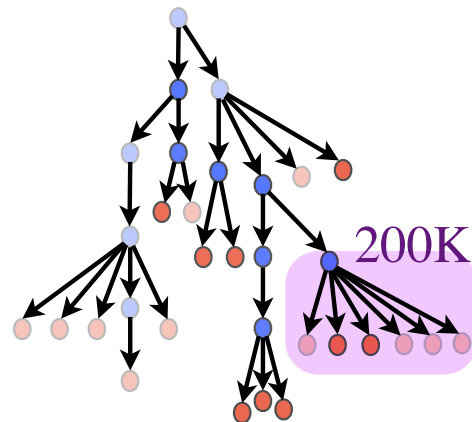
Performance Analysis with Expectations

- You have performance expectations for your parallel code
 - strong scaling: linear speedup
 - weak scaling: constant execution time
- Put your expectations to work
 - measure performance under different conditions
 - e.g., different levels of parallelism or different inputs
 - express your expectations as an equation
 - compute the deviation from expectations for each calling context
 - for both inclusive and exclusive costs
 - correlate the metrics with the source code
 - explore the annotated call tree interactively

Pinpointing and Quantifying Scalability Bottlenecks



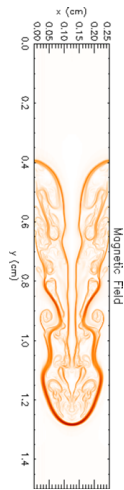
coefficients for analysis
of weak scaling



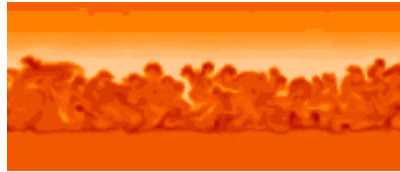
Scalability Analysis Demo

Code:
Simulation:
Platform:
Experiment:
Scaling type:

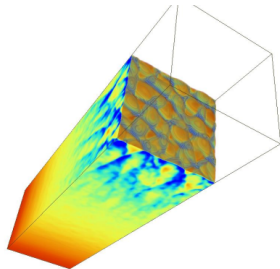
University of Chicago FLASH
white dwarf detonation
Blue Gene/P
8192 vs. 256 processors
weak



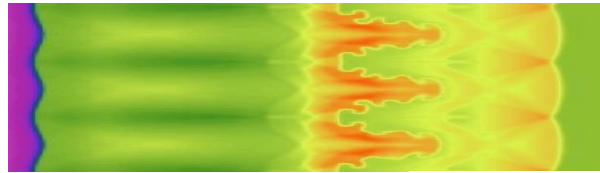
Magnetic Rayleigh-Taylor



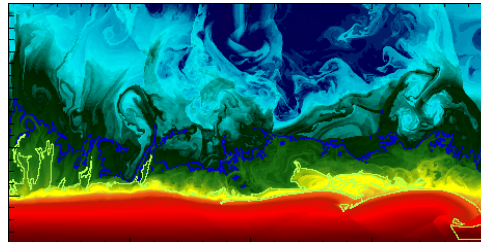
Nova outbursts on white dwarfs



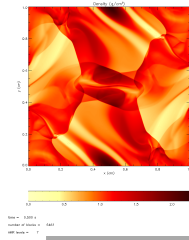
Cellular detonation



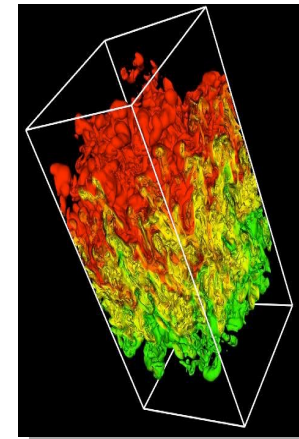
Laser-driven shock instabilities



Helium burning on neutron stars



Orzag/Tang MHD vortex



Rayleigh-Taylor instability

Scalability Analysis of Flash (Demo)

hpcviewer: FLASH/white dwarf: IBM BG/P, weak 256->8192

Driver_initFlash.F90 local_tree_build.F90

```

206 !-----First pass only add lrefine = 1 blocks to tree(s)
207 !-----Second pass add the rest of the blocks.
208     Do ipass = 1,2
209
210         lnblocks_old = lnblocks
211         proc = mype
212 !-----Loop through all processors
213     Do iproc = 0, nprocs-1
214
215         If (iproc == 0) Then
216             off_proc = .False.
217         Else
  
```

Calling Context View Callers View Flat View

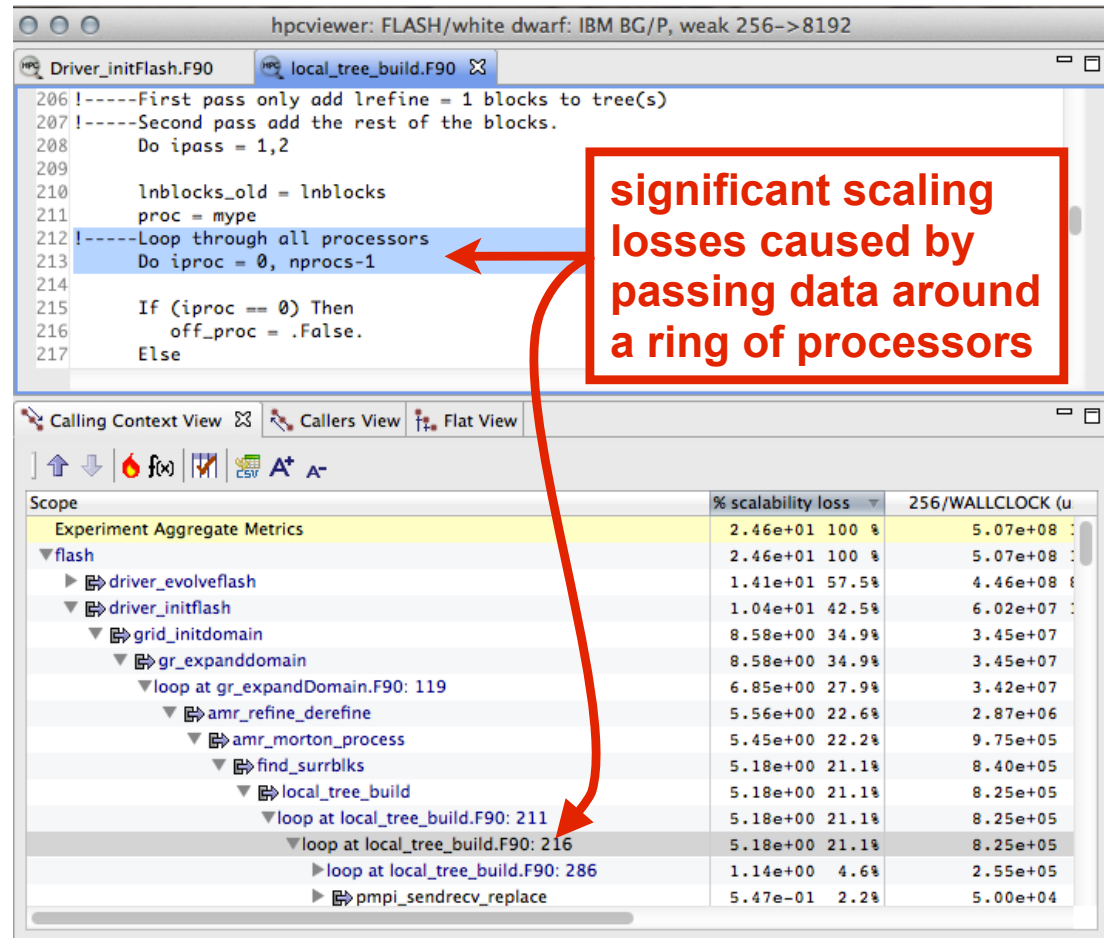
Scope	% scalability loss	256/WALLCLOCK (u)
Experiment Aggregate Metrics	2.46e+01 100 %	5.07e+08
▼ flash	2.46e+01 100 %	5.07e+08
▶ driver_evolveflash	1.41e+01 57.5%	4.46e+08
▼ driver_initflash	1.04e+01 42.5%	6.02e+07
▼ grid_initdomain	8.58e+00 34.9%	3.45e+07
▼ gr_expanddomain	8.58e+00 34.9%	3.45e+07
▼ loop at gr_expandDomain.F90: 119	6.85e+00 27.9%	3.42e+07
▼ amr_refine_derefine	5.56e+00 22.6%	2.87e+06
▼ amr_morton_process	5.45e+00 22.2%	9.75e+05
▼ find_surrblks	5.18e+00 21.1%	8.40e+05
▼ local_tree_build	5.18e+00 21.1%	8.25e+05
▼ loop at local_tree_build.F90: 211	5.18e+00 21.1%	8.25e+05
▼ loop at local_tree_build.F90: 216	5.18e+00 21.1%	8.25e+05
▶ loop at local_tree_build.F90: 286	1.14e+00 4.6%	2.55e+05
▶ pmpi_sendrecv_replace	5.47e-01 2.2%	5.00e+04

Scalability Analysis

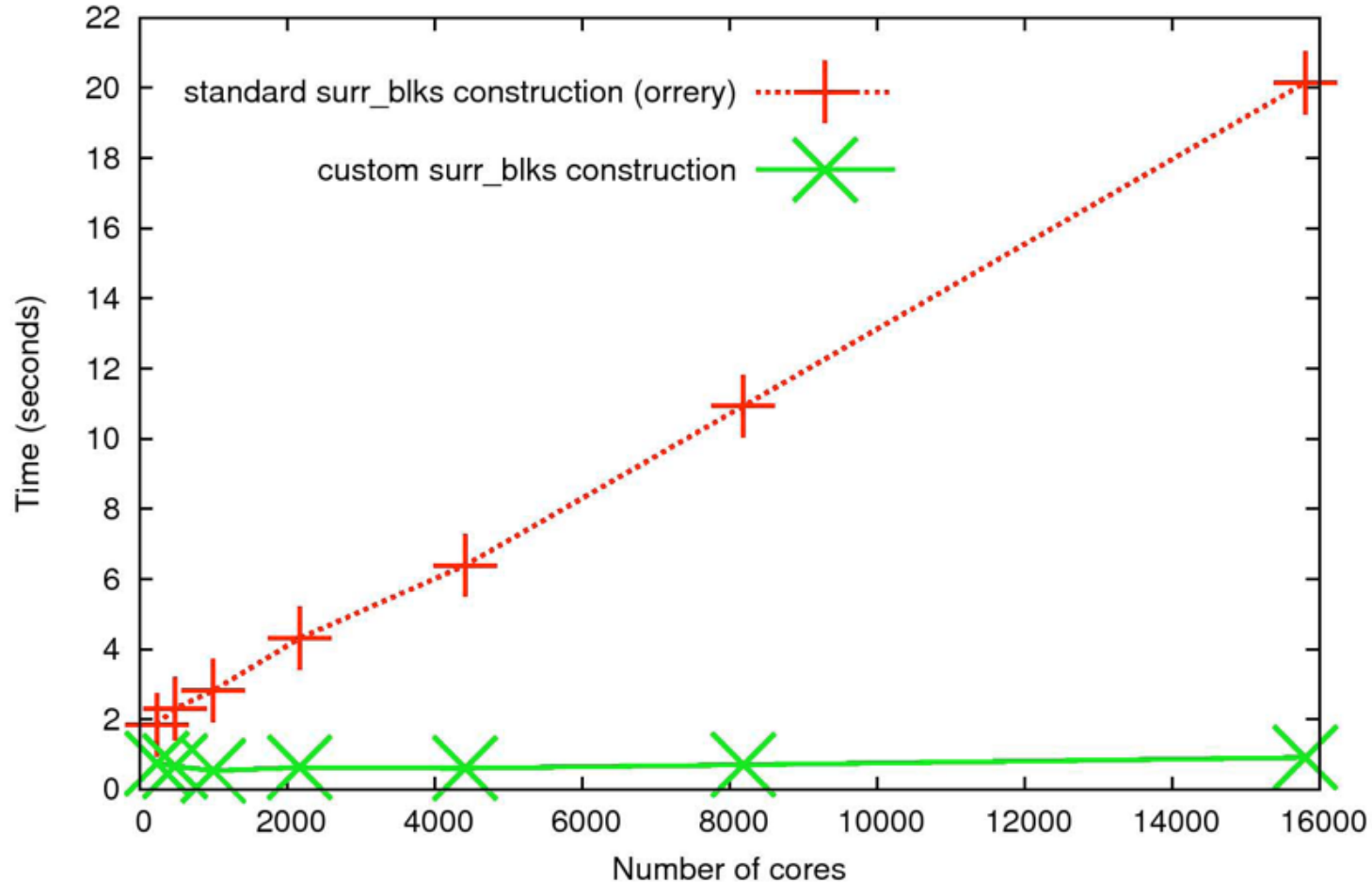
- Difference call path profile from two executions

- different number of nodes
- different number of threads

- Pinpoint and quantify scalability bottlenecks within and across nodes



Improved Flash Scaling of AMR Setup



Graph courtesy of Anshu Dubey, U Chicago

Using Differential Performance Analysis

- The example shown was a hand-crafted database created using a single MPI rank from each of two executions at different scales
- You can do strong or weak scaling analysis on your own by
 - providing two measurement directories to `hpcprof/hpcprof-mpi`
 - writing an equation to compute the scaling loss from one to the other
 - or a worker thread did the work