BERKELEY LAB
LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

Parallel I/O Performance: From Events to Ensembles

In collaboration with:

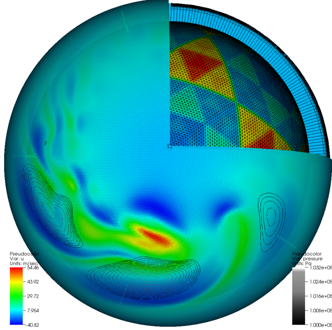
- Lenny Oliker
- David Skinner
- Mark Howison
- Nick Wright
- Noel Keen
- John Shalf
- Karen Karavanic

Andrew Uselton

National Energy Research Scientific Computing Center

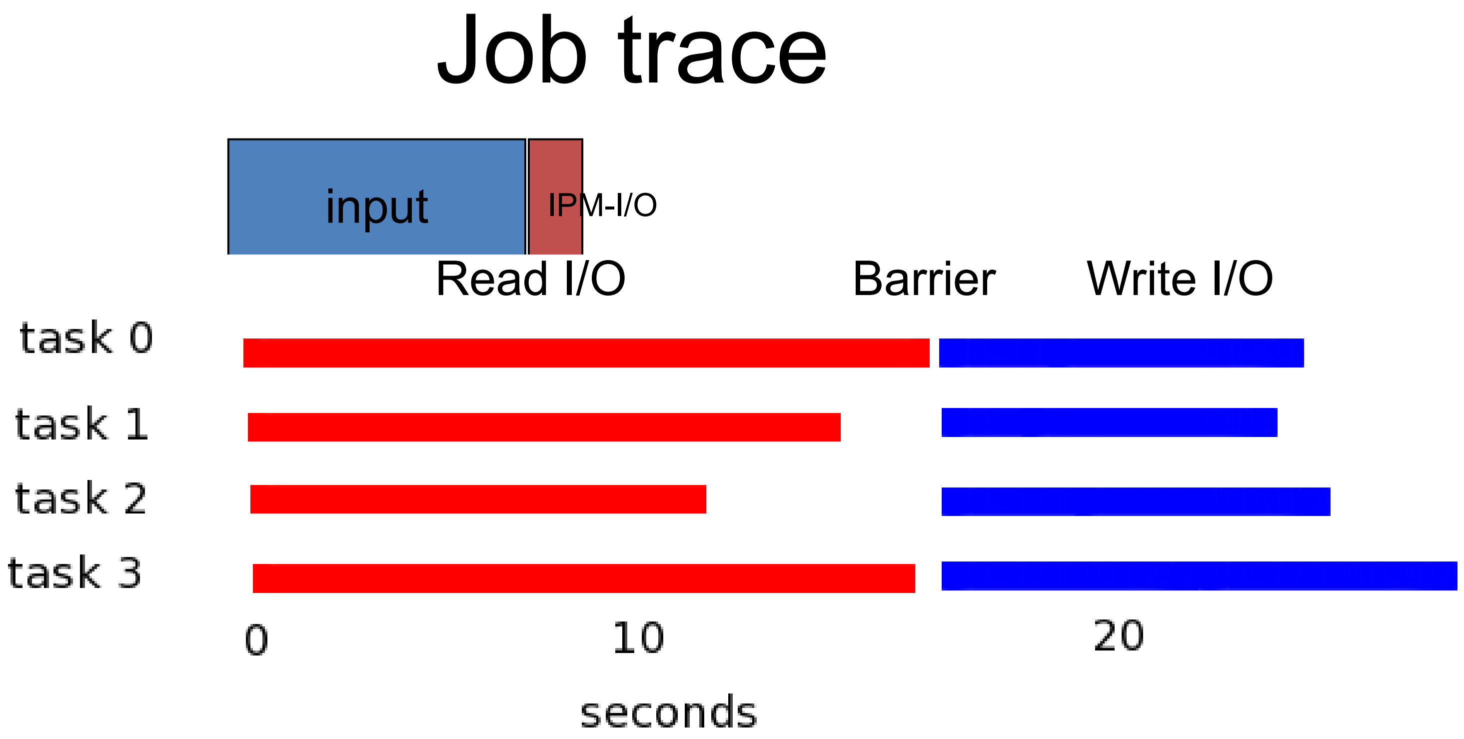
Lawrence Berkeley National Laboratory

Parallel I/O Evaluation and Analysis



- Explosion of sensor & simulation data make I/O a critical component
- Petascale I/O requires new techniques: analysis, visualization, diagnosis
- Statistical methods can be revealing
- Present case studies and optimization results for:
  - **MADbench – A cosmology application**
  - **GCRM – A climate simulation**

IPM-I/O is an interposition library that wraps I/O calls with tracing instructions



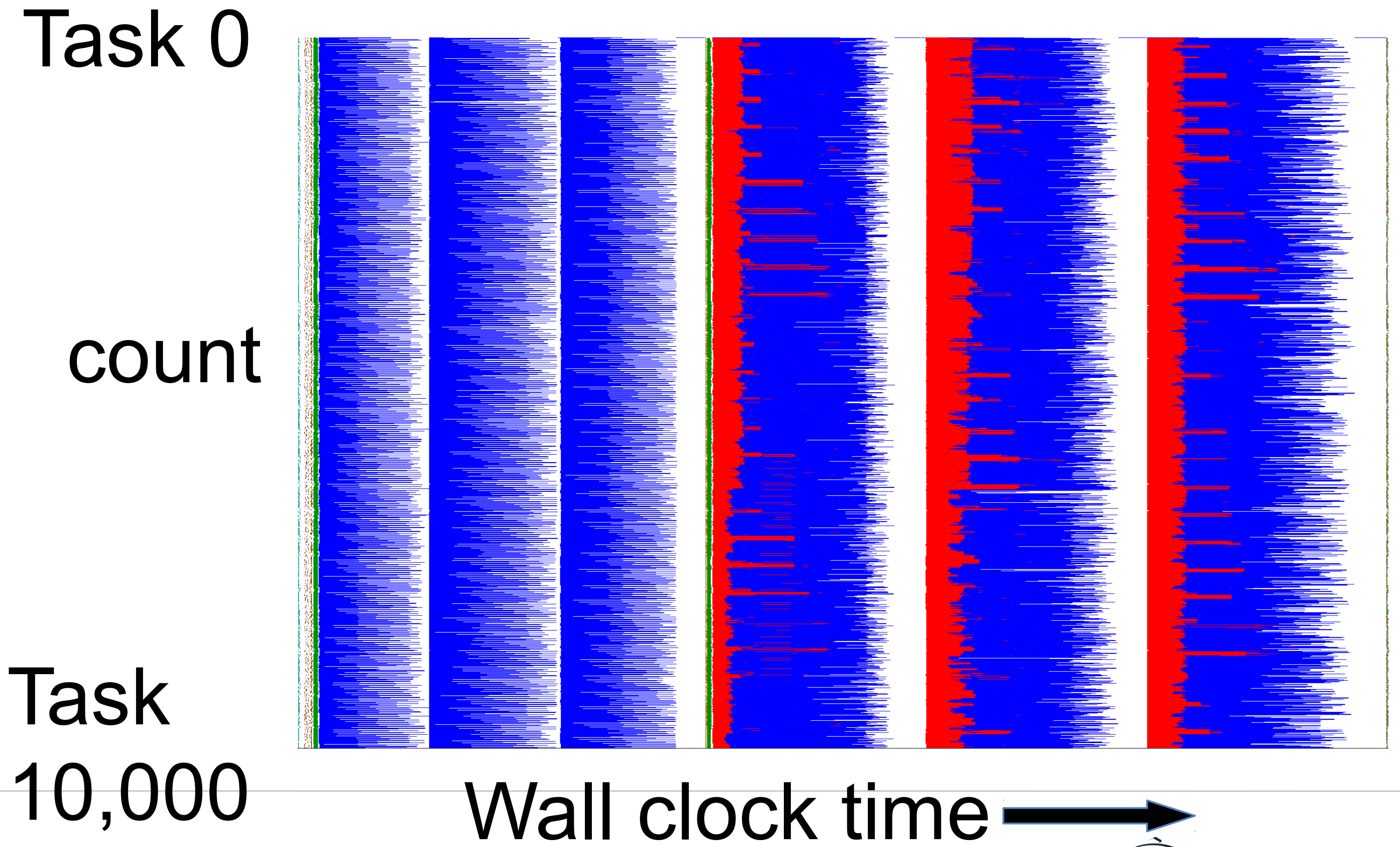
Events to Ensembles

The details of a trace can obscure as much as they reveal

And it does not scale

Statistical methods reveal what the trace obscures

And it does scale



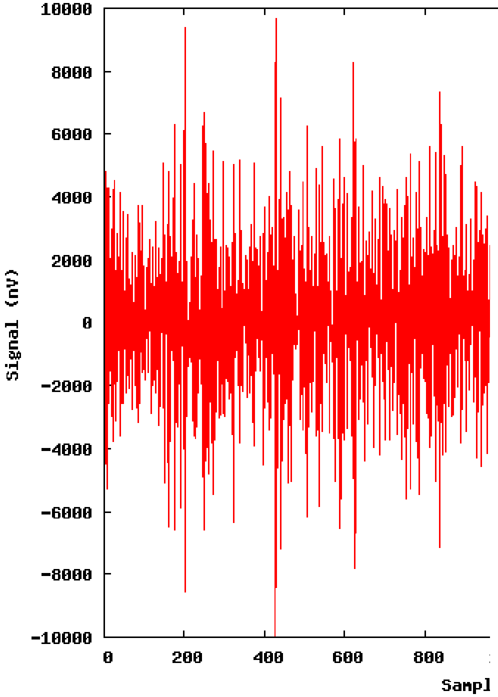
Case Study #1:

MADCAP analyzes the Cosmic Microwave Background radiation.

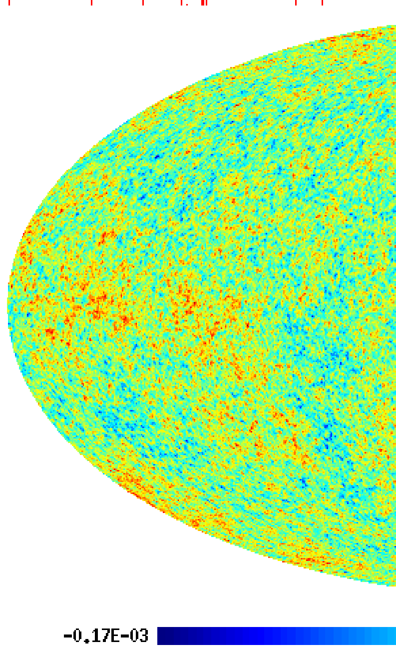
Madbench – An out-of-core matrix solver writes and reads all of memory multiple times.

CMB Data Analysis

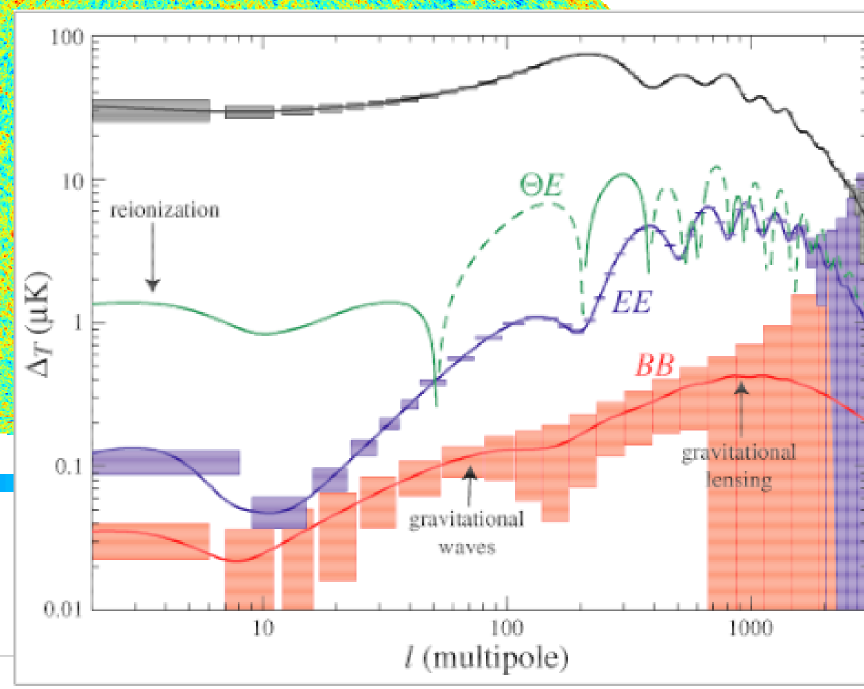
time domain - *O*(1012)



pixel sky map - *O*(108)



angular power spectrum - *O*(104)



MADbench Overview

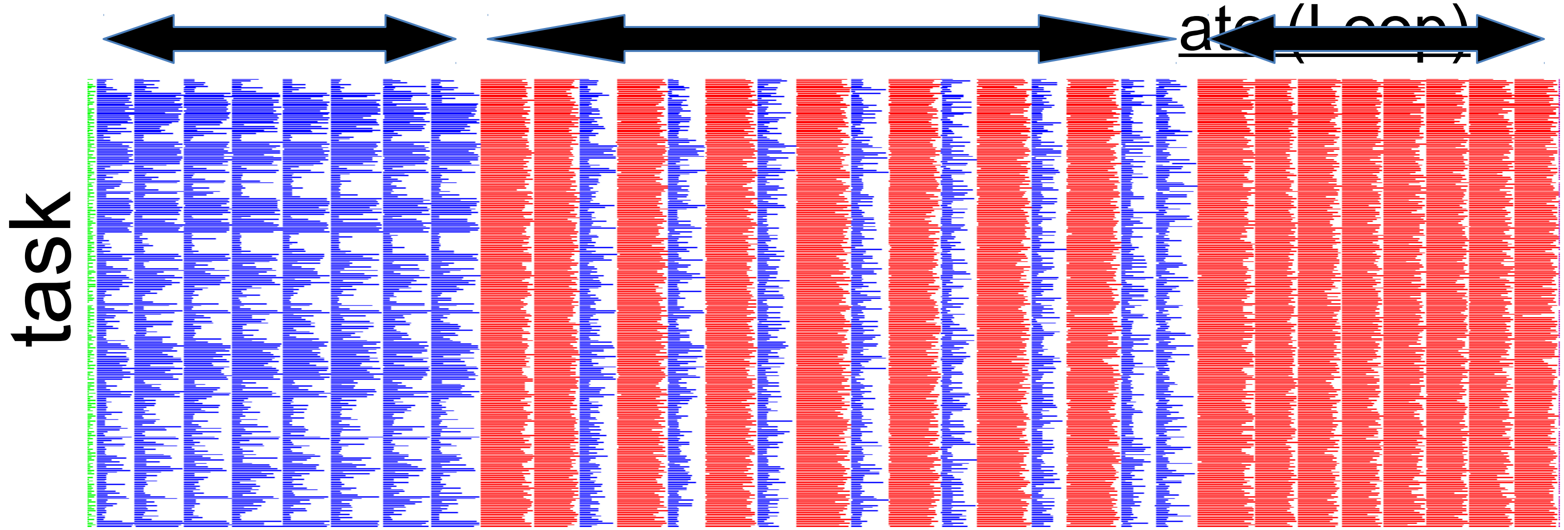
- MADCAP is the maximum likelihood CMB angular power spectrum estimation code
- MADbench is a lightweight version of MADCAP
- Out-of-core calculation due to large size and number of pix-pix matrices

Computational Structure

I. Compute, Write (Loop)

III. Read, Compute, Write (Loop)

IV. Read, Compute/Communicate (Loop)

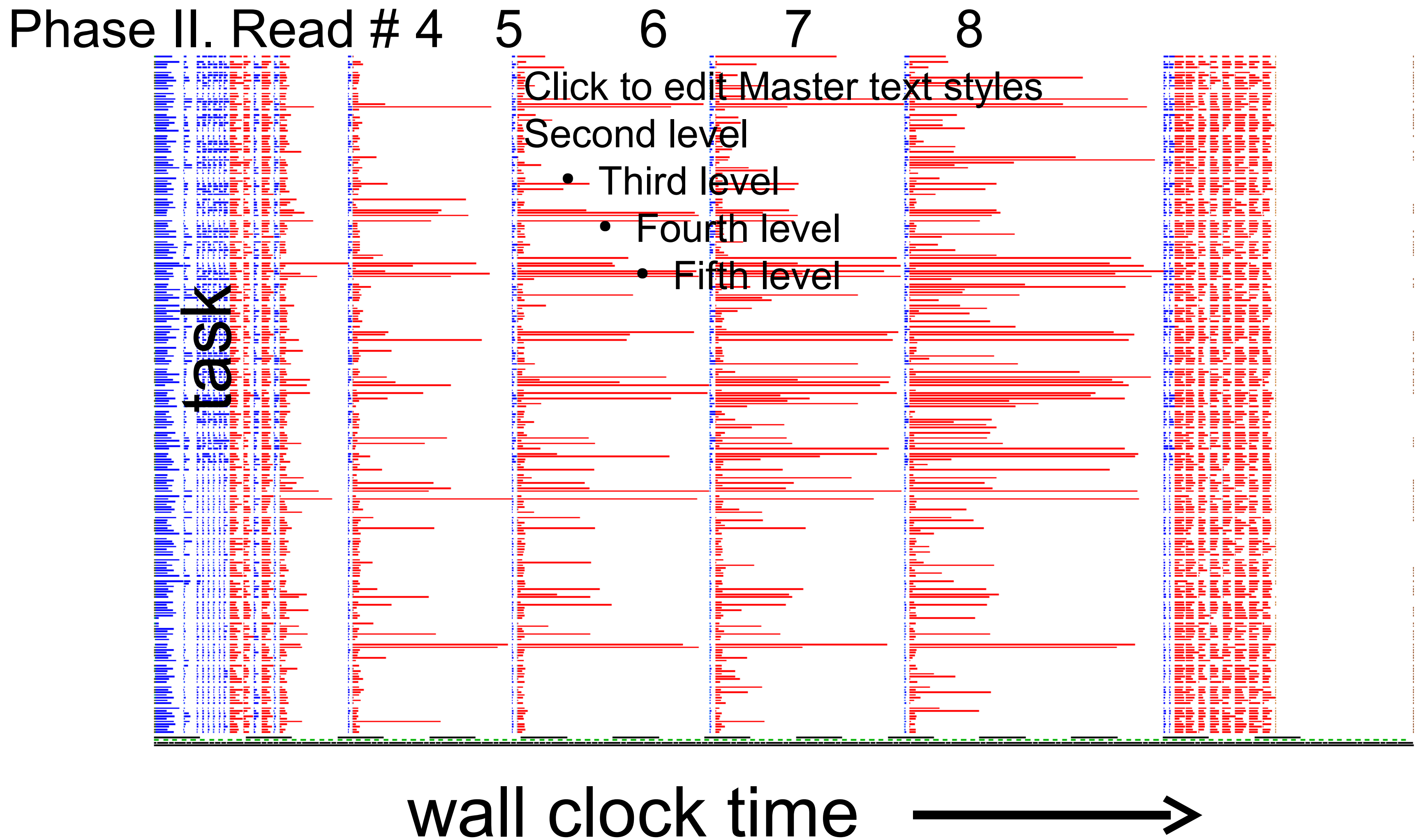


wall clock time

II. Compute/Communicate (no I/O)

The compute intensity can be tuned down to emphasize I/O

MADbench I/O Optimization



MADbench I/O Optimization

count

duration (seconds)

MADbench I/O Optimization

Cumulative Probability

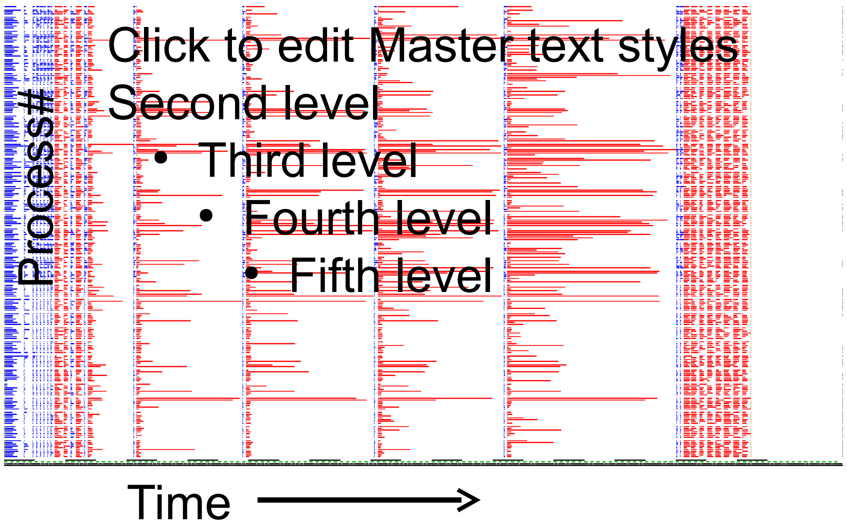
A statistical approach revealed a systematic pattern

duration (seconds)

MADbench I/O Optimization

Click to edit Master text styles
Second level
- Third level
  - Fourth level
    - Fifth level

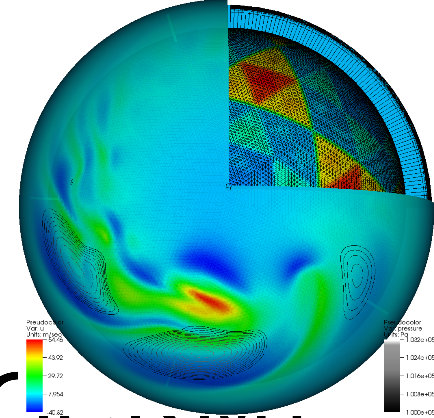
Before



After

Lustre patch eliminated slow reads

Case Study #2:

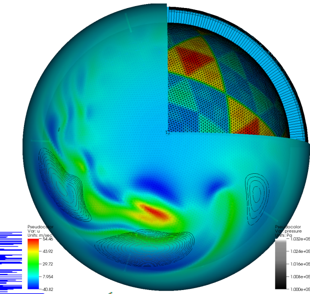


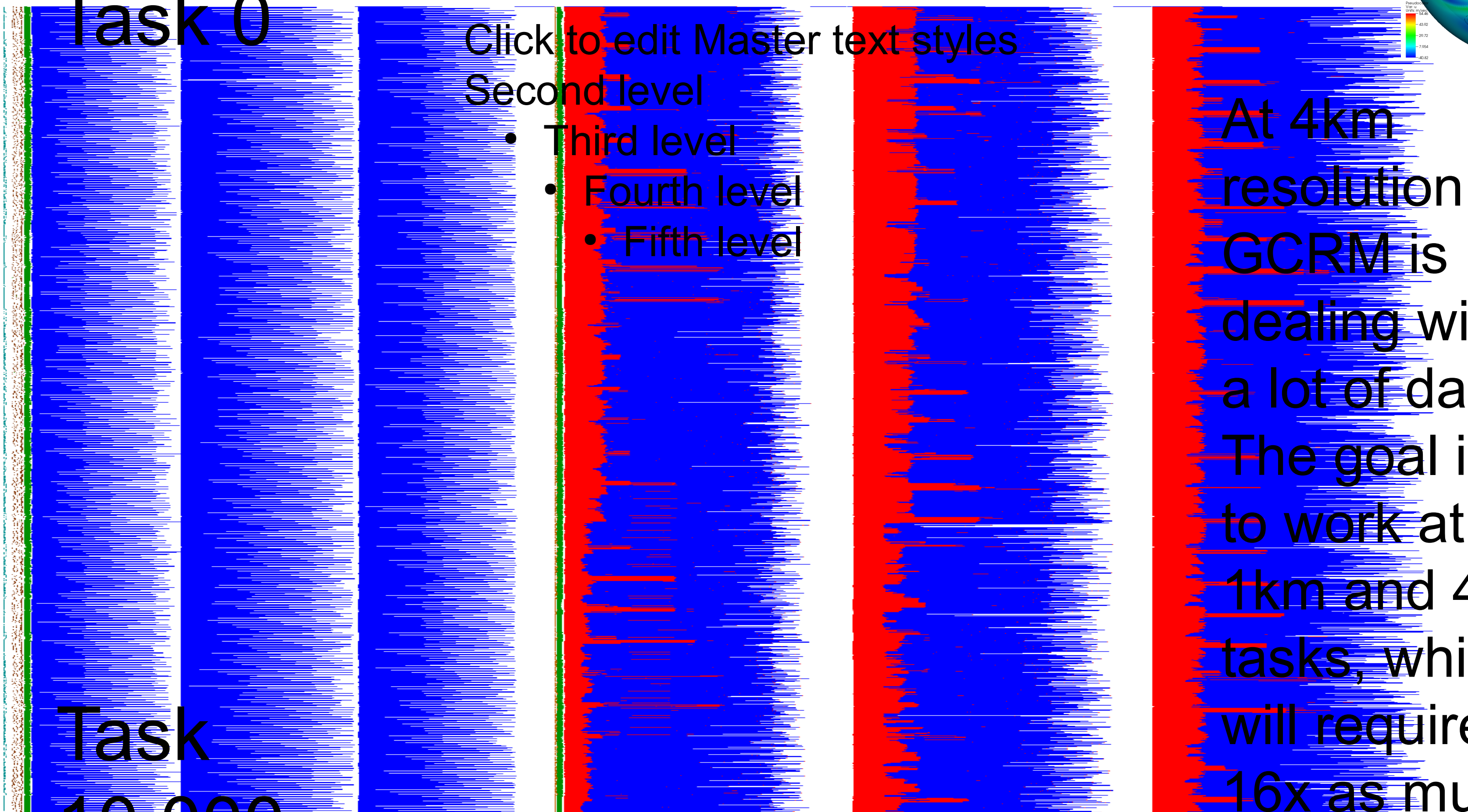
Global Cloud Resolving Model (GCRM) developed by scientists at CSU

Runs resolutions fine enough to simulate cloud formulation and dynamics

Mark Howison’s analysis fixed it

GCRM I/O Optimization





Click to edit Master text styles

Second level

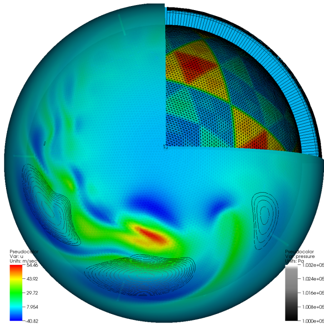
- Third level
  - Fourth level
    - Fifth level

Wall clock time

desired checkpoint time

At 4km resolution GCRM is dealing with a lot of data. The goal is to work at 1km and 40k tasks, which will require 16x as much data.

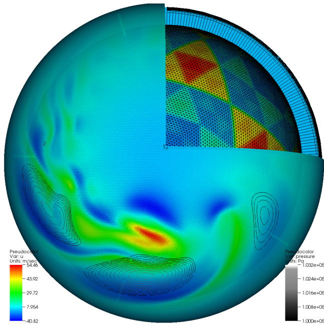
GCRM I/O Optimization



Worst case
20 sec

Insight: all 10,000 are happening at once

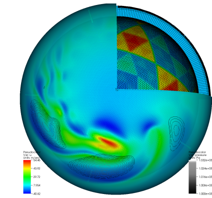
GCRM I/O Optimization



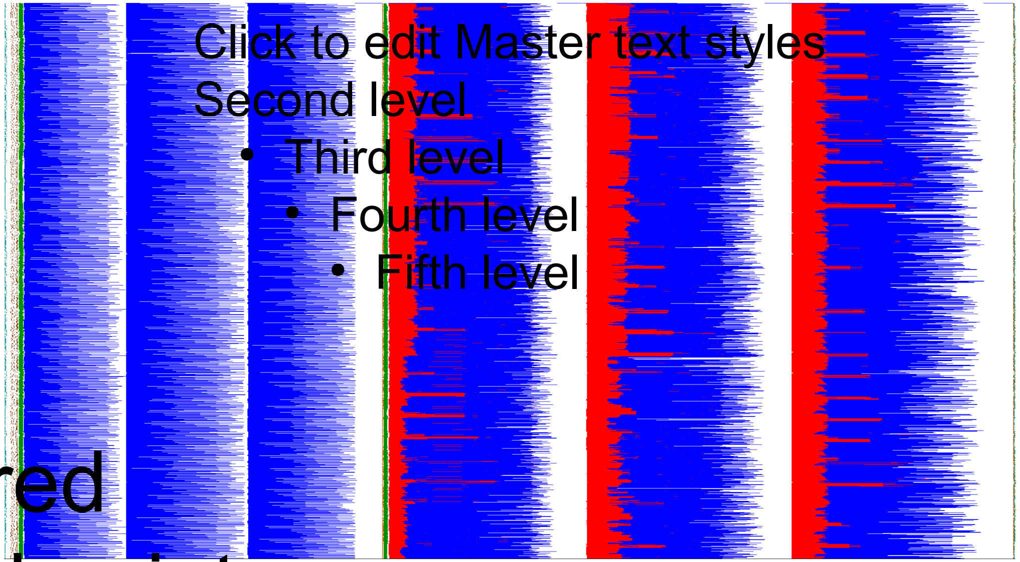
Worst case
3 sec

Collective
buffering
reduces
concurrency

GCRM I/O Optimization



Before



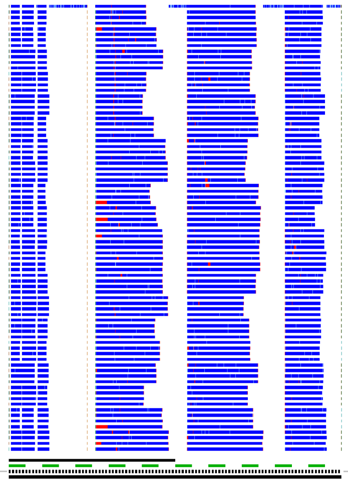
Click to edit Master text styles

Second level

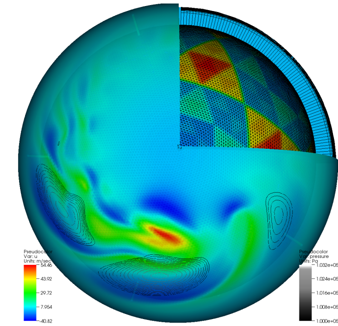
- Third level
  - Fourth level
    - Fifth level

desired checkpoint time

After



GCRM I/O Optimization

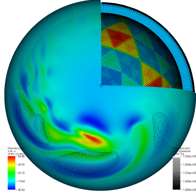


Insight:
Aligned
I/O

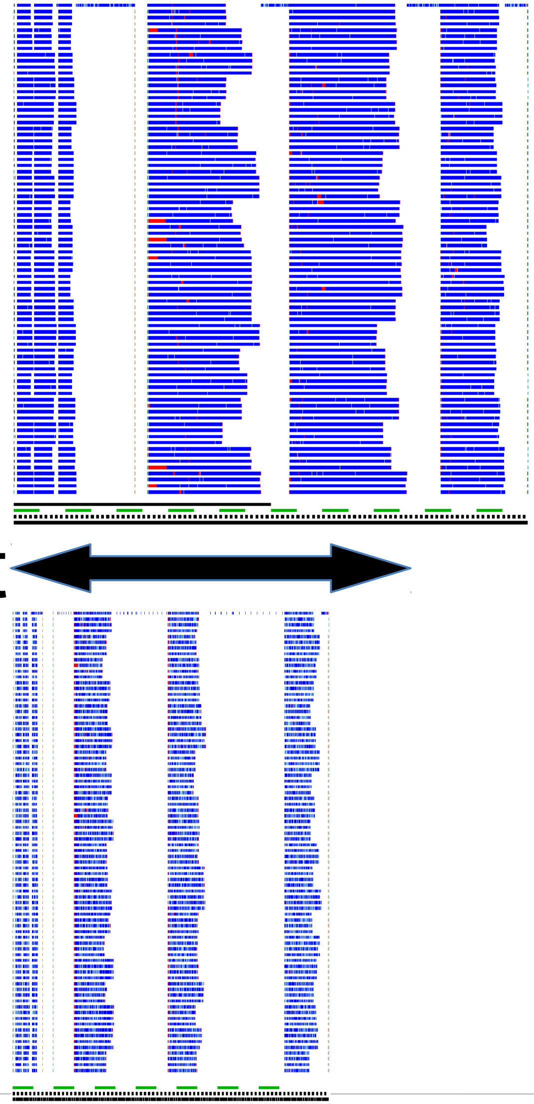
Still need
better
worst case
behavior

Worst case
1 sec

GCRM I/O Optimization



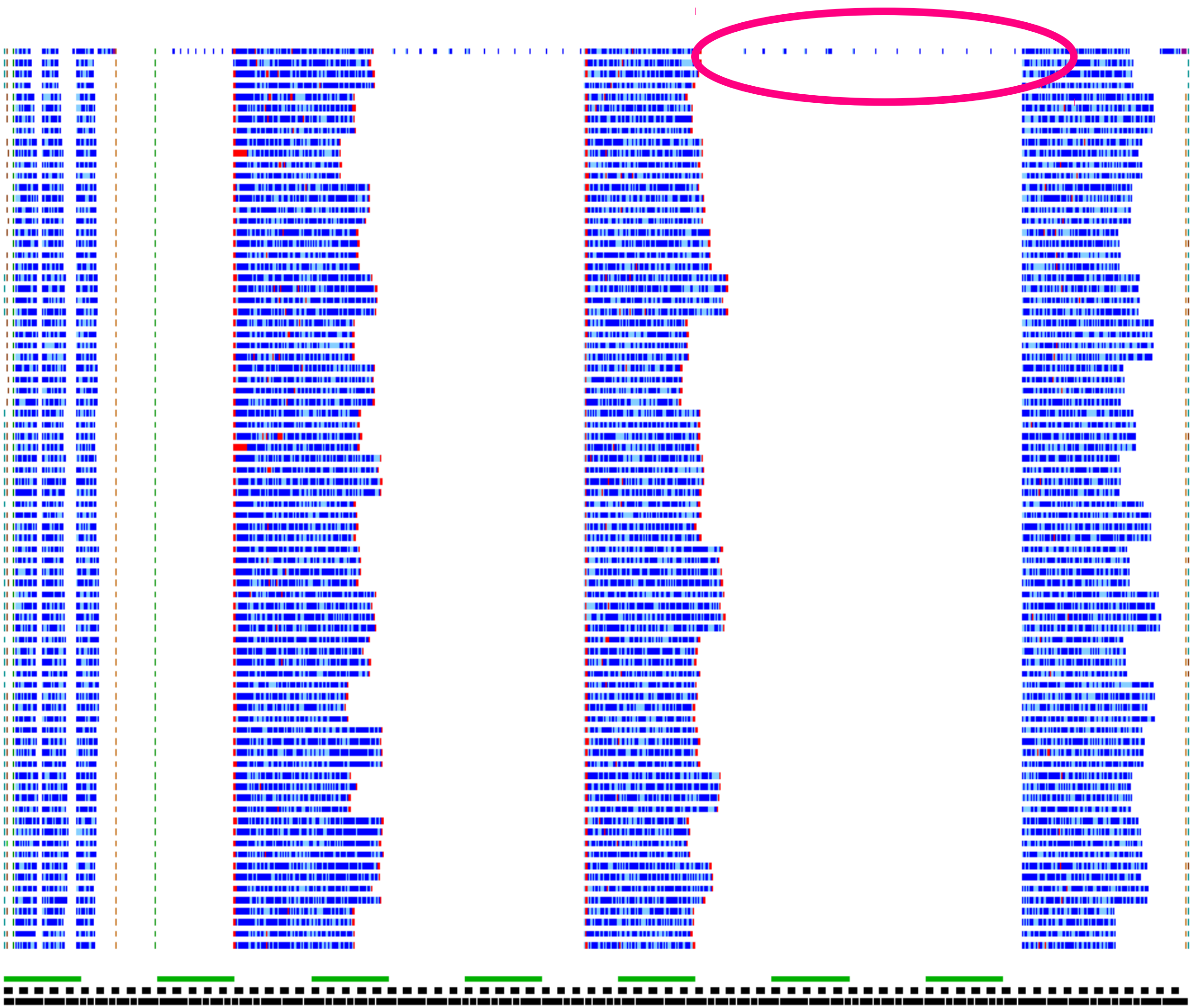
Before

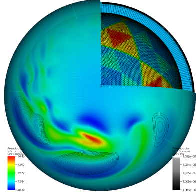


desired checkpoint time

After

GCRM I/O Optimization

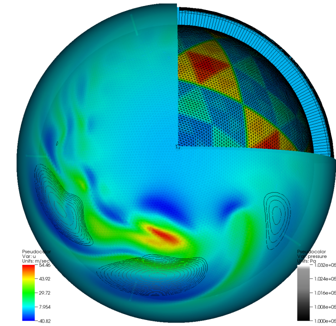




Sometimes the trace view is the right way to look at it

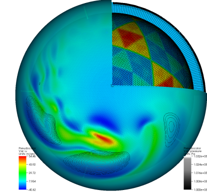
Metadata is being serialized through task 0

GCRM I/O Optimization



Defer metadata ops so there are fewer and they are larger

GCRM I/O Optimization

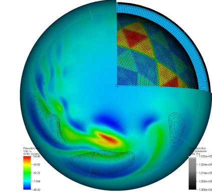


Before

desired checkpoint time

After

Conclusions and Future Work



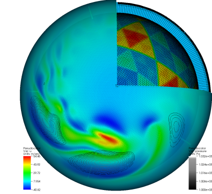
Traces do not scale, can obscure underlying features

Statistical methods scale, give useful diagnostic insights into large datasets

Future work: gather statistical info directly in IPM

Future work: Automatic recognition of model and moments within IPM

Acknowledgements



- Julian Borrill wrote MADCAP/MADbench
- Mark Howison performed the GCRM optimizations
- Noel Keen wrote the I/O extensions for IPM
- Kitrick Sheets (Cray) and Tom Wang (SUN/Oracle) assisted with the diagnosis of the Lustre bug

- This work was funded in part by the DOE Office of Advanced Scientific Computing Research (ASCR) under contract number DE-C02-05CH11231

BERKELEY LAB
LAWRENCE BERKELEY NATIONAL LABORATORY