Performance Engineering and Debugging HPC Applications

David Skinner
deskinner@lbl.gov

U.S. DEPARTMENT OF ENERGY | Office of Science

NERSC National Energy Research Scientific Computing Center

BERKELEY LAB Lawrence Berkeley National Laboratory

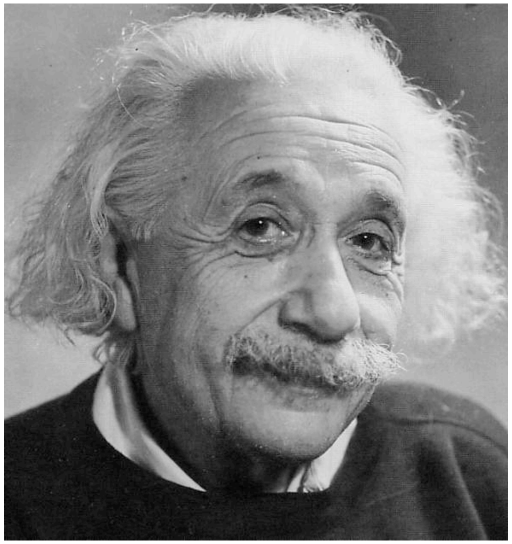
Today: Tools for Performance and Debugging

- **Principles**
  - Topics in performance scalability
  - Examples of areas where tools can help
- **Practice**
  - Where to find tools
  - Specifics to NERSC and Hopper

Big Picture of Scalability and Performance

Performance is Relative

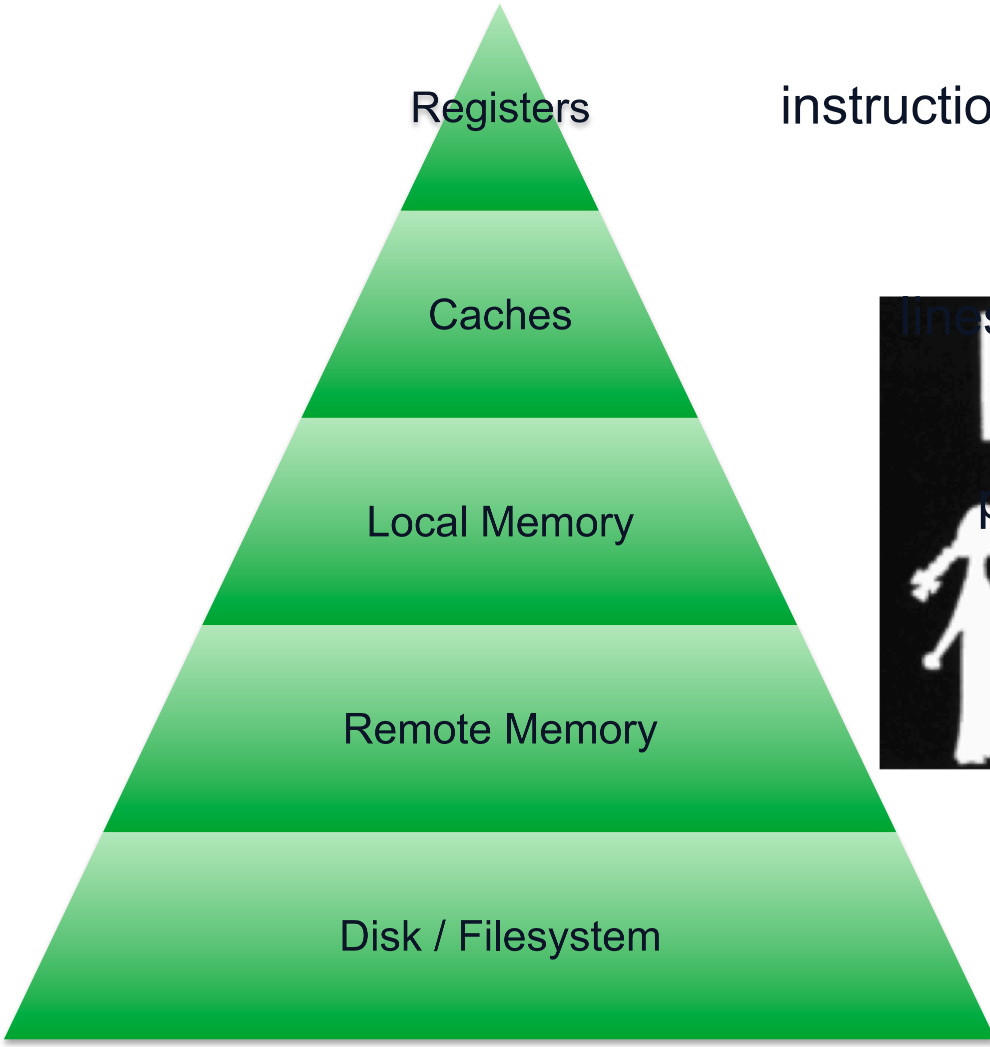
- **To your goals**
  - Time to solution, $T_{queue}+T_{run}$
  - Your research agenda
  - Efficient use of allocation



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

Suggestion:
Focus on specific use cases as opposed to making *everything* perform well.
Bottlenecks can shift.

Performance is Hierarchical



instructions & operands

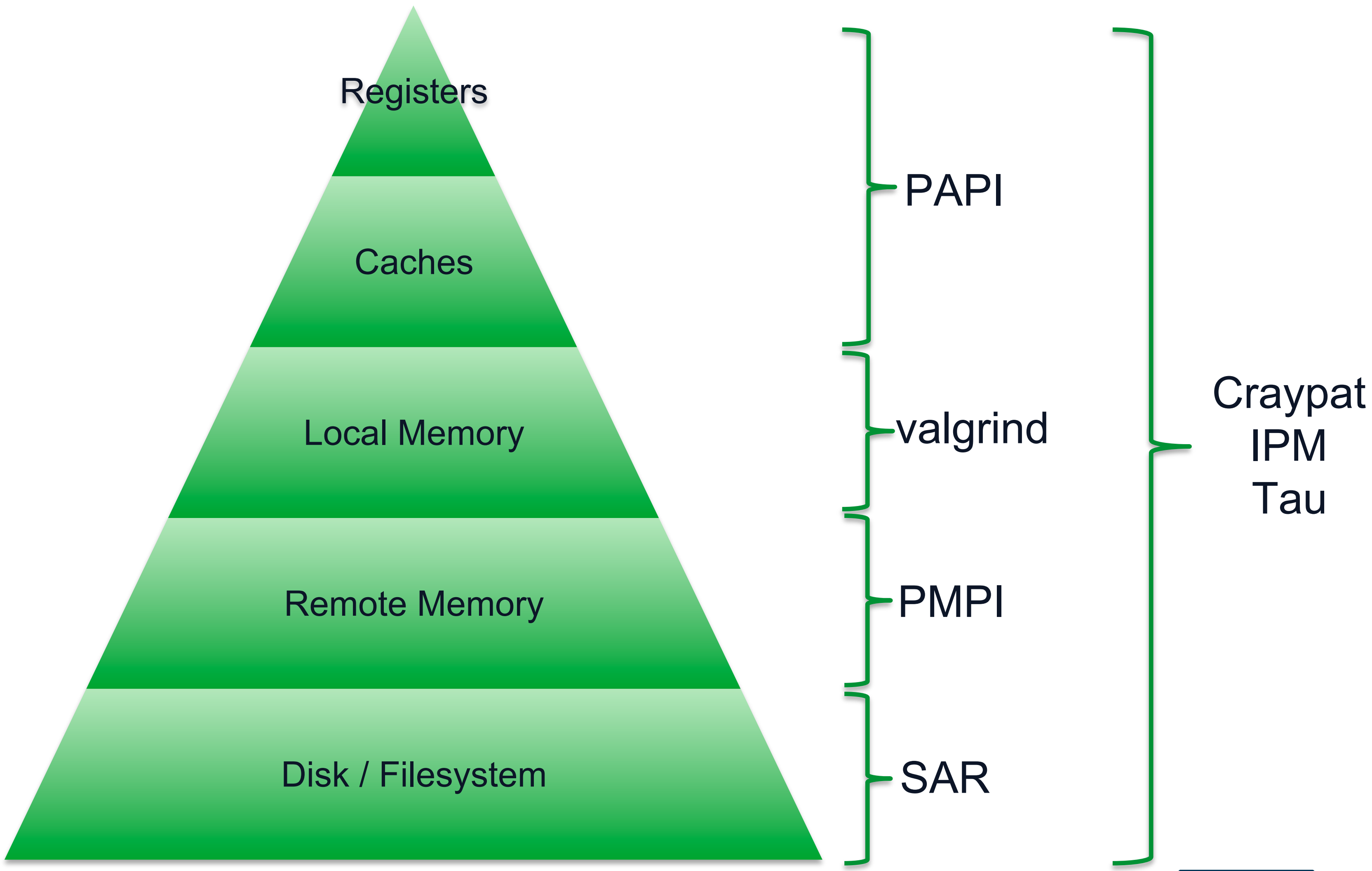
lines

pages

messages

blocks, files

Tools are Hierarchical



Using the right tool

Tools can add overhead to code execution

- What level can you tolerate?

Tools can add overhead to scientists

- What level can you tolerate?

Scenarios:

- Debugging code that ~isn't working
- Performance debugging
- Performance monitoring in production

One tool example: IPM on XE

1) Do "module load ipm", link with $IPM, then 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 ☺

HPC Tool Topics

- **CPU and memory usage**
  - FLOP rate
  - Memory high water mark
- **OpenMP**
  - OMP overhead
  - OMP scalability (finding right # threads)
- **MPI**
  - % wall time in communication
  - Detecting load imbalance
  - Analyzing message sizes

Examples of HPC tool usage

Scaling: definitions

- **Scaling studies involve changing the degree of parallelism. Will we be change the problem also?**
- **Strong scaling**
  - Fixed problem size
- **Weak scaling**
  - Problem size grows with additional resources
- **Speed up = $T_s/T_p(n)$**
- **Efficiency = $T_s/(n*T_p(n))$**

Be aware there are multiple definitions for these terms

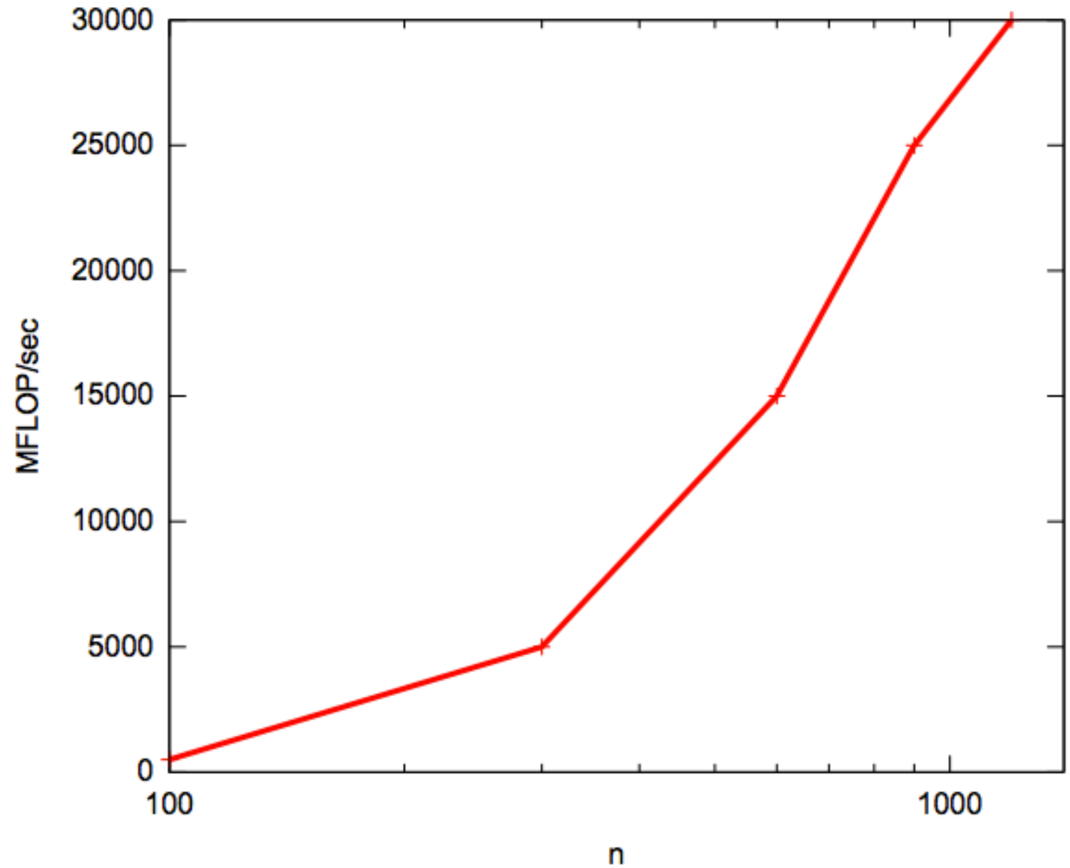
Conducting a scaling study

With a particular goal in mind, we systematically vary concurrency and/or problem size

Example:

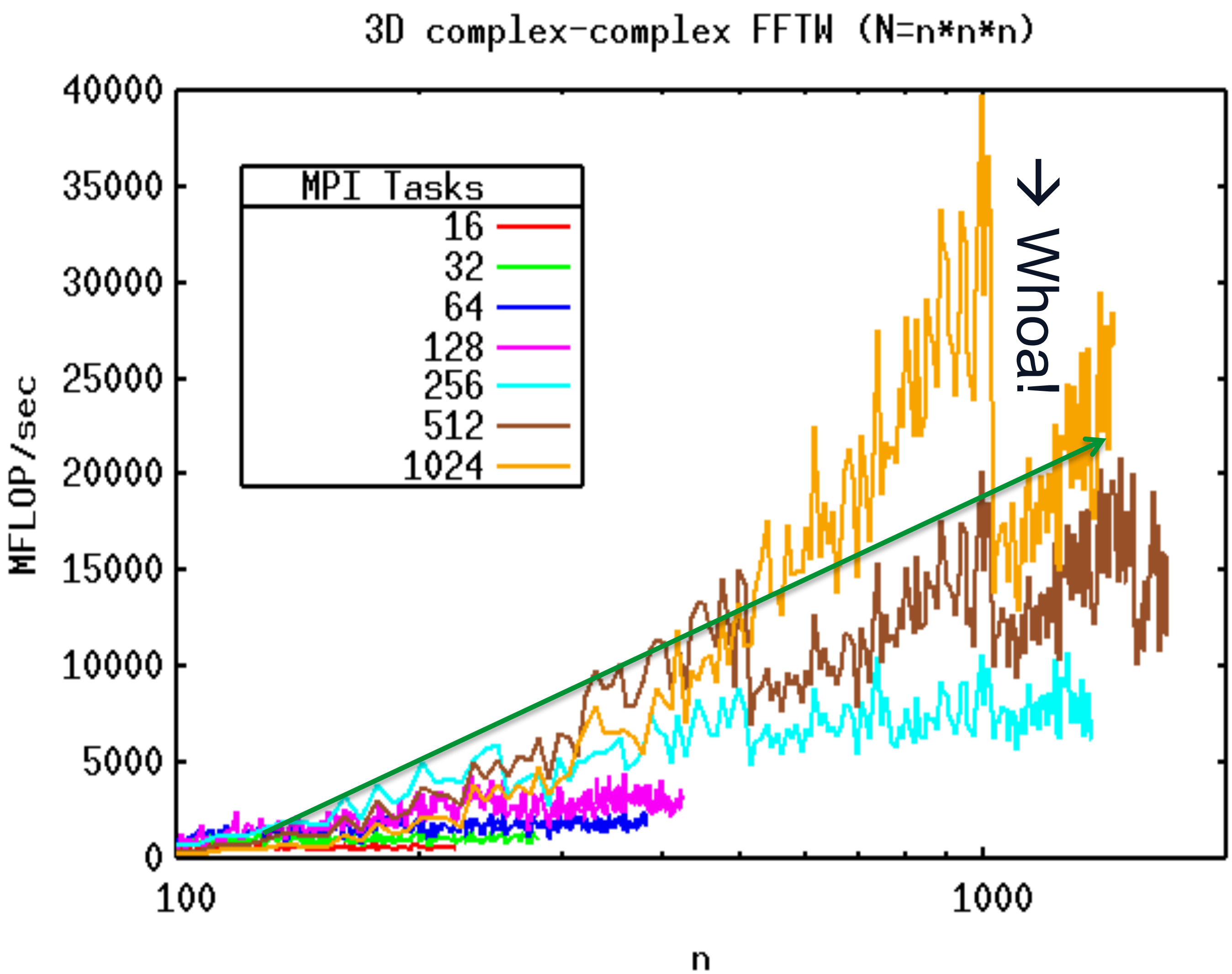
How large a 3D (n^3) FFT can I efficiently run on 1024 cpus?

Looks good?



Let’s look a little deeper….

The scalability landscape



Why so bumpy?

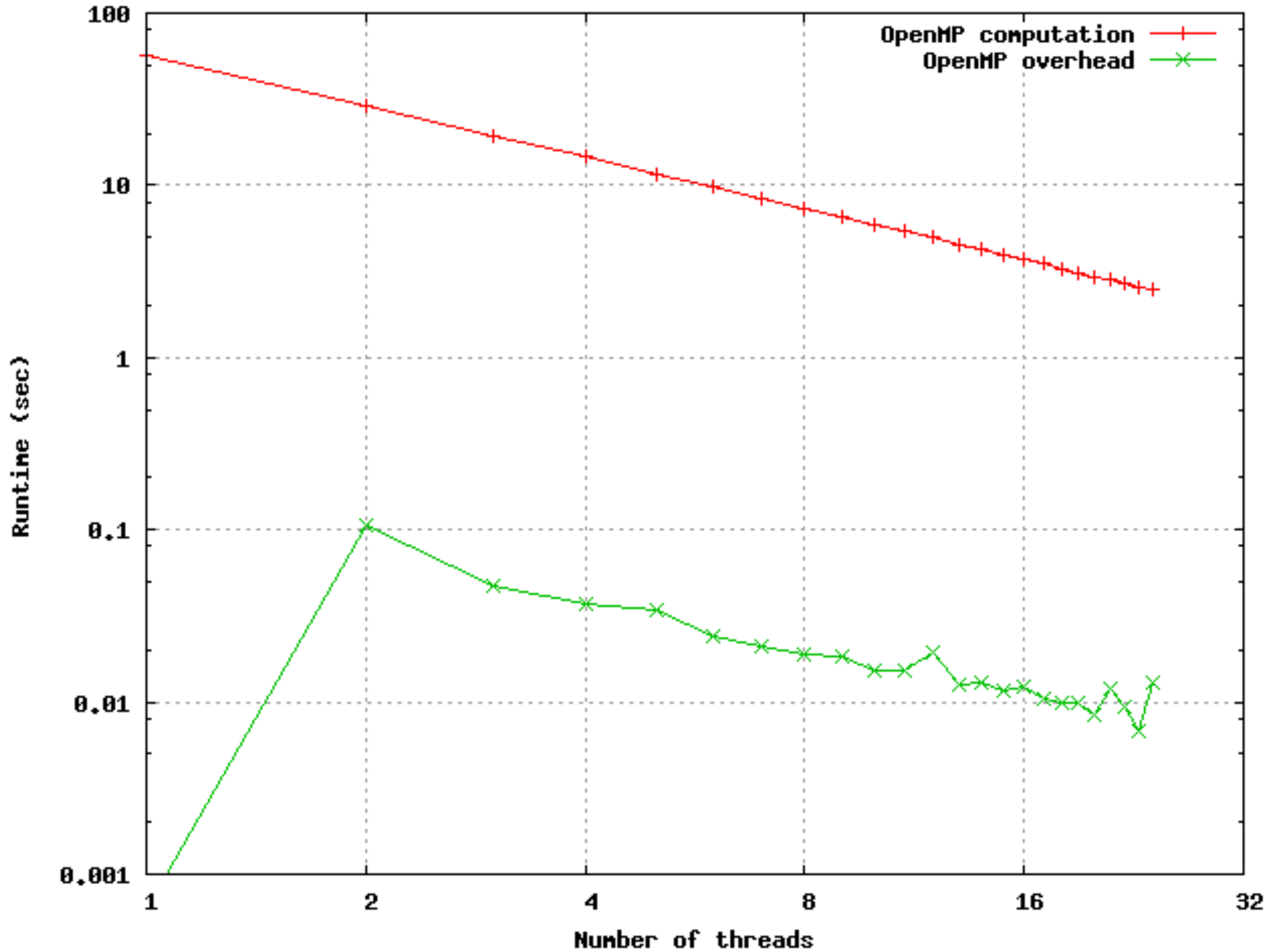
- Algorithm complexity or switching
- Communication protocol switching
- Inter-job contention
- ~bugs in vendor software

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB Lawrence Berkeley National Laboratory

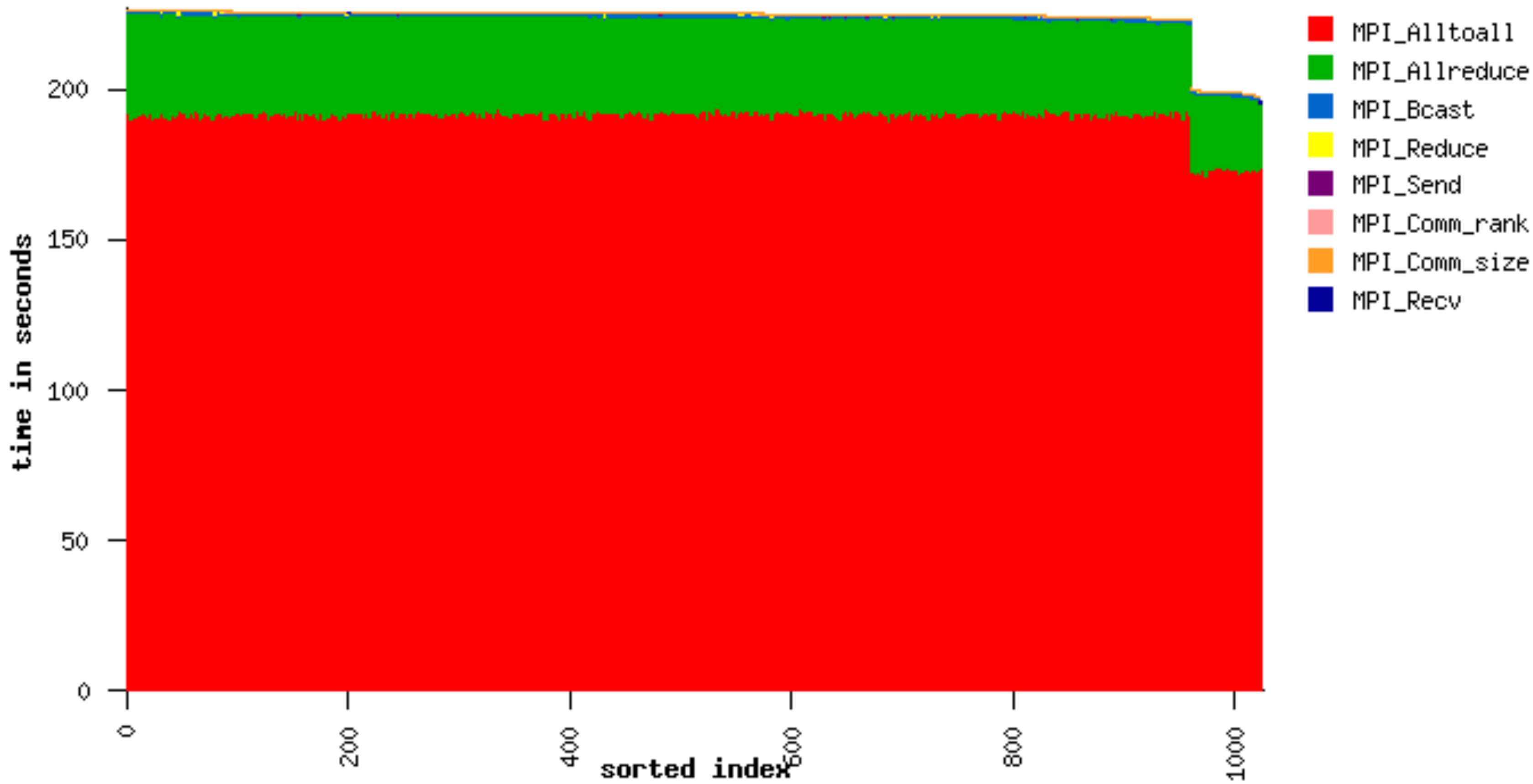
Not always so tricky

Main loop in jacobi_omp.f90; ngrid=6144 and maxiter=20



Load (Im)balance

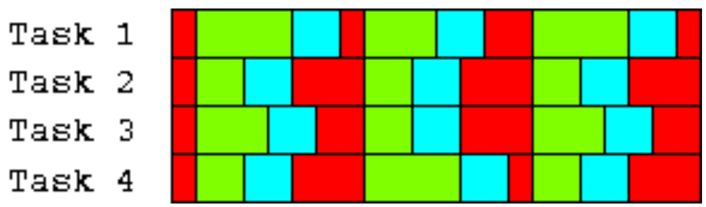
Communication Time: 64 tasks show 200s, 960 tasks show 230s



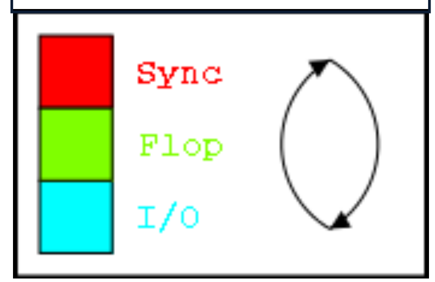
MPI ranks sorted by total communication time

Load Balance : cartoon

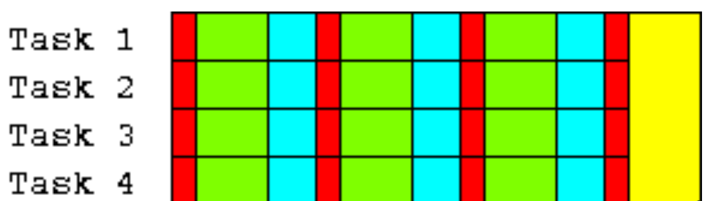
Unbalanced:



Universal App

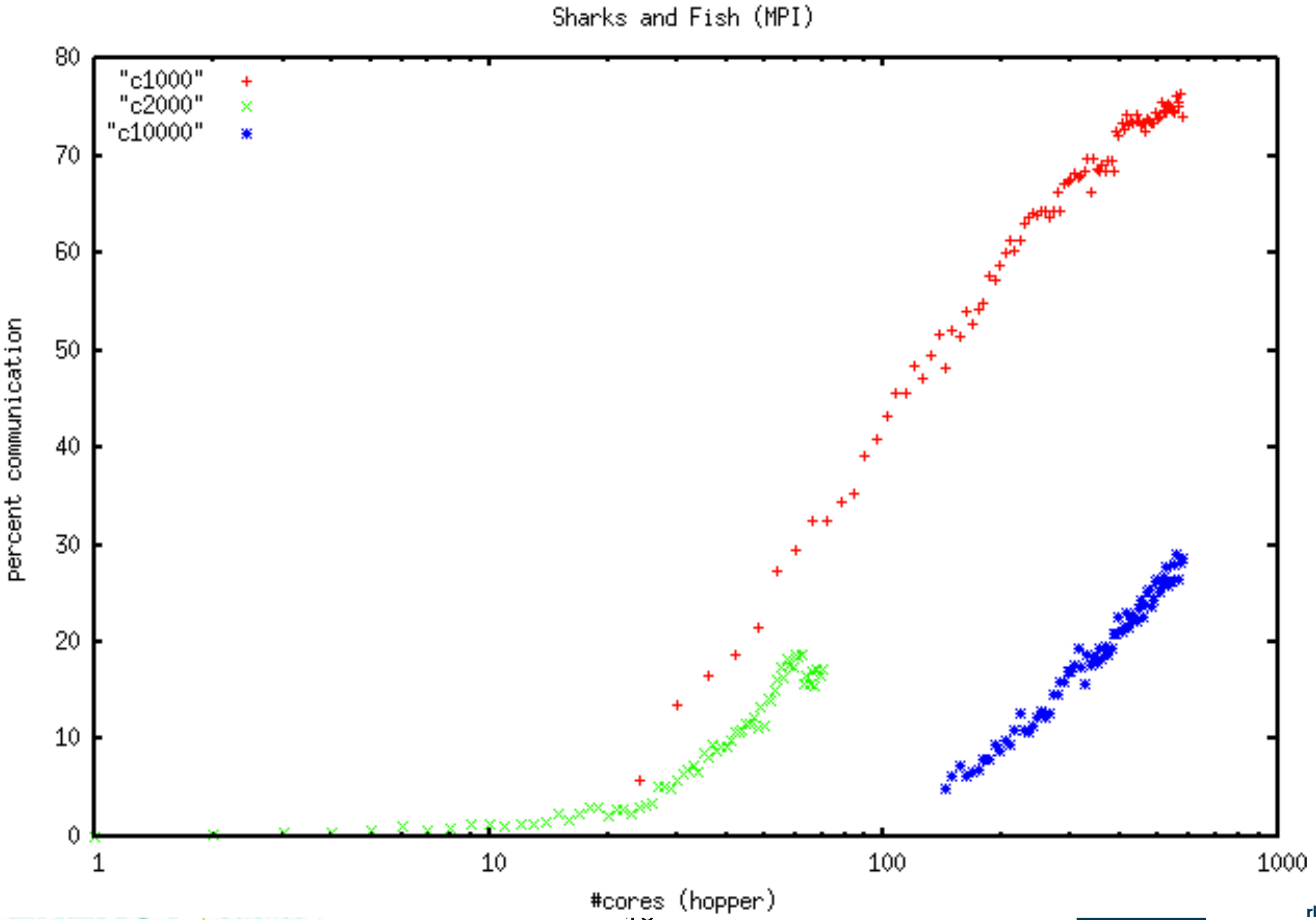


Balanced:



Time saved by load balance

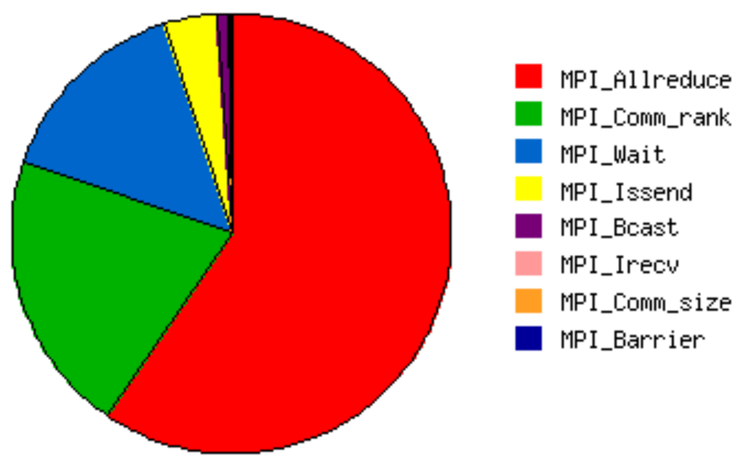
Too much communication



Simple Stuff: What's wrong here?

Communication

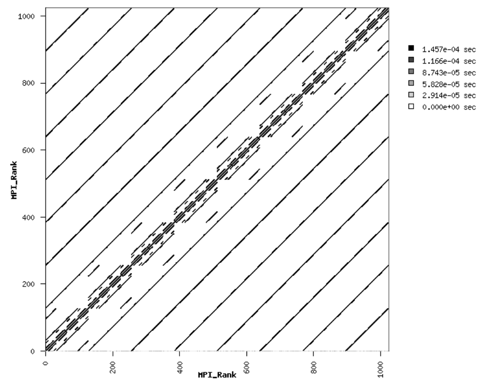
% of MPI Time



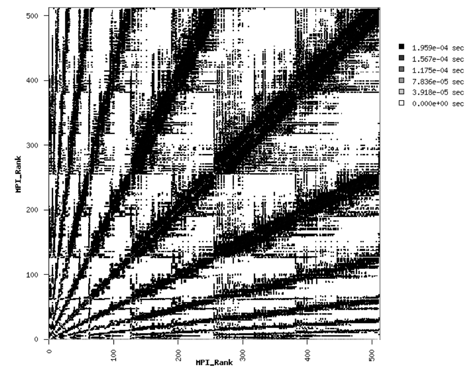
Communication Event Statistics (100.00% detail)

	Buffer Size	Ncalls	Total Time	Min Time	Max Time	%MPI	%Wall
MPI_Allreduce	8	3278848	124132.547	0.000	114.920	59.35	16.88
MPI_Comm_rank	0	35173439489	43439.102	0.000	41.961	20.77	5.91
MPI_Wait	98304	13221888	15710.953	0.000	3.586	7.51	2.14
MPI_Wait	196608	13221888	5331.236	0.000	5.716	2.55	0.72
MPI_Wait	589824	206848	5166.272	0.000	7.265	2.47	0.70

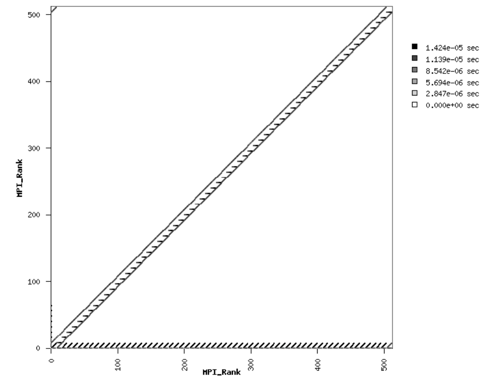
Not so simple: Comm. topology



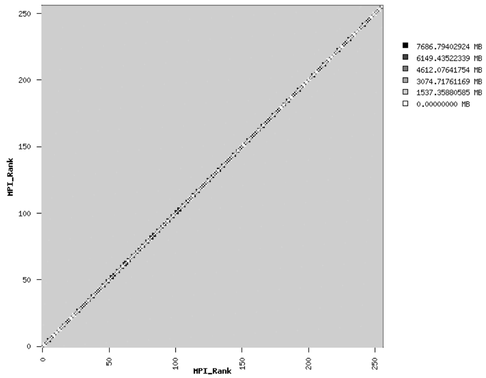
MILC



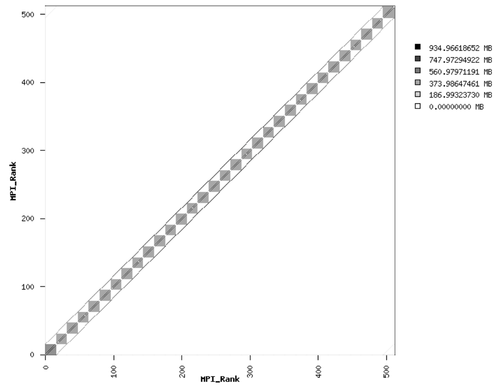
MAESTRO



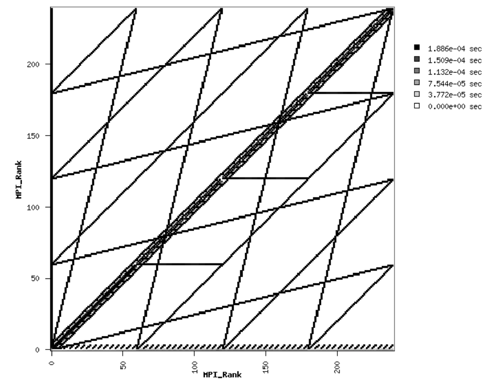
GTC



PARATEC



IMPACT-T



CAM

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

The state of HW counters

- The transition to many-core has brought complexity to the once orderly space of hardware performance counters. NERSC, UCB, and UTK are all working on improving things

- IPM on XE, currently just the banner is in place. We think PAPI is working (recently worked with Cray on bug fixes)

NERSC

Next up…Richard.