

Profiling NERSC Workloads on the iDataPlex

**David Skinner
Lawrence Berkeley Lab**



**Summer Clouds
June 17, 2011**



in this talk...

- **NERSC, Magellan Overview**
- **Integrated Performance Monitoring**
- **Magellan Workload**



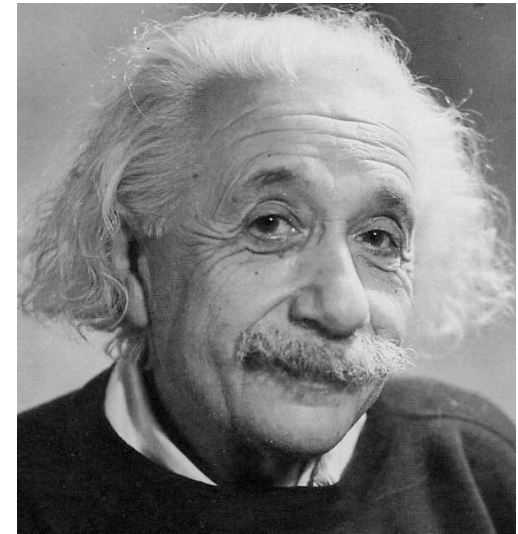
a quick preface on HPC performance...



Performance is Relative

- **To your goals**
 - Time to solution, $T_{\text{queue}} + T_{\text{run}}$
 - Efficient use of allocation
 - Do FLOPs even matter?

- **To the**
 - application code
 - input deck
 - machine type/state



In general the first bottleneck wins.



One Slide about NERSC

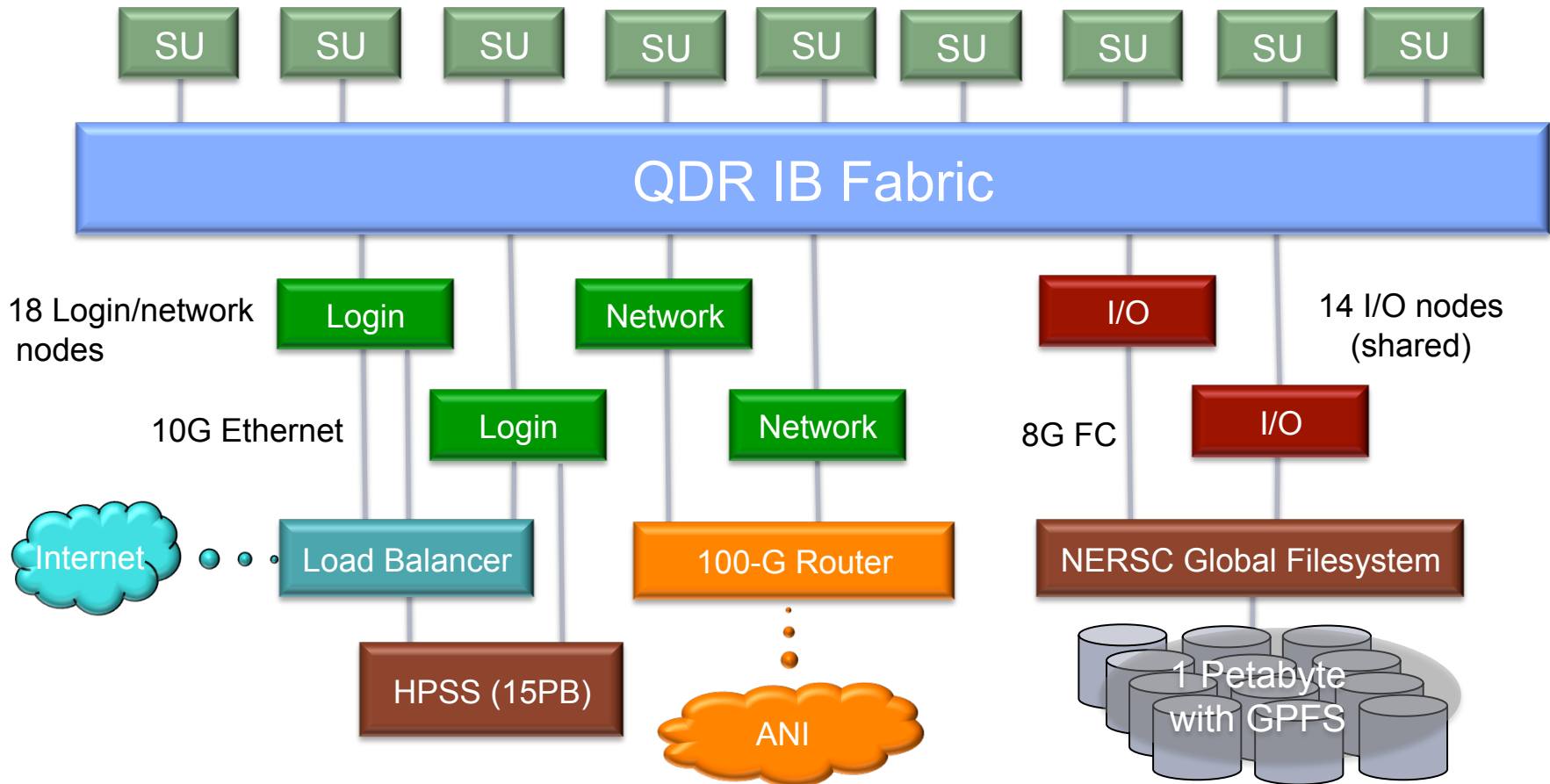
- **Serving all of DOE Office of Science**
 - domain breadth
 - range of scales
- **Science driven**
 - sustained performance
- **Lots of users**
 - ~4K active
 - ~500 logged in
 - ~300 projects
- **Architecture aware**
 - procurements driven by workload needs



Magellan Test Bed at NERSC

Purpose-built for Science Applications

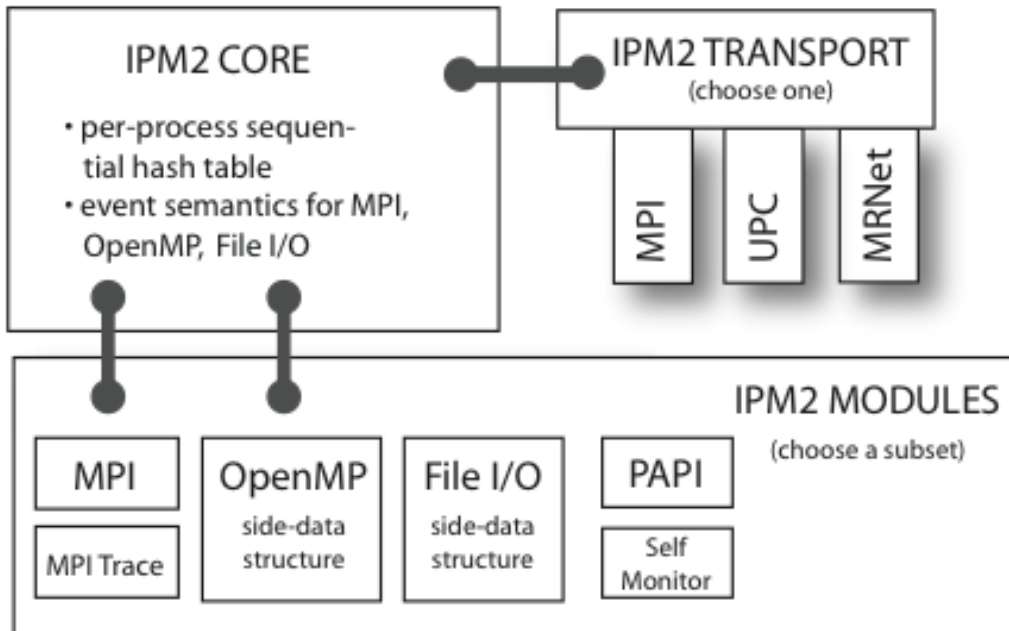
720 nodes, 5760 cores in 9 Scalable Units (SUs) → 61.9 Teraflops
SU = IBM iDataplex rack with 640 Intel Nehalem cores





IPM Overview

- IPM = Integrated Performance Monitoring
- IPM started as POE+ at NERSC
 - “How to profile apps from 400 projects asking for time?”
- Lightweight scalable profiling layer



Provides performance summaries to HPC users and center staff



What can IPM do?

- **What's going on overall in my code?**
 - How much comp, comm, I/O?
 - Where to start with optimization?
- **Provide high level performance numbers with tiny overhead**
 - To get an initial read on application runtimes
 - For allocation/reporting, ERCAP perf data
 - To check the performance weather
- **How is my load balance?**
 - Domain decomposition vs. concurrency (M work on N tasks)



IPM: Origin and Motivation

- **One of many:** There are lots of good vendor supplied tools, we encourage their use
- **Adaptable :** If you can't get what you need from those we can adapt IPM based on your feedback
- **Performance Portability:** IPM provides long-term continuity to performance data between machines, applications, allocations



Using IPM @ NERSC

- 1) Do “module load ipm”, run normally
- 2) Upon completion you get

```
##IPM2v0.xx#####
#####
#
# command      : ./fish -n 10000
# start        : Tue Feb 08 11:05:21 2011      host          : nid06027
# stop         : Tue Feb 08 11:08:19 2011      wallclock    : 177.71
# mpi_tasks    : 25 on 2 nodes                %comm        : 1.62
# mem [GB]     : 0.24                          gflop/sec    : 5.06
...

```

Maybe that’s enough. If so you’re done.
 Have a nice day ☺



Using IPM @ Magellan

- 1) We “module load ipm” for users silently**
- 2) Upon completion you get normal output (leave no tracks)**
- 3) A logfile is written to a shared space**
- 4) NERSC then analyzes 300K+ application profiles**



IPM Profile Report

IPM profile for 5478299.nid00003 - Mozilla Firefox

http://www.nersc.gov/projects/SDSA/workload/N6ipm/mlc_1024/

IPM profile for 5478299.nid00003

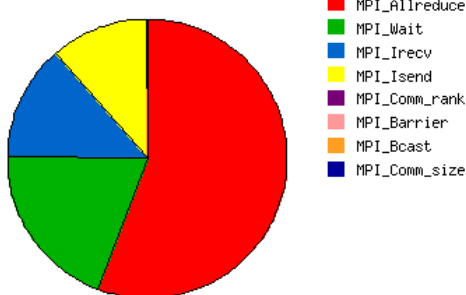
5478299.nid00003

- Load Balance
- Communication Balance
- Message Buffer Sizes
- Communication Topology
- Switch Traffic
- Memory Usage
- Executable Info
- Host List
- Environment
- Developer Info

Powered by IPM

command: ./su3_rmd_ipm

codename:	unknown	state:	running
username:	unknown	group:	unknown
host:	nid00192 (x86_64_Linux)	mpi_tasks:	1024 on 512 hosts
start:	07/31/08/10:22:15	wallclock:	1.01934e+03 sec
stop:	07/31/08/10:39:14	%comm:	22.3665512120588
total memory:	89.7124593999995 gbytes	total gflop/sec:	996.231781348717
switch(send):	0 gbytes	switch(rcv):	0 gbytes

Computation			Communication	
Event	Count	Pop	% of MPI Time	
PAPI_FP_OPS	1015498895269727	*		
PAPI_TOT_CYC	2504745952383680	*		
PAPI_TOT_INS	1938008453032501	*		
PAPI_VEC_INS	416352019568617	*		

Done

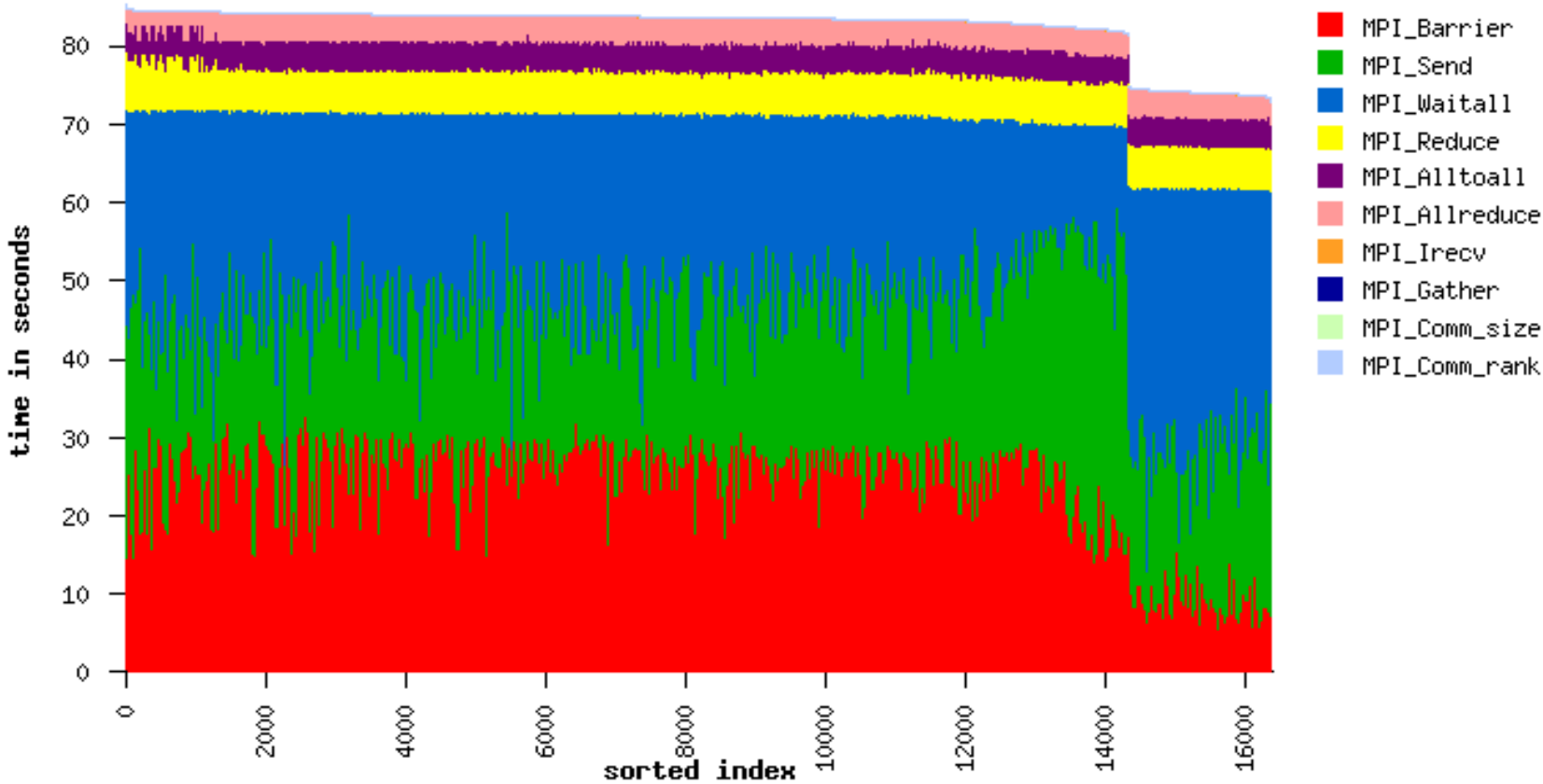


Office of
Science





Load imbalance is a common bottleneck



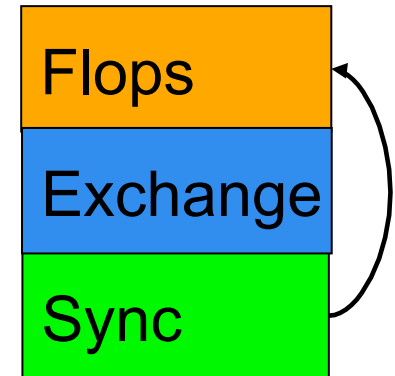


Dynamic disorder: What we miss with IPM

MPI Rank →

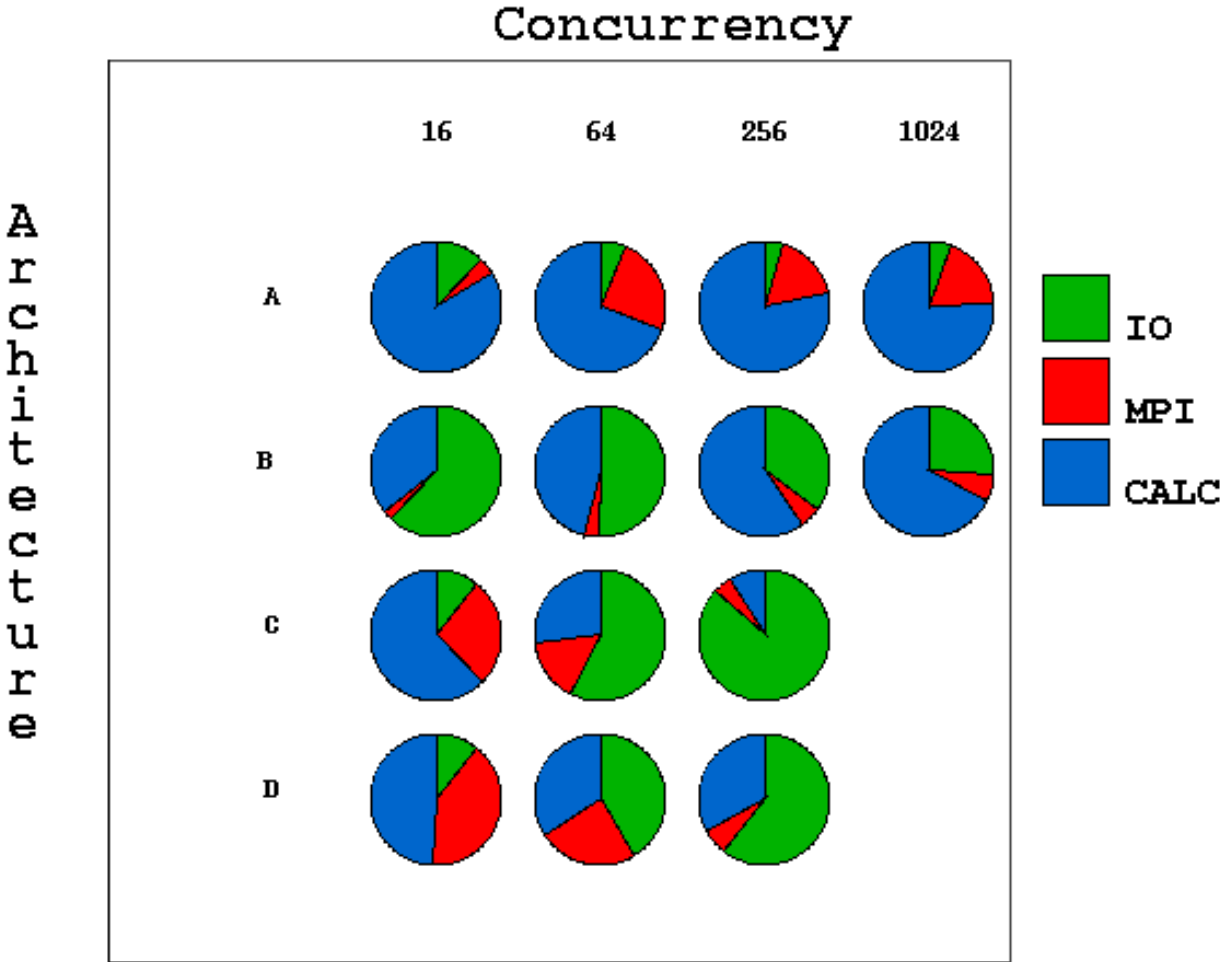


Time →



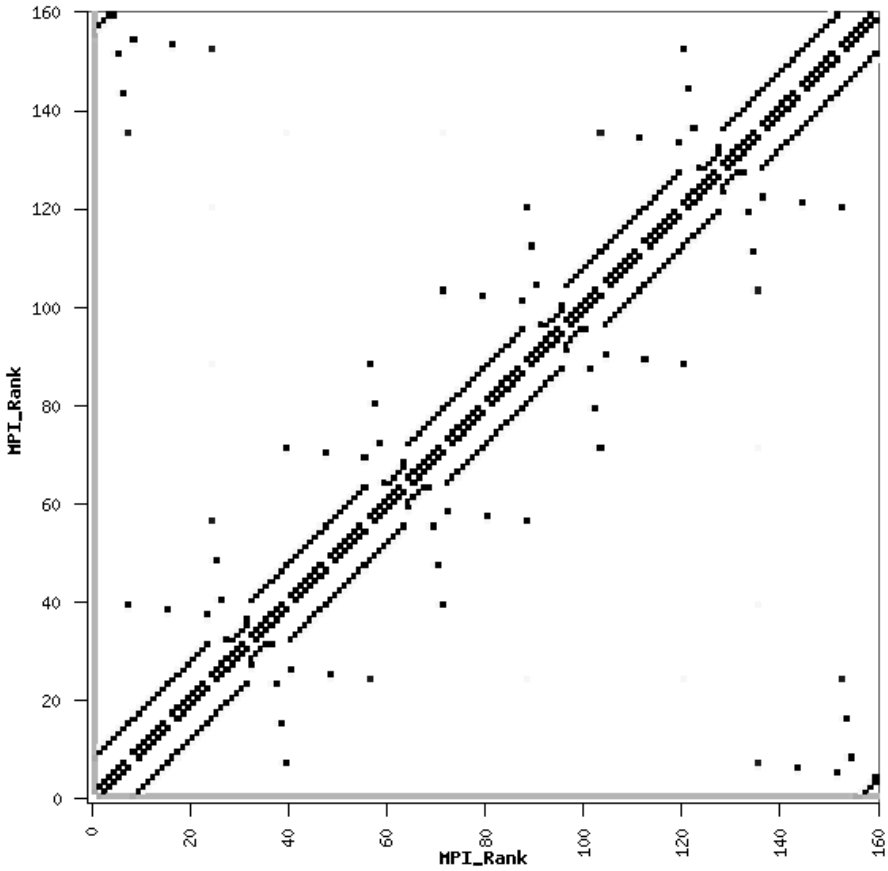


Performance Portability

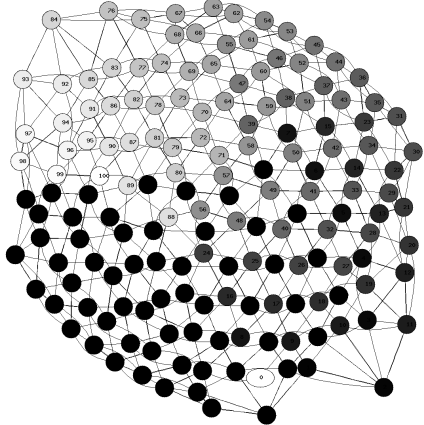
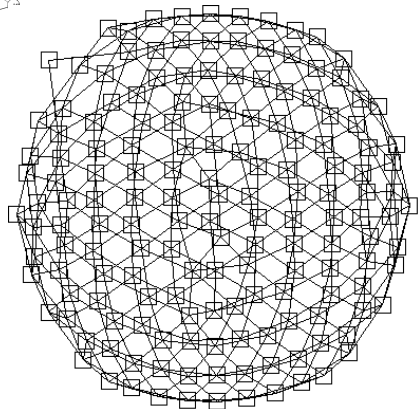
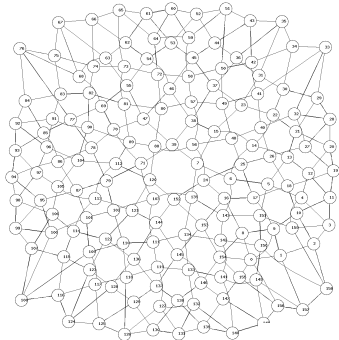




Application Topology

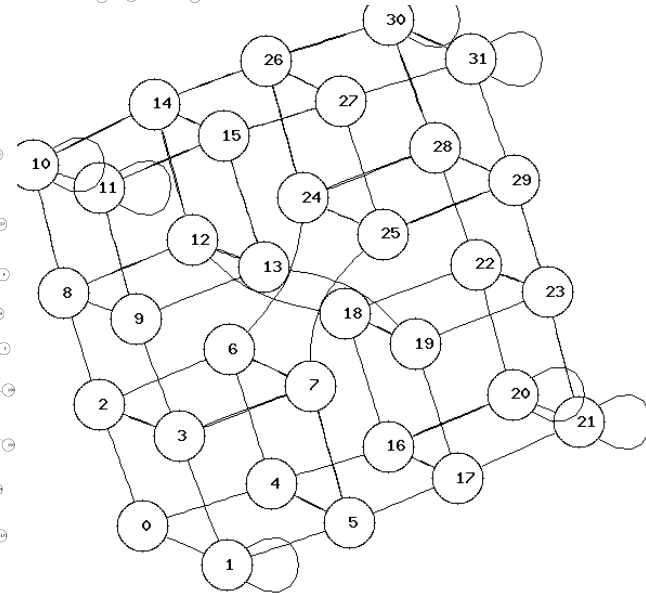
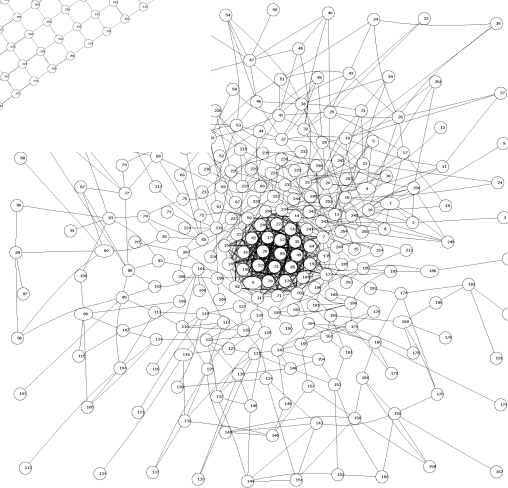
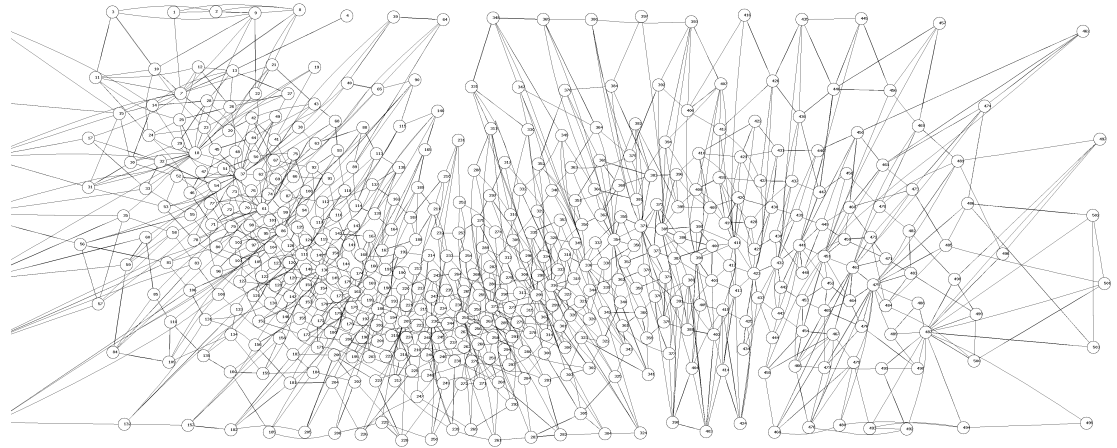
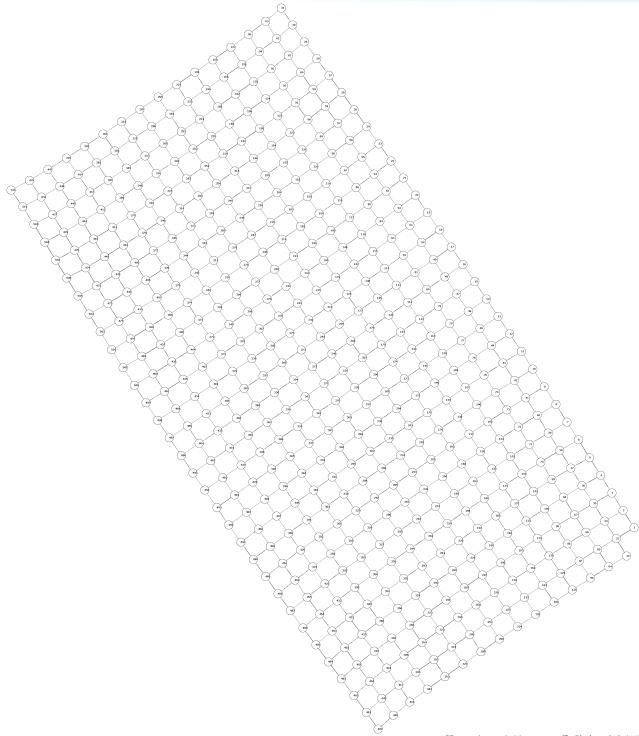


- 0.00389099 MB
- 0.00311279 MB
- 0.00233459 MB
- 0.00155640 MB
- 0.00077820 MB
- 0.00000000 MB





Application Shape and Symmetry



U.S. DEPARTMENT OF
ENERGY

Office of
Science





on to workloads...

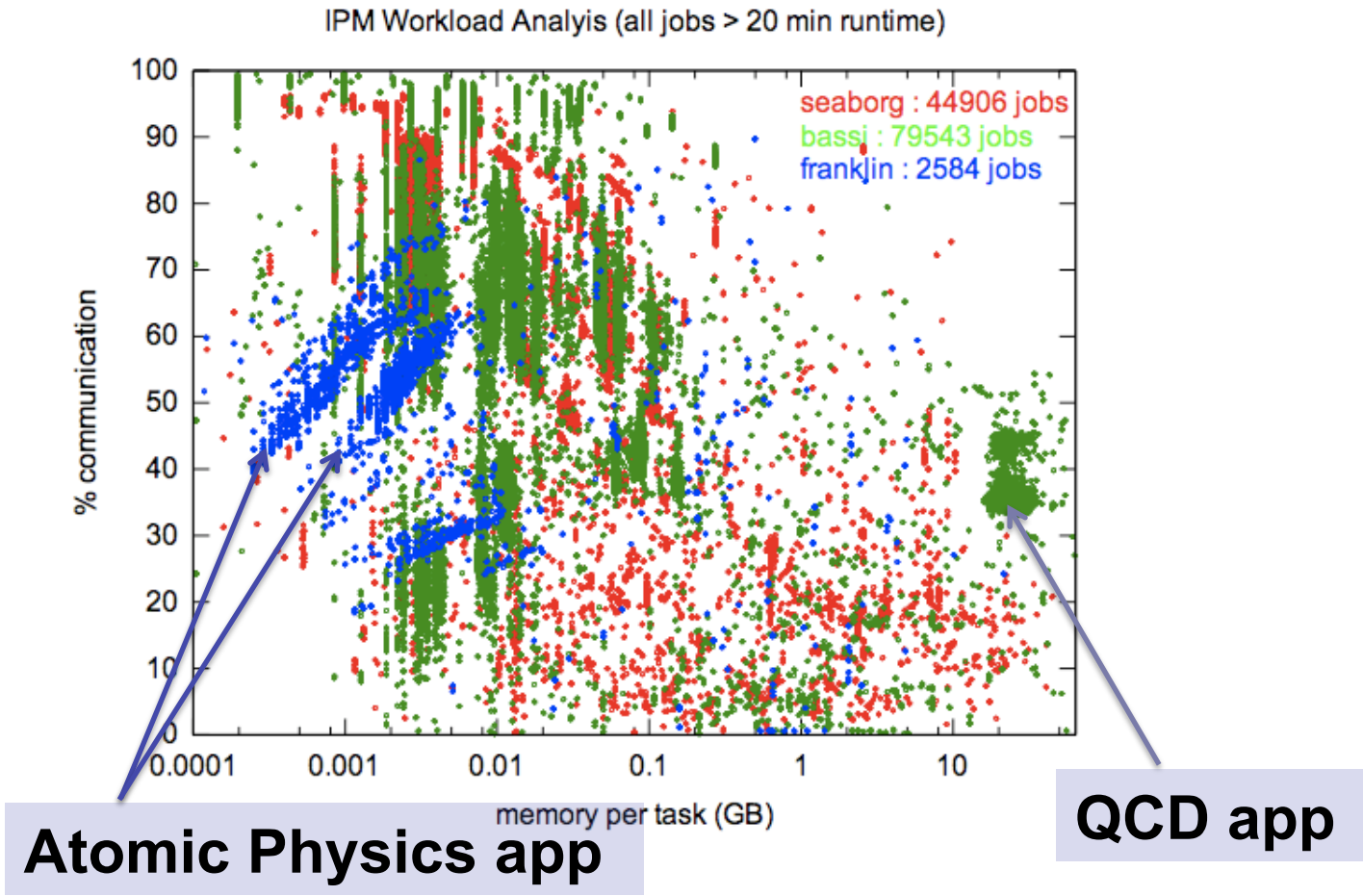


IPM: Ease of use → It gets used

- **We have collected over 300k IPM profiles**
 - **Jobs running longer than 20min (400k otherwise)**
 - **Covers a period of 6 years**
 - **Covers 5 HPC architectures**
- **This may pave the way for using IPM by default on all NERSC production systems**



300K IPM Application Profiles



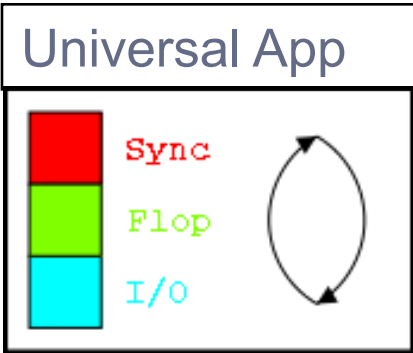
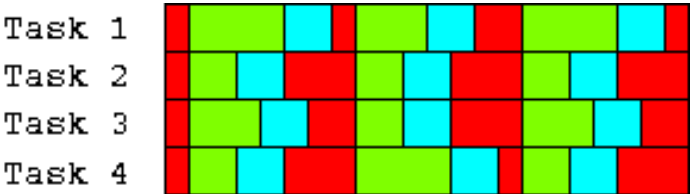


Generalities in Scalability and Performance



Load Balance : cartoon

Unbalanced:



Balanced:

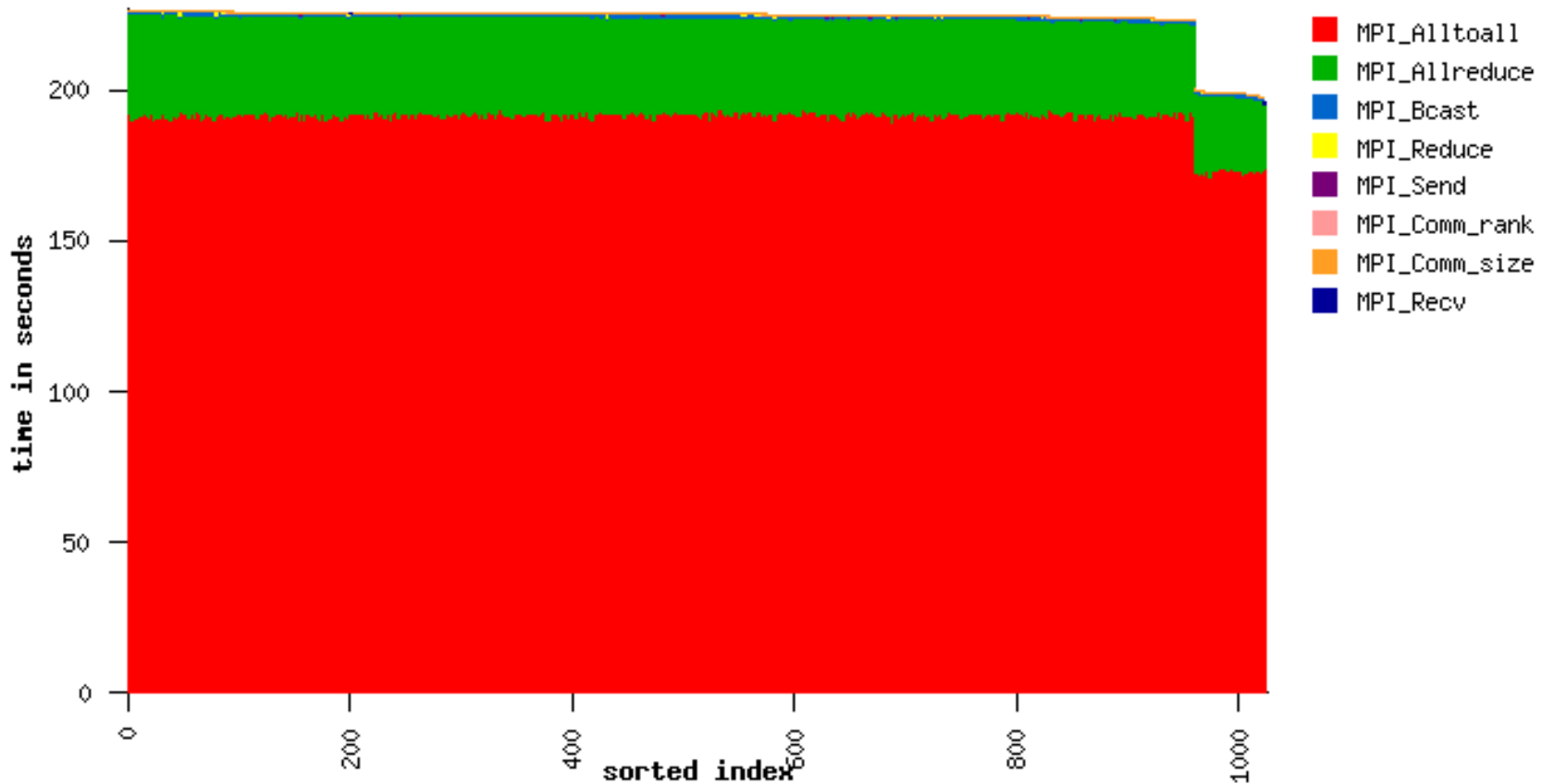


Time saved by load balance



Load (Im)balance

Communication Time: 64 tasks show 200s, 960 tasks show 230s



MPI ranks sorted by total communication time

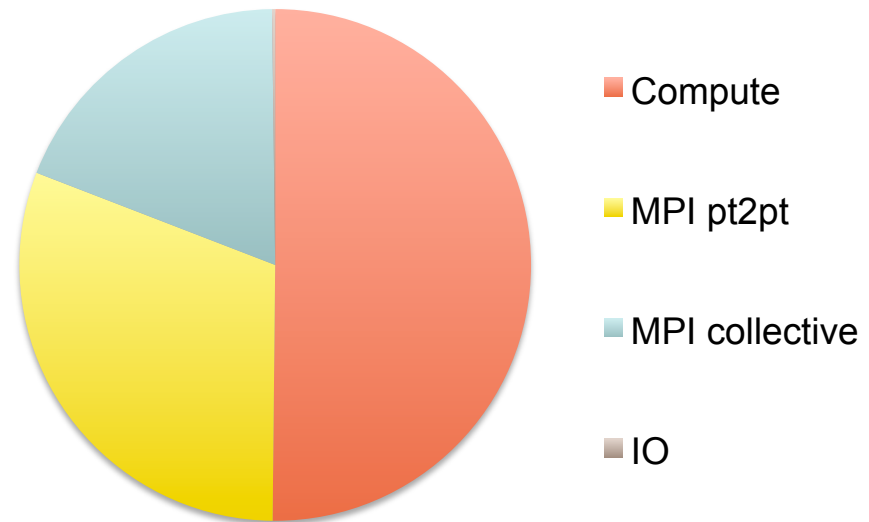


Some specific examples From Magellan



Workload Analysis on Magellan

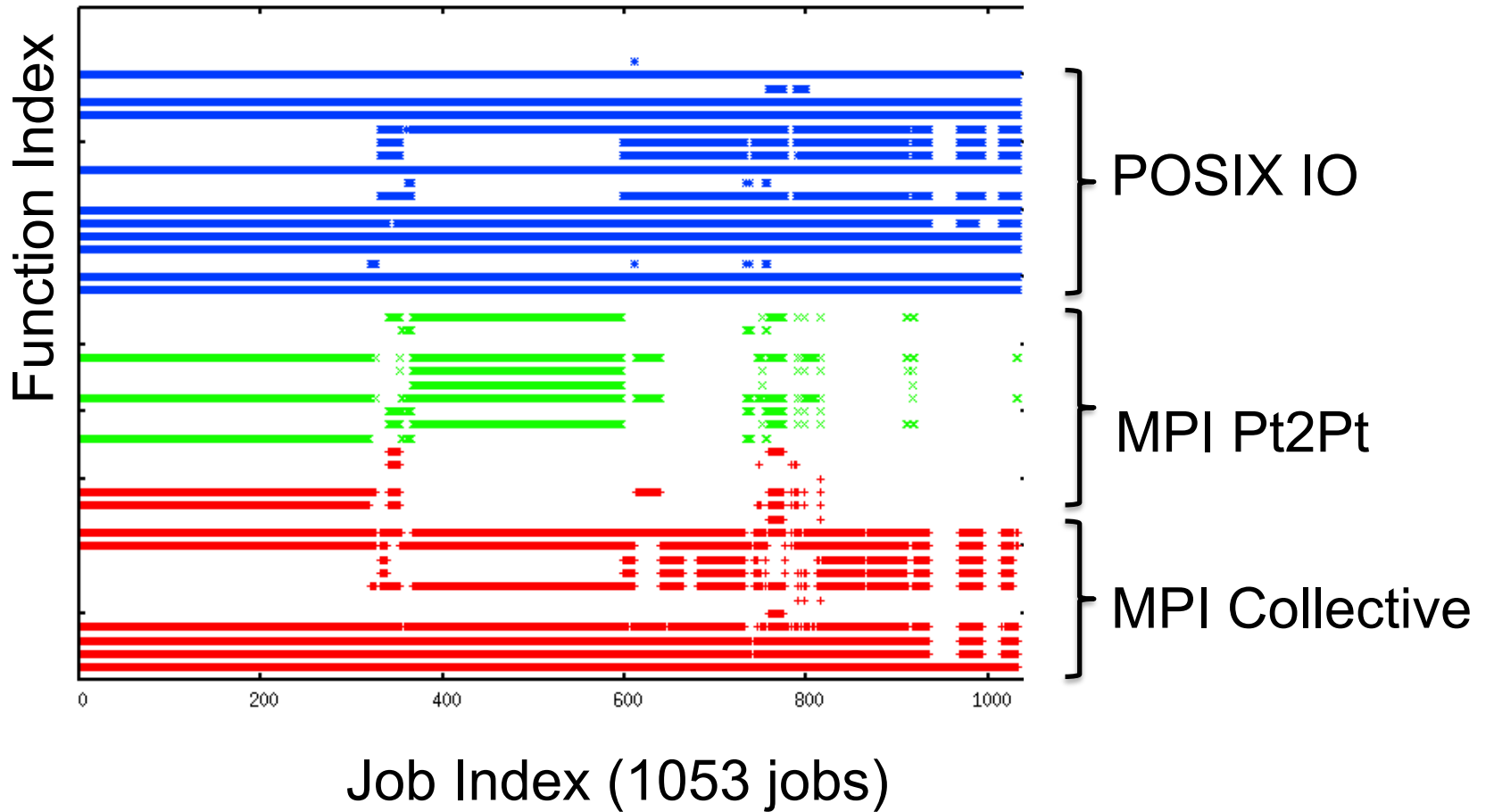
- **Most HPC profiling is done on an opt-in basis. Users deploy tools to understand their code.**
- **Magellan profiling is system wide, passive, and automatic, a workload approach.**
- **October 4-27 2010 :**
 - **1053 batch jobs**
 - **37 users**
 - **18 applications**
 - **4K cores**
 - **Preliminary results(*)**



(*) does not yet include non-MPI jobs



Workload Coverage: which jobs use which resources

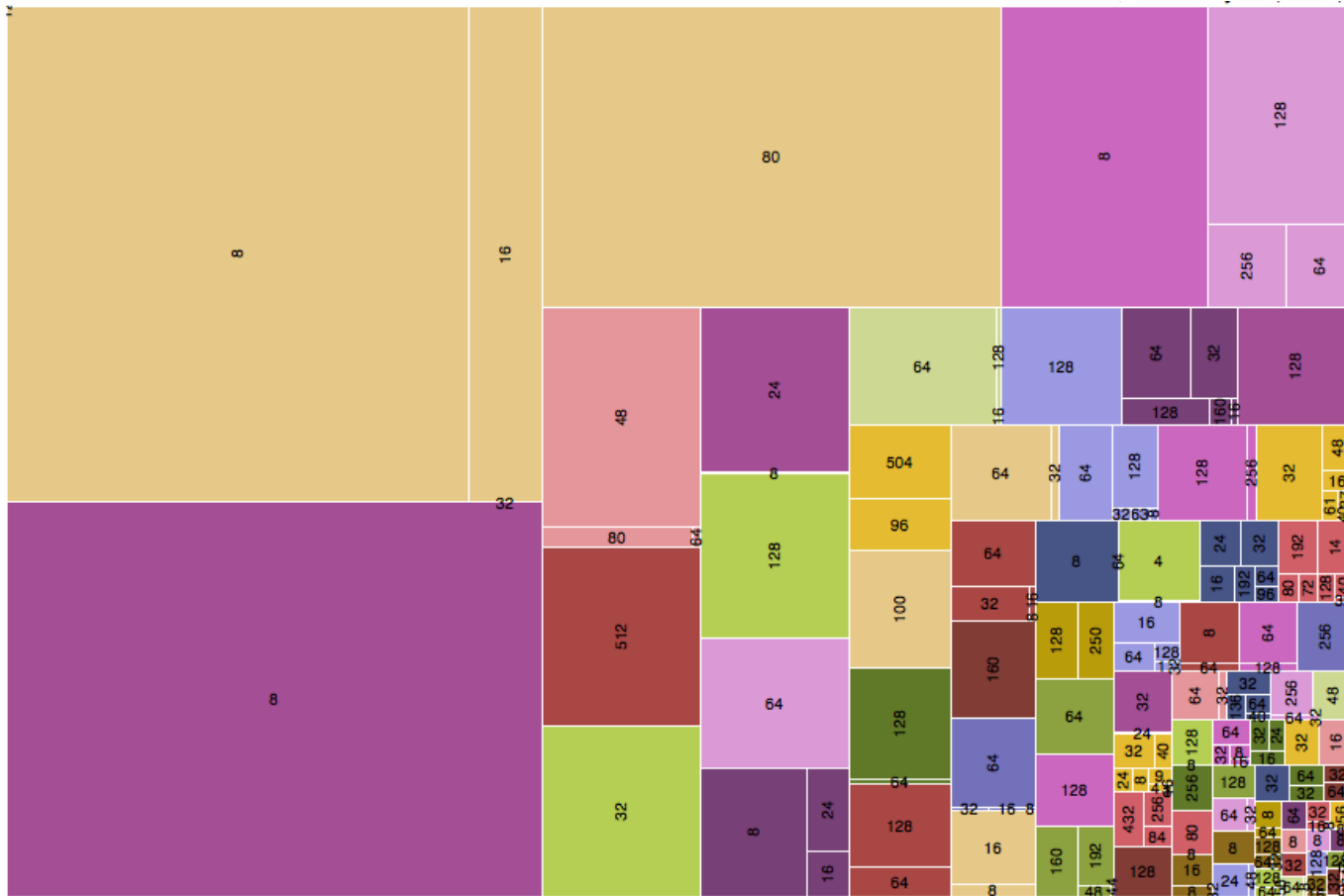




Examining 22K Magellan Jobs

207 user-concurrency pairs

size=core-hours
color=user
label=cores



U.S. DEPARTMENT OF
ENERGY

Office of
Science





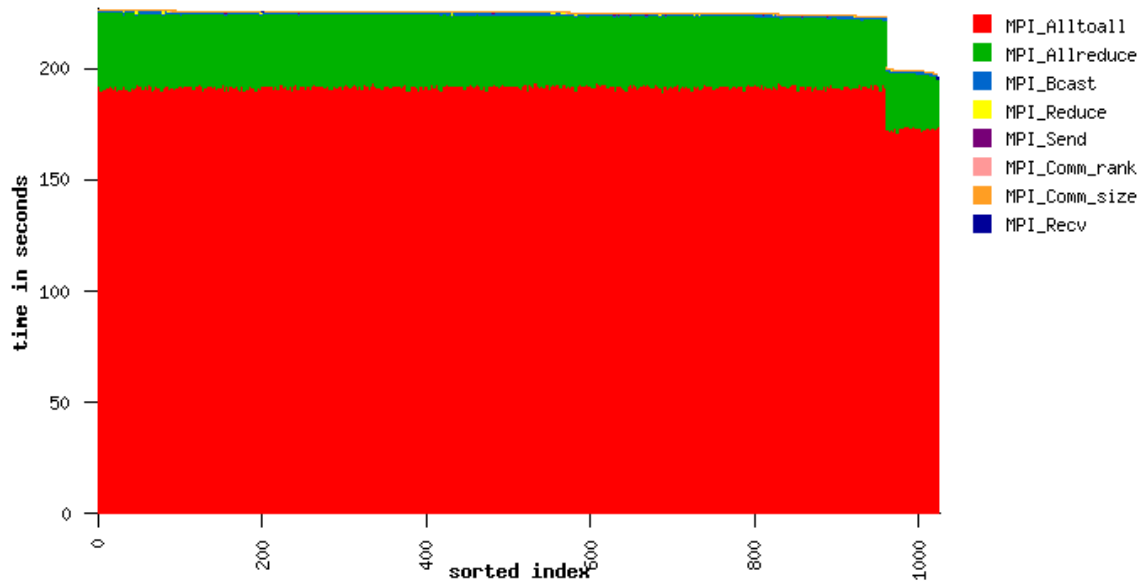
Magellan Workload

- **How to proceed making workload level statements about resource needs?**
- **How to find bottlenecks?**
- **Each run is potentially distinct:
Changes in code, inputs, compilation,
runtime, system of study, etc.**
- **Let revisit the familiar case of load imbalance**



Inferences on workloads

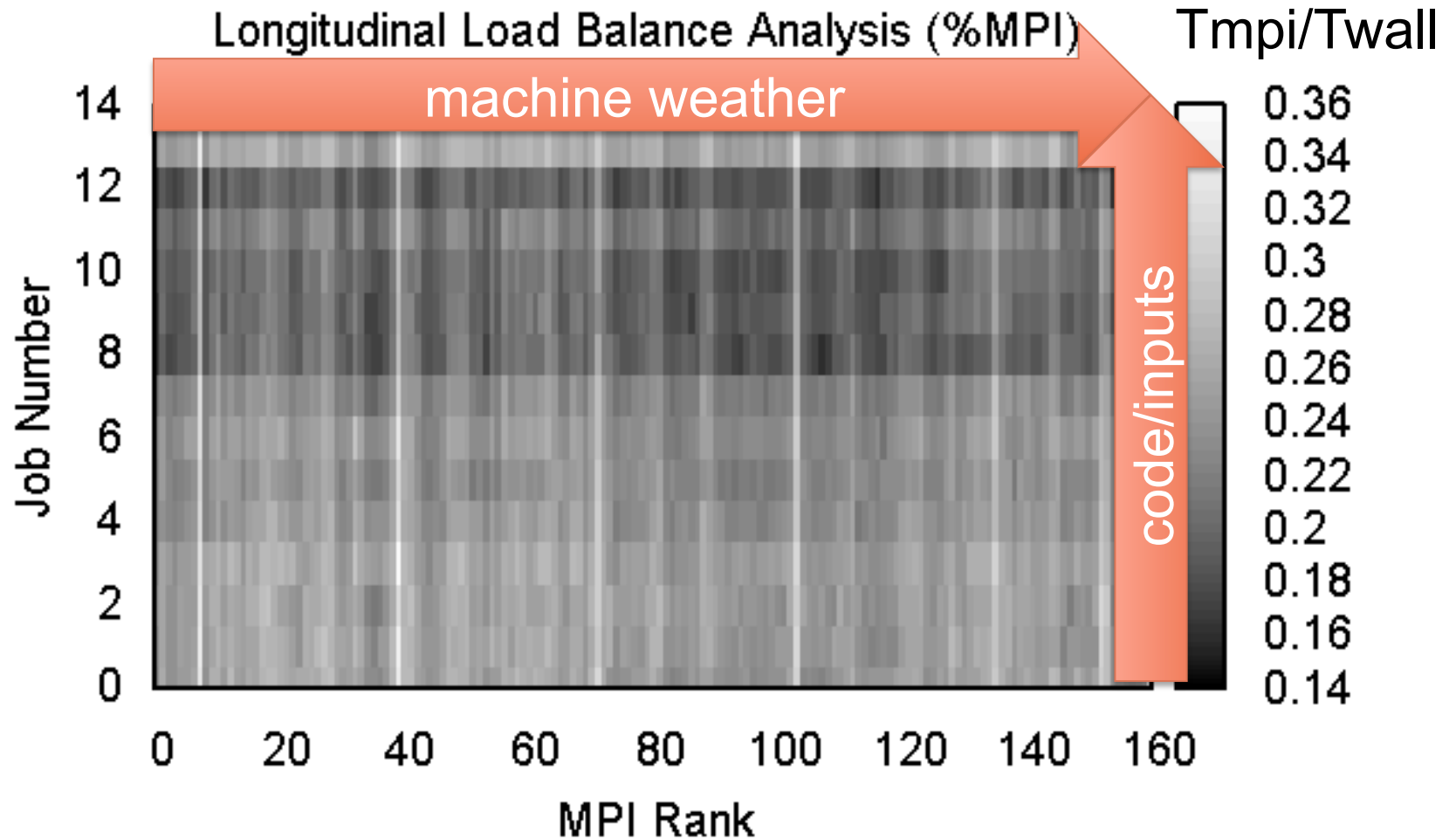
- What changes between runs?
- What stays the same?



code
input
configs
batch script
runtime
machine
switch traffic

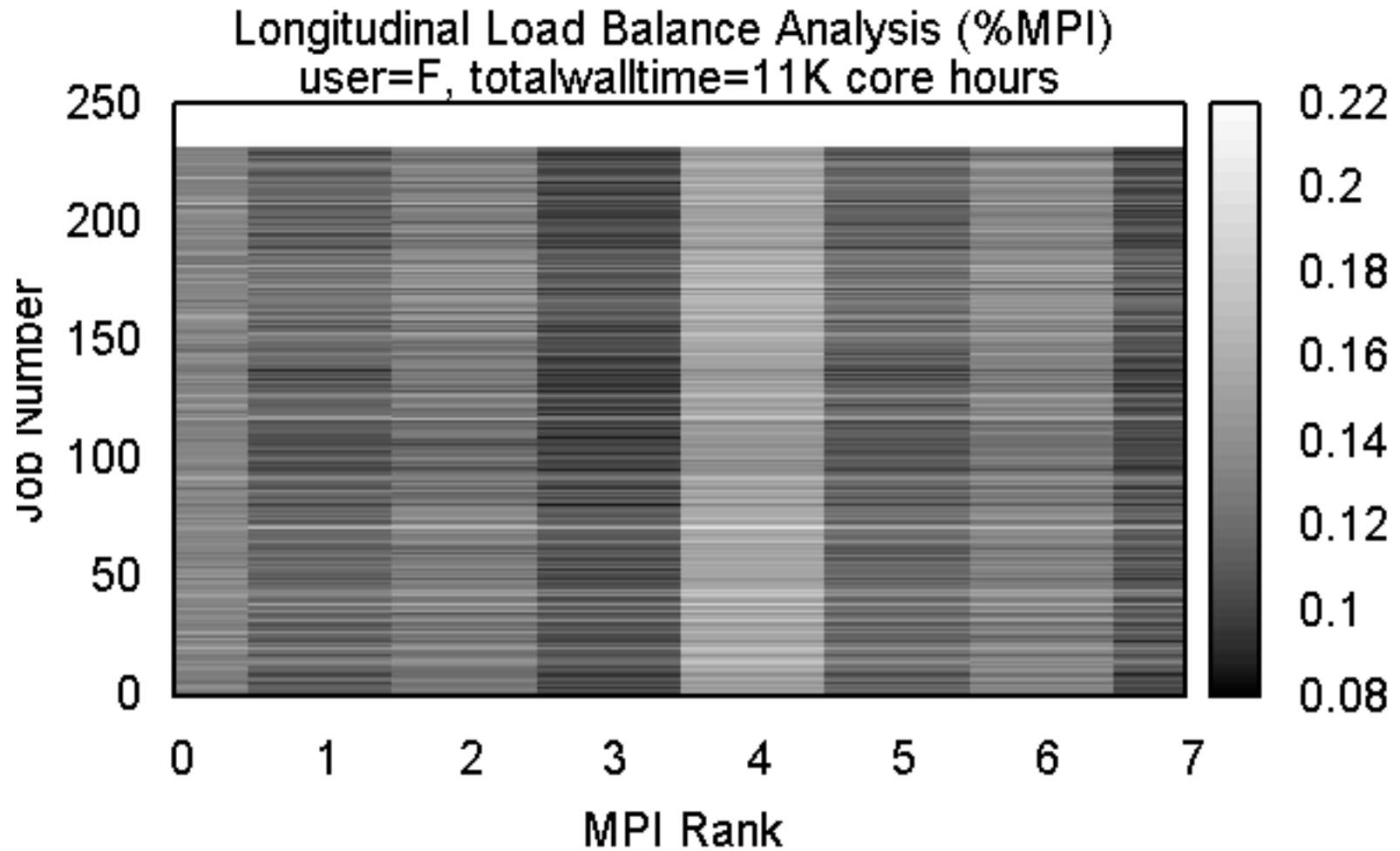


User D runs code A 13 times





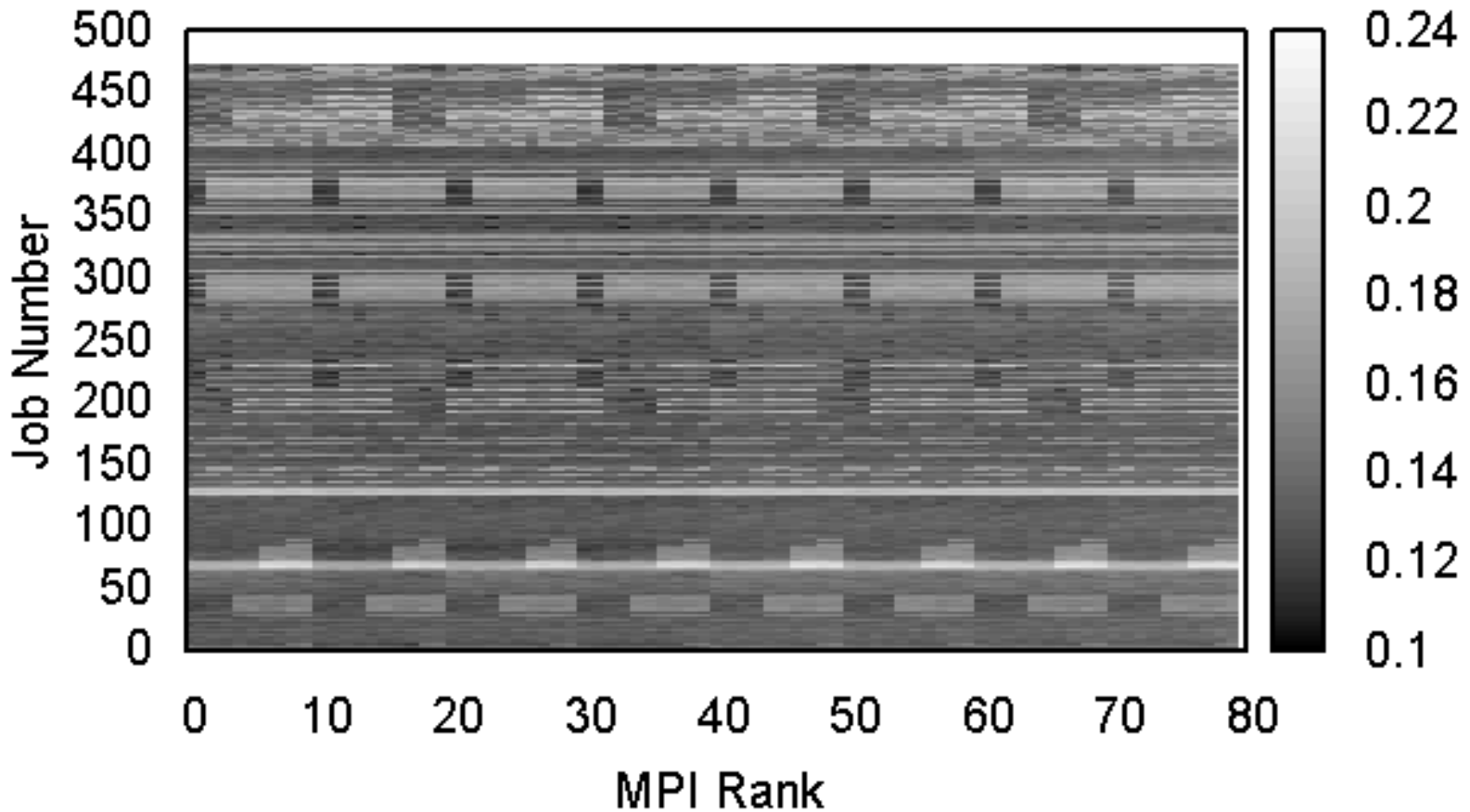
User F runs Code V 220 times





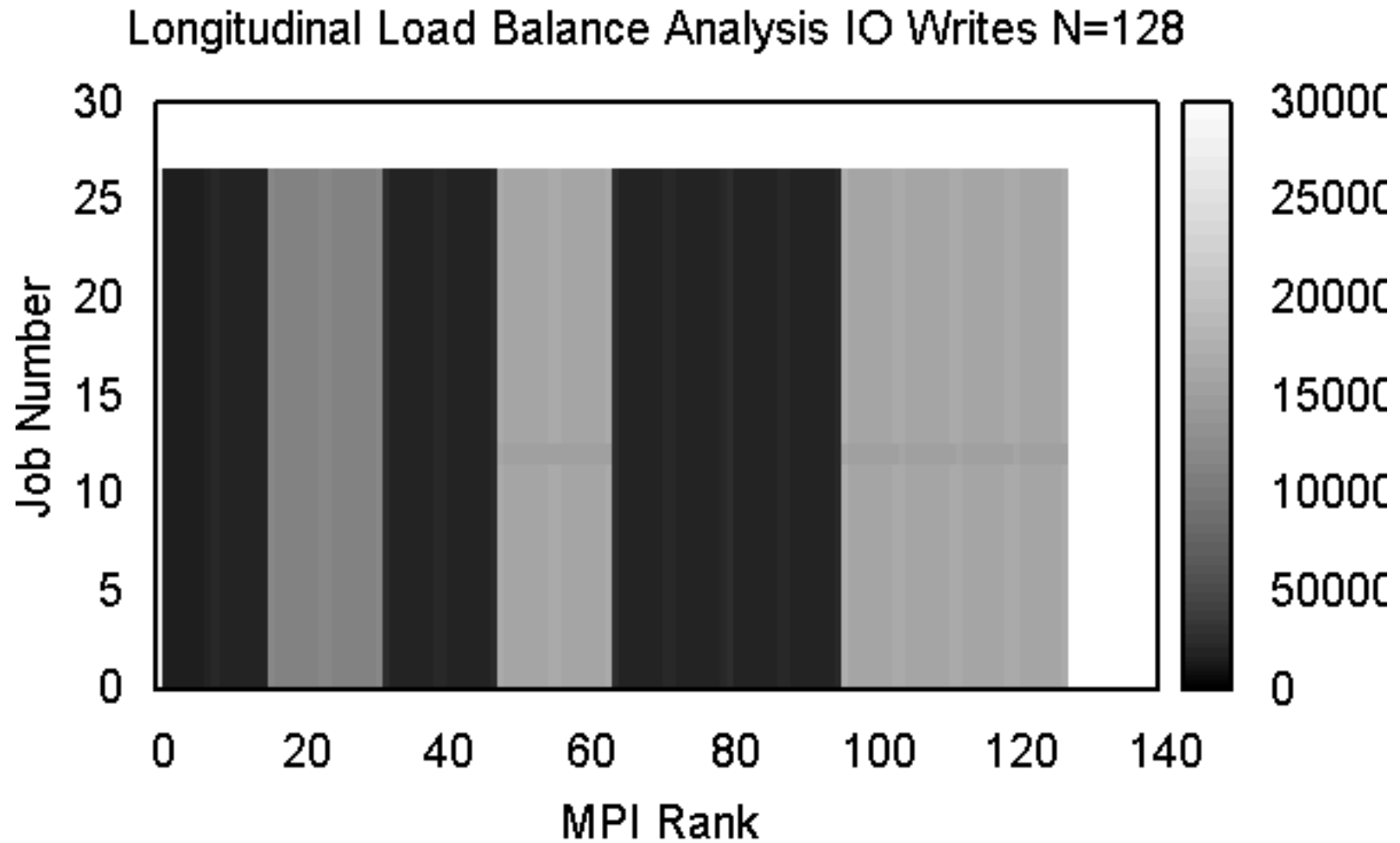
Last two were easy: discontinuities/structure

Longitudinal Load Balance Analysis user=C N=80





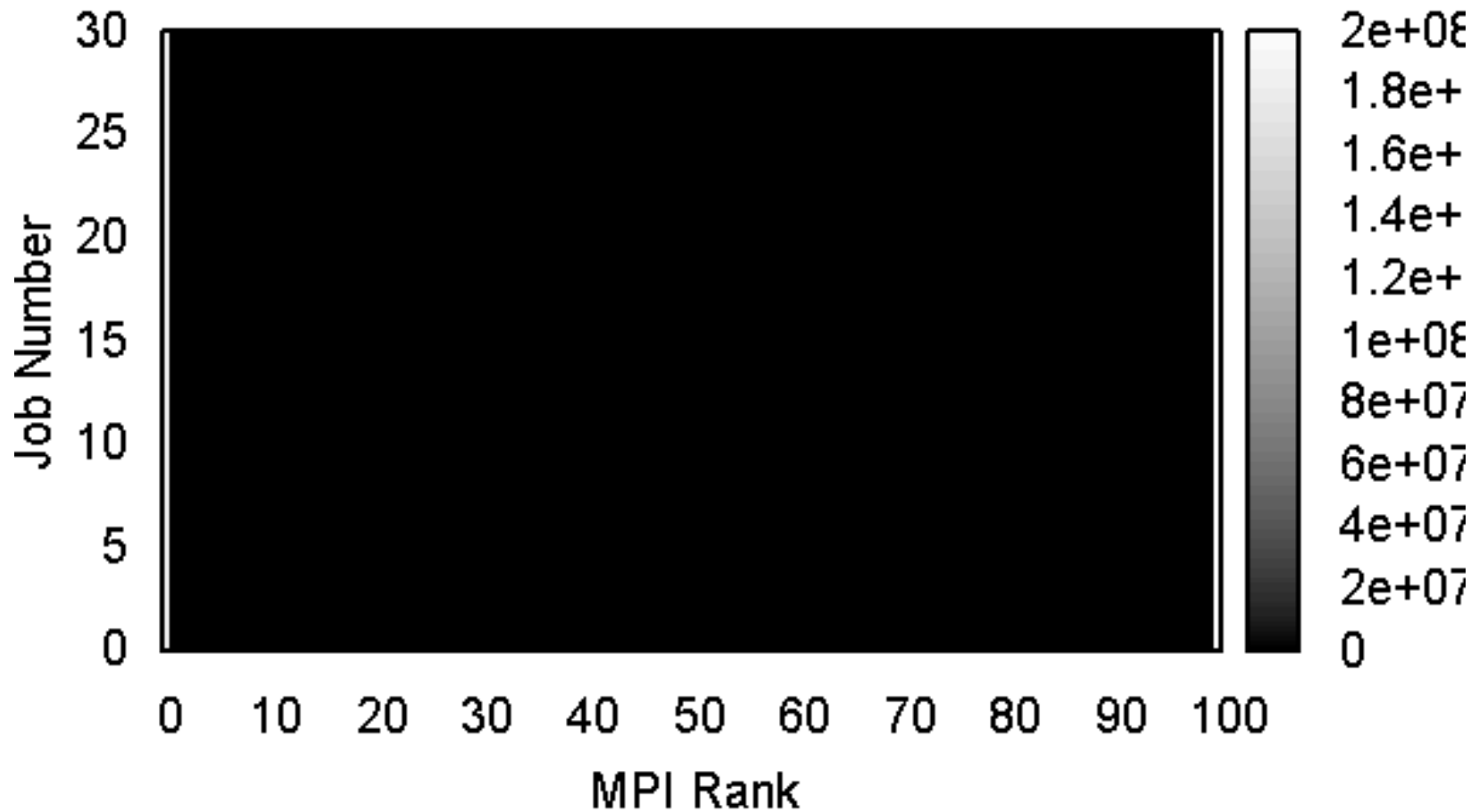
We can do this with IO too.





Ever seen this IO strategy?

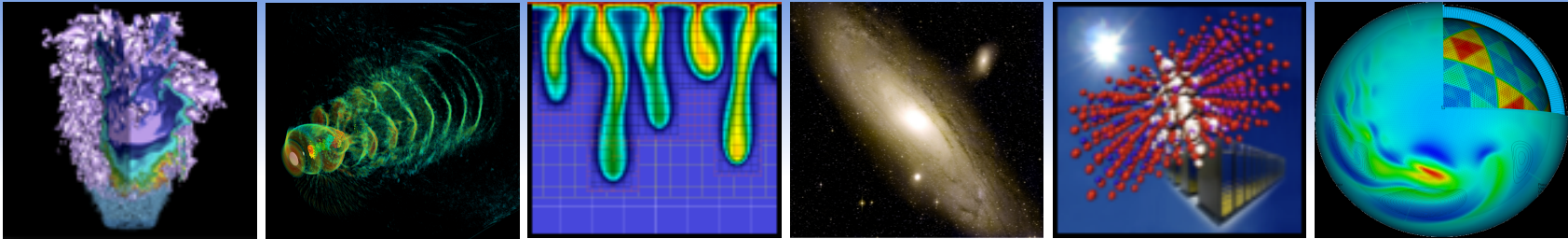
Longitudinal Load Balance Analysis IO:bytes W N=100





Conclusions

- **“Always on” profiling looks doable/promising.**
- **Performance in practice vs. performance in principle.**
- **Comparison across jobs can allow some confidence in which bottlenecks are worth attention**
- **If you have ideas about/for IPM, I am interested in collaborations.**



Thank you



Contact Info:
David Skinner
deskinner@lbl.gov
<http://ipm-hpc.sf.net>

