Application Performance Monitoring on Hopper: Integrated Performance Monitoring

David Skinner
deskinner@lbl.gov

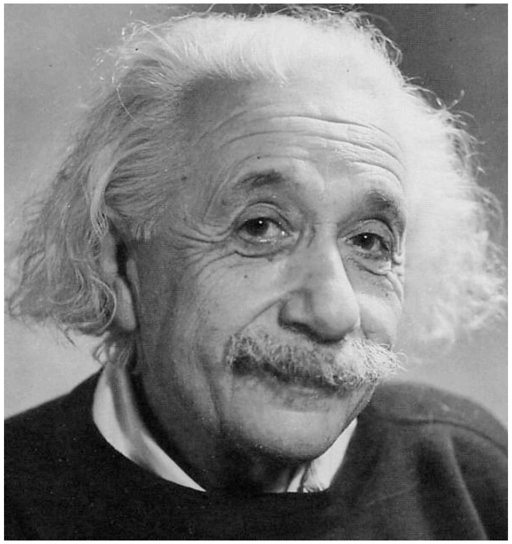
U.S. DEPARTMENT OF ENERGY | Office of Science
NERSC National Energy Research Scientific Computing Center
BERKELEY LAB Lawrence Berkeley National Laboratory

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, ERCAP etc.

Performance is Relative

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



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

In general the first bottleneck wins.
IPM can help find first order bottlenecks

What can IPM do?

- **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 (not an issue on XE knock wood)
- **What's going on overall in my code?**
  - How much comp, comm, I/O?
  - Where to start with optimization?
- **How is my load balance?**
  - Domain decomposition vs. concurrency (M work on N tasks)

How to use IPM : XE basics

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 ☺

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB Lawrence Berkeley National Laboratory

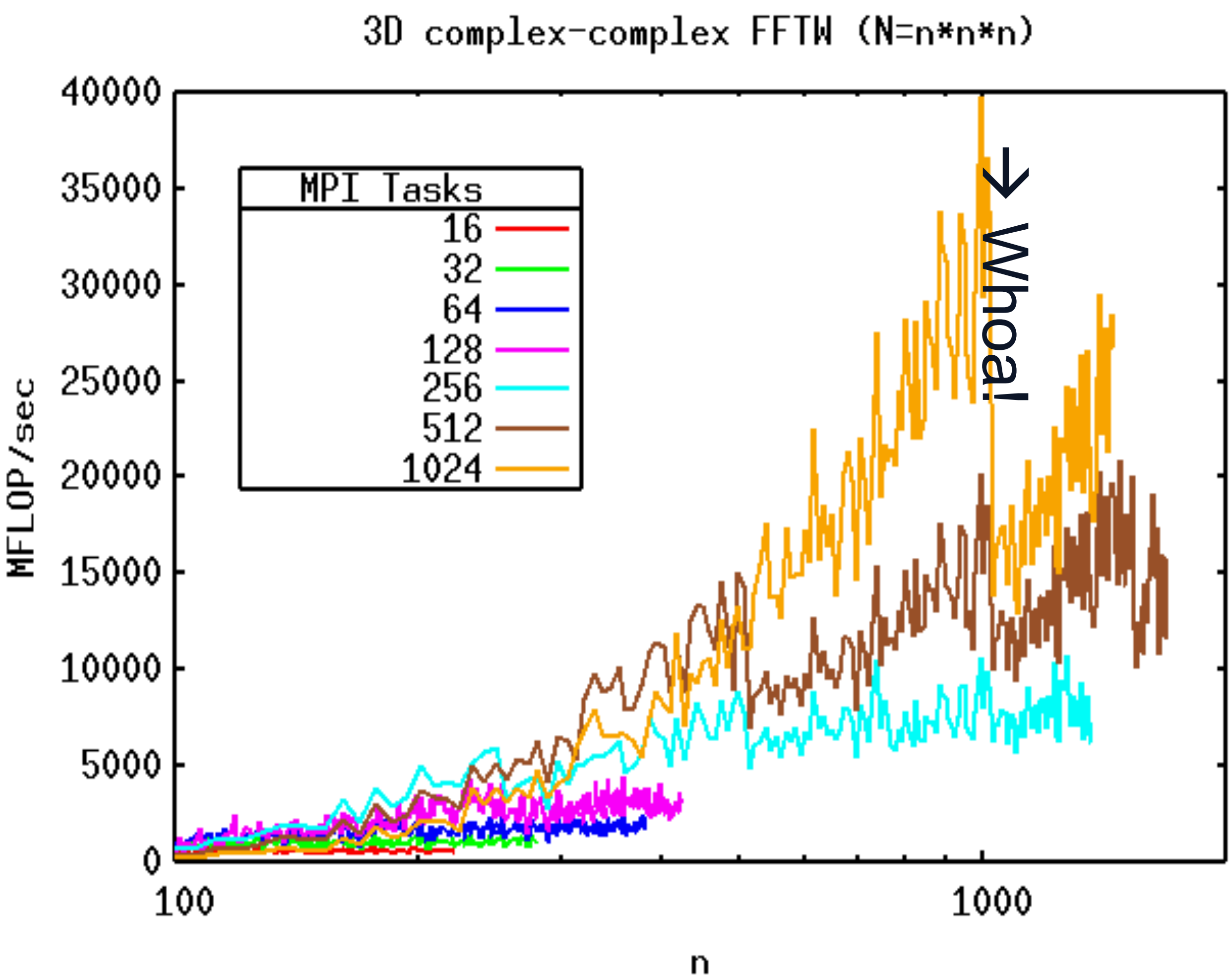
Generalities in Scalability and Performance

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

The scalability landscape

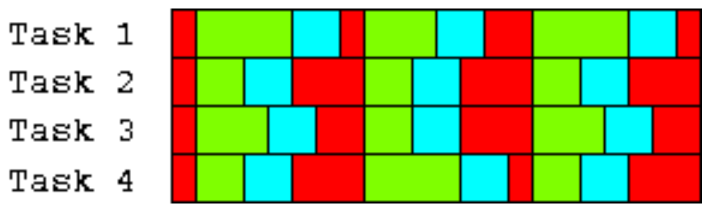


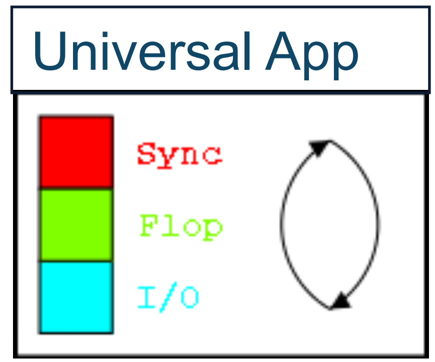
Why does efficiency drop?

- Serial code sections → Amdahl's law
- Surface to Volume → Communication bound
- Algorithm complexity or switching
- Communication protocol switching

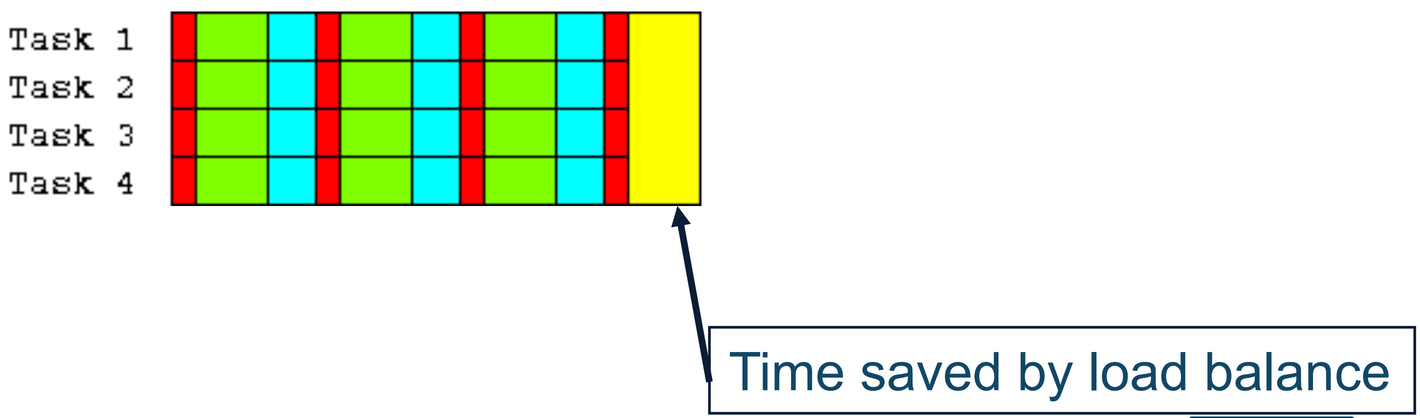
Load Balance : cartoon

Unbalanced:



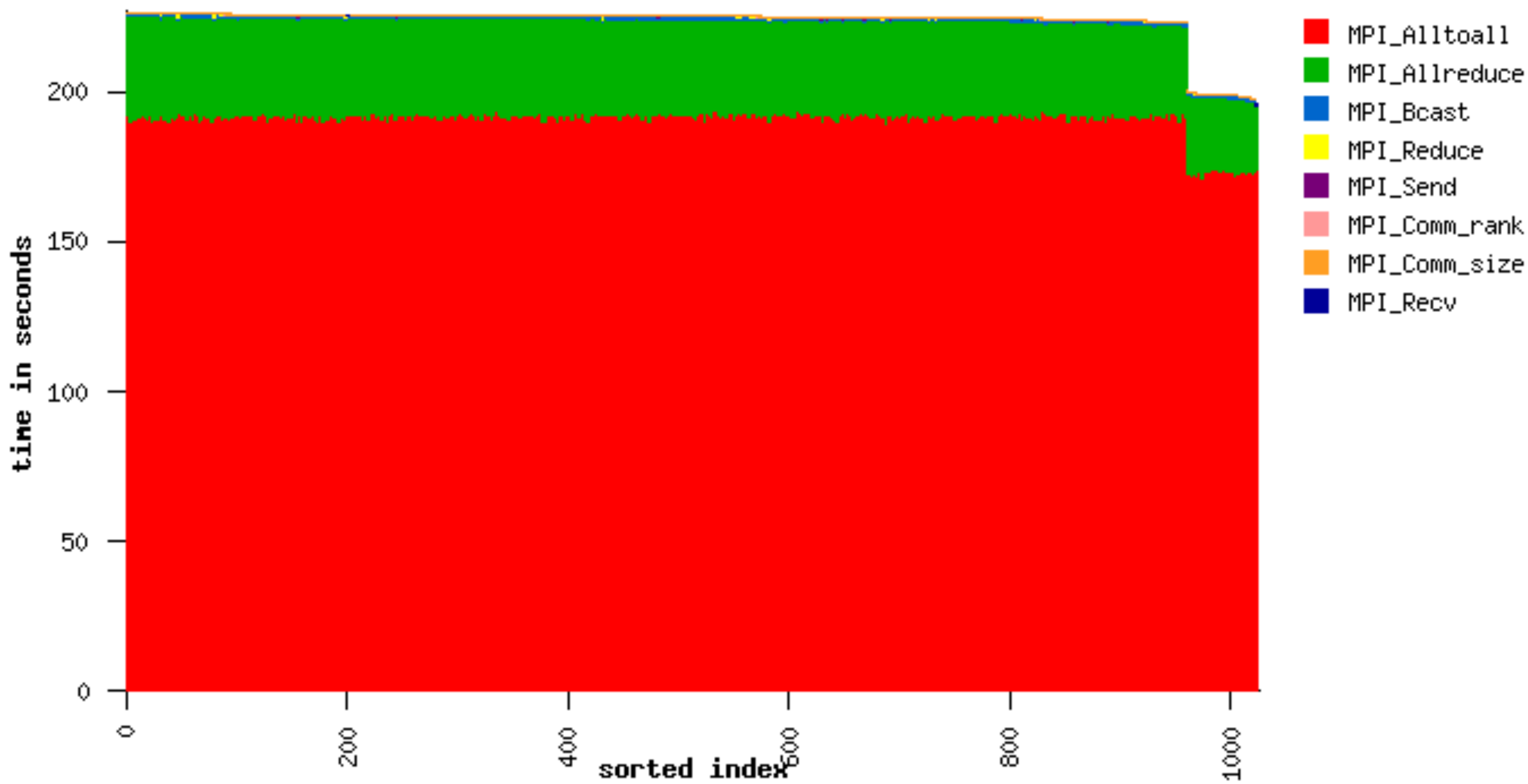


Balanced:



Load (Im)balance

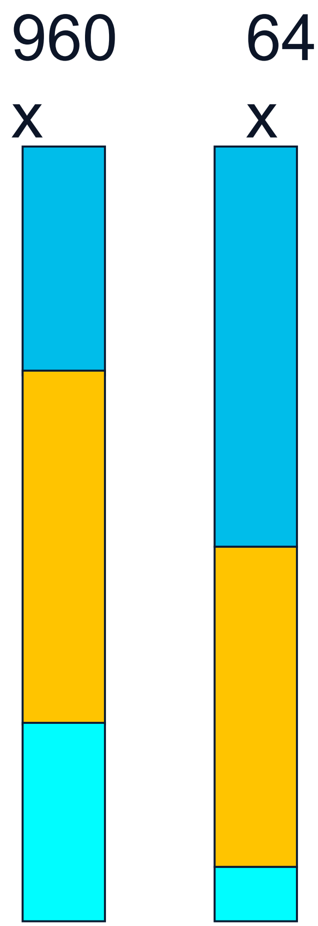
Communication Time: 64 tasks show 200s, 960 tasks show 230s



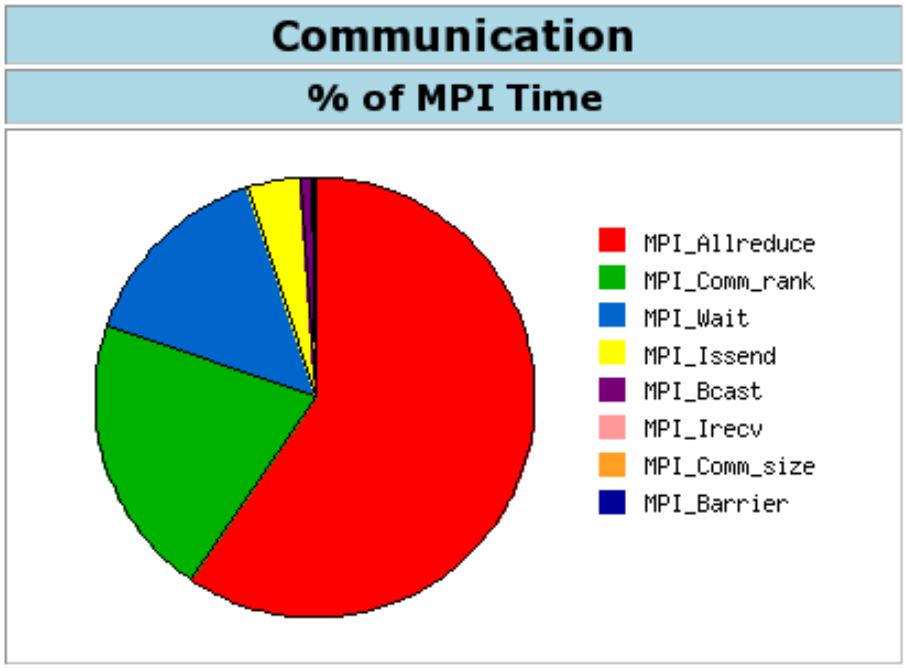
MPI ranks sorted by total communication time

Load Balance: ~code

```
while(1) {
 do_flops(Ni);
 MPI_Alltoall
   ();
 MPI_Allreduce
   ();
}
```



Simple Stuff: What's wrong here?



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

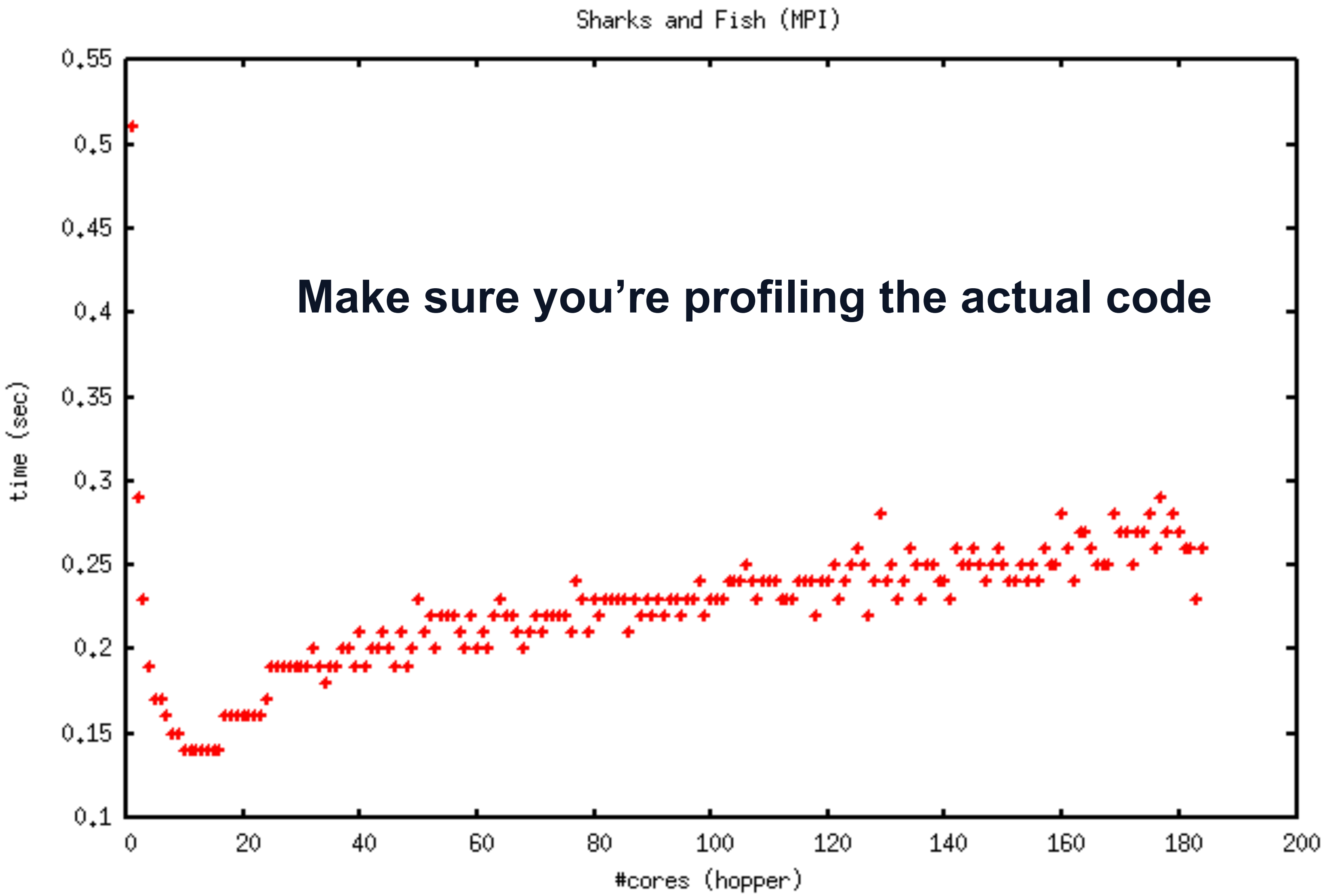
Some more specific examples

Example

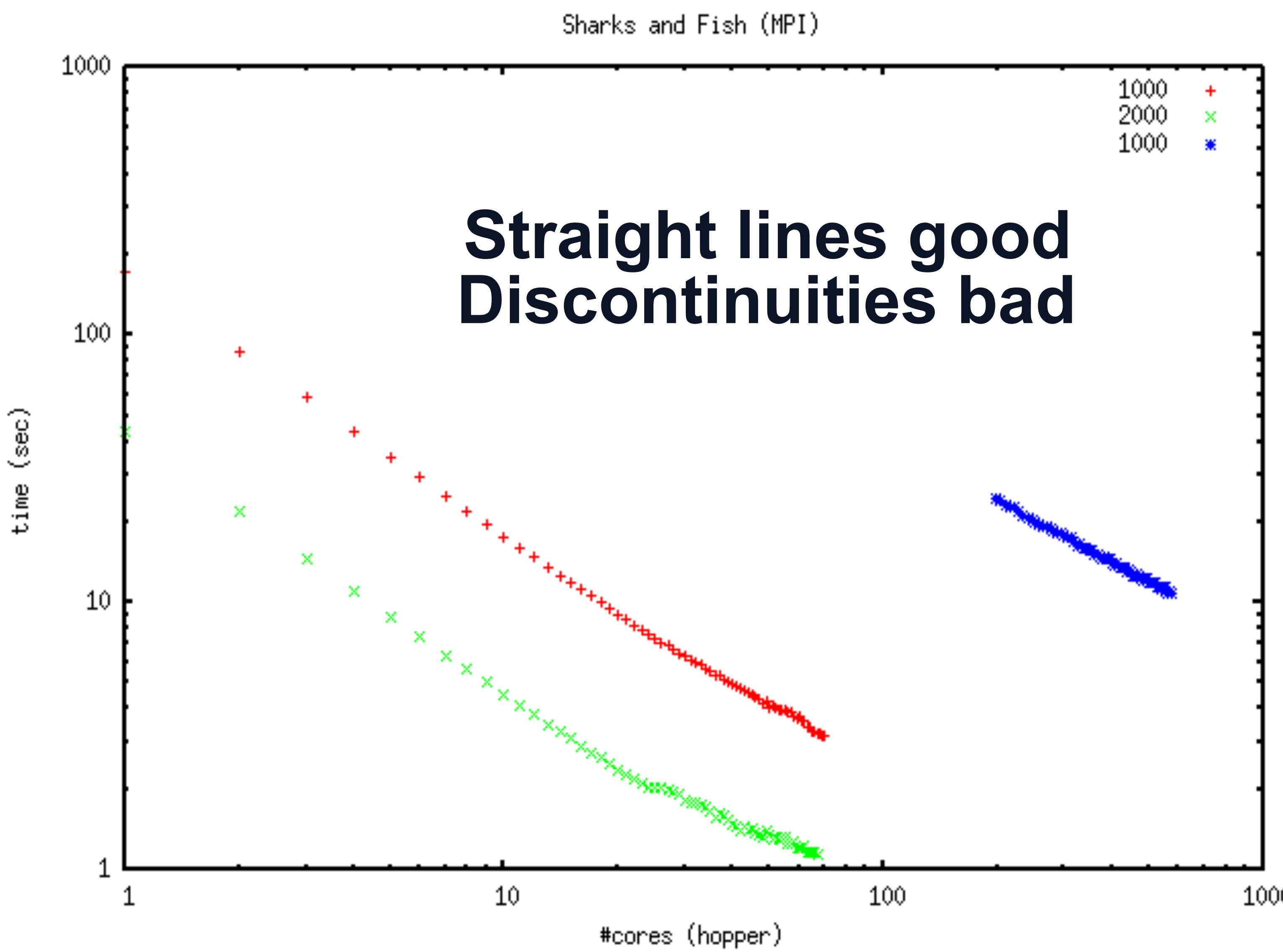
- **We can use your own code in the hands-on session**
- **In prep for that here is a worked example with the Sharks and Fish code**
  - A Newtonian particle pushing code w/ predator-prey dynamics between sharks and fish. Used in UCB CS267
  - See a glimpse here:

http://www.leinweb.com/snackbar/wator/

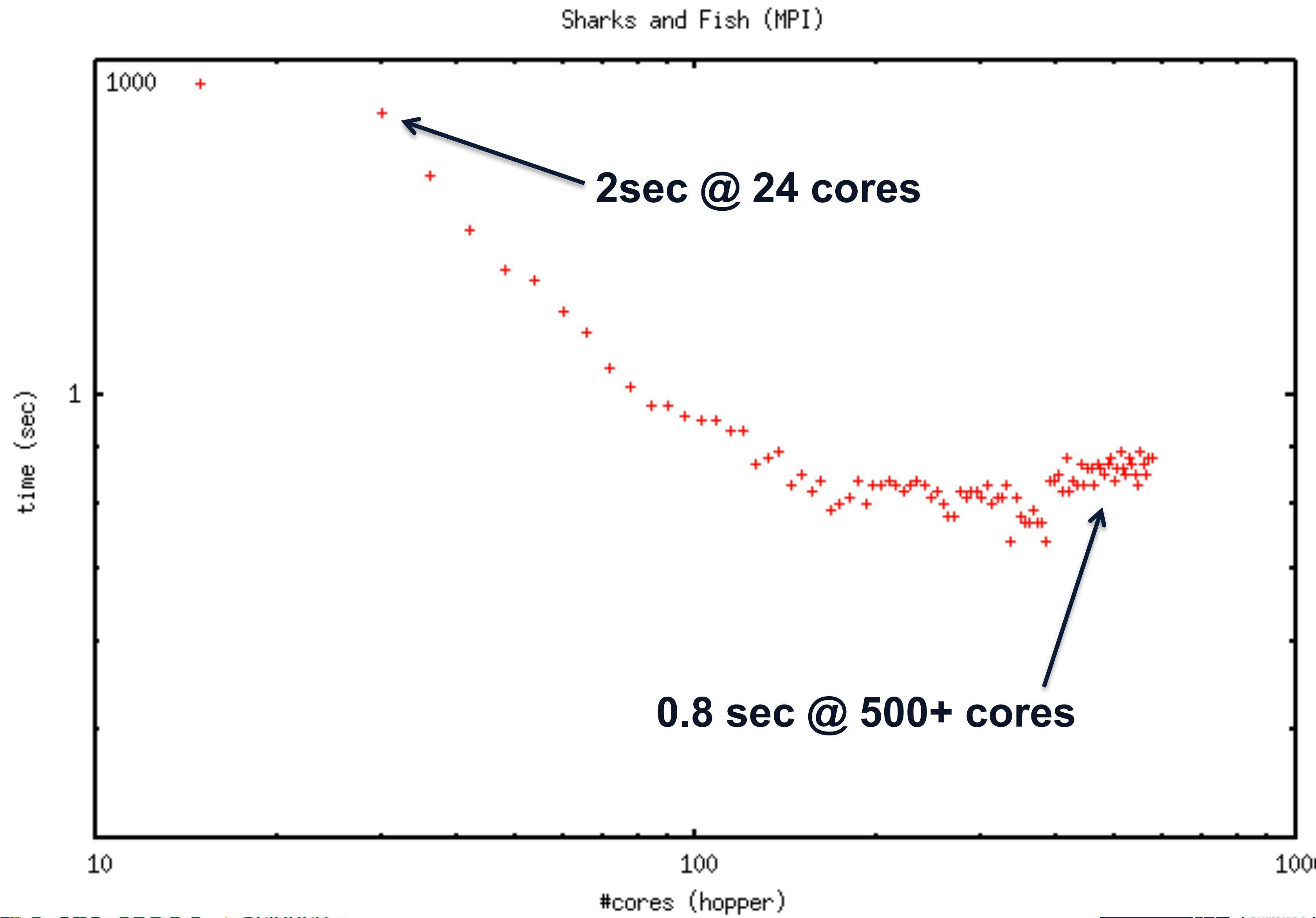
A scaling study w/ 100 Fish



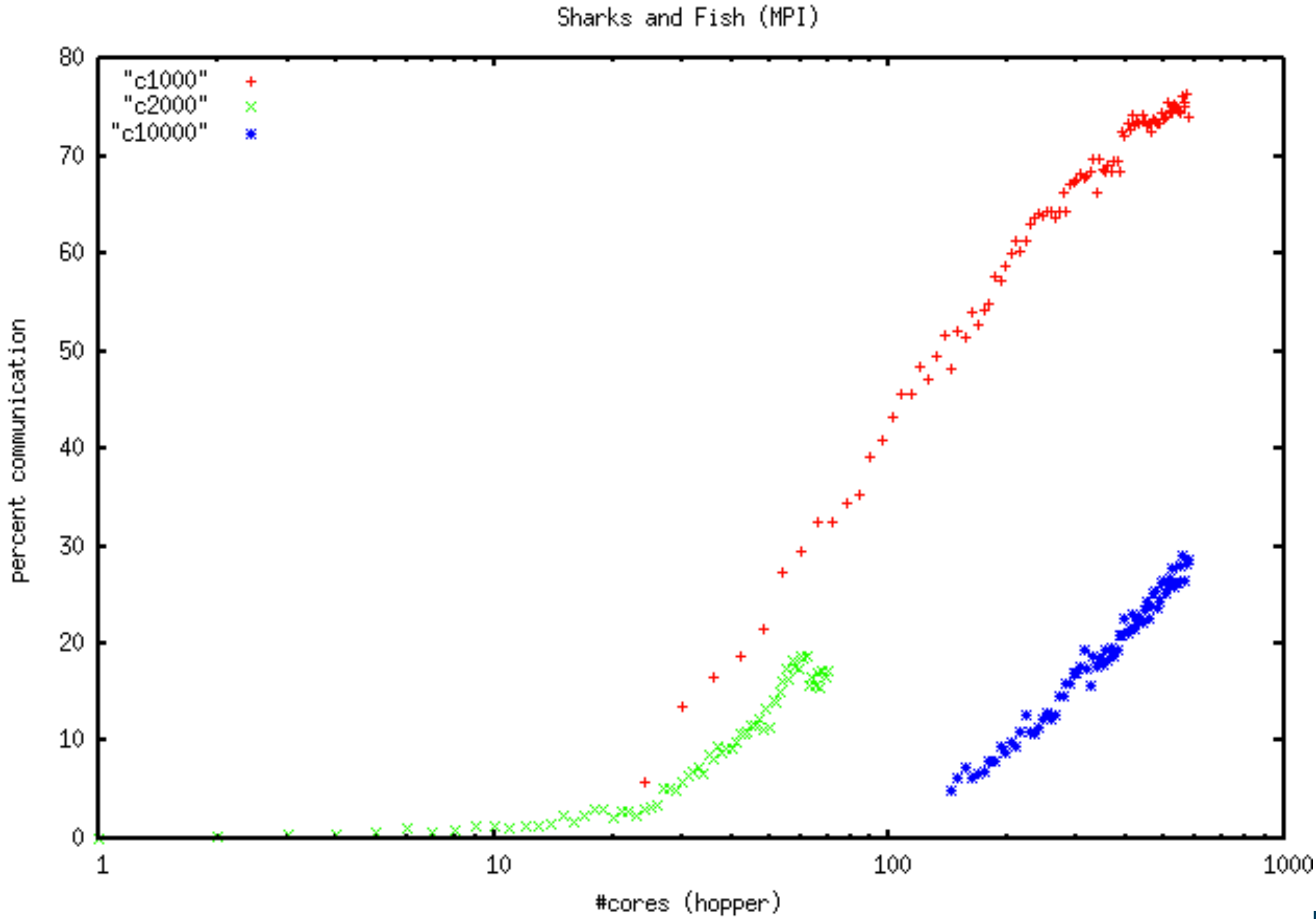
A real scaling study



Off the rails



Too much communication



Summary

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

...
```

We value your feedback on how to extend or improve IPM

help@nersc.gov

The state of HPM and IPM

- 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)

Thanks!

Questions about IPM?

deskinner@lbl.gov