

Assignment 3

Frequently Asked Questions

Big picture: Coding for scalability tests

`global_pi.c` ← `global_avg.c` + `pi.c` (fixed problem-size scaling)
`global_pi_iso.c` ← `global_pi.c` (isogranular scaling)
 a few lines change

Why does the measured runtime vary across different Slurm jobs?

Discovery cluster is composed of a **heterogeneous mixture of computing nodes with varying CPUs and GPUs**, hence different performance. This causes runtime variation across different Slurm jobs, depending on which nodes were allocated to the job. You can find the CPU information for one of the allocated nodes (on which your script is being executed) by including the following line in your Slurm script:

```
cat /proc/cpuinfo > cpuinfo.txt
```

```
cpuinfo.txt
```

```
model name : AMD EPYC 7513 32-Core Processor
```

```
...
```

Discovery Compute Nodes

<https://www.carc.usc.edu/user-guides/hpc-systems/discovery/resource-overview-discovery>

```
$ nodeinfo // The Slurm output reported SLURM_JOB_NODELIST = a01-[02-05]
```

Partition	Timelimit	CPU model	CPUs/ node	Memory(MB)/ node	GPU model	State	Nodes	Nodelist
debug	1:00:00	epyc-7313	32	256000	gpu:a40:2(S:0-1)	idle	1	b11-09
debug	1:00:00	xeon-4116	24	192000	(null)	idle	2	d05-[41-42]
debug	1:00:00	xeon-2640v4	20	128000	gpu:p100:2(S:0-1)	idle	1	e23-02
epyc-64	2-00:00:00	epyc-7513	64	256000	(null)	plnd	5	b09-[01-02,04,07-08]
epyc-64	2-00:00:00	epyc-7513	64	256000	(null)	drng	4	b09-17,b10-[02,07],b15-11
epyc-64	2-00:00:00	epyc-7513	64	256000	(null)	drain	2	b10-08,b15-17
epyc-64	2-00:00:00	epyc-7513	64	256000	(null)	mix	56	b04-[02-08,21],b05-[01-08,15-2
epyc-64	2-00:00:00	epyc-7513	64	256000	(null)	alloc	11	b04-[01,15-20],b09-[15,19],b10
main*	2-00:00:00	epyc-7542	64	256000	(null)	plnd	5	b22-[06-10]
main*	2-00:00:00	epyc-7513	64	256000	(null)	down*	4	a04-[16-19]
main*	2-00:00:00	epyc-7513	64	256000	(null)	comp	1	a01-14
main*	2-00:00:00	epyc-7513	64	256000	(null)	mix	31	a01-[02-05,11-13],a02-[11-14],
main*	2-00:00:00	epyc-7542	64	256000	(null)	mix	14	b22-[01-05,11-16,22-23,31]
main*	2-00:00:00	xeon-4116	24	94000+	(null)	mix	44	d05-[26-29,31-34,36,38-40],d06
main*	2-00:00:00	xeon-2640v4	20	63400+	(null)	mix	7	d17-[24-26,33-34,37],e16-05

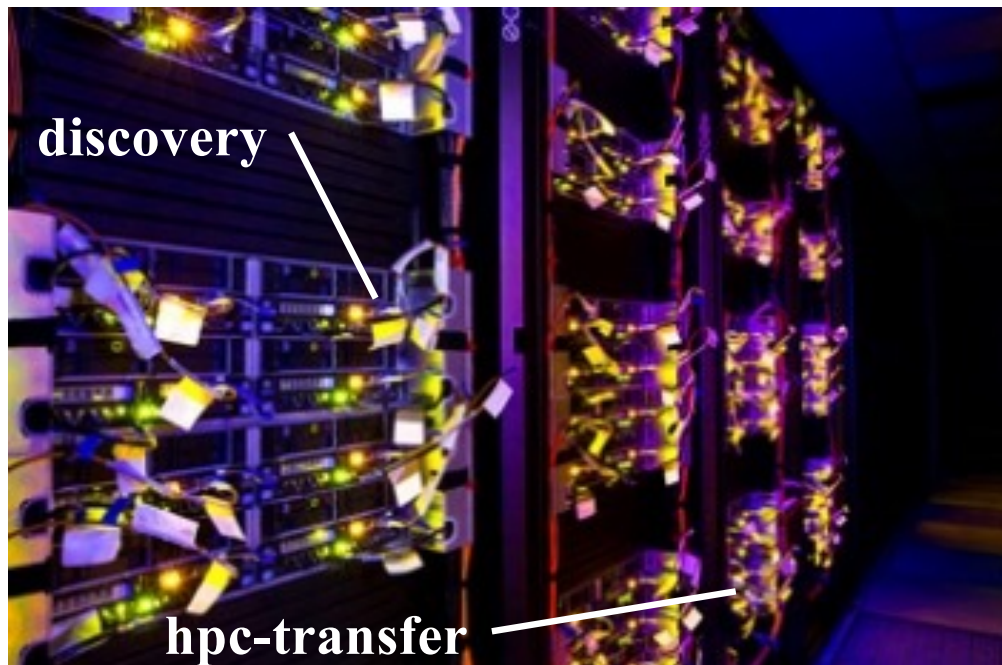
Also, optimization flag for compiler matters: `mpicc -O -o global_pi global_pi.c -lm`

Why is the measured runtime nonmonotonic as a function of the number of processors in some isogranular-scaling tests?

Even if you have dedicated access to the allocated computing nodes, you are still sharing network with other users. The communication time that `MPI_Send()` and `MPI_Recv()` take is thus affected by **network interference**. (Like your Internet speed slows down when someone at your home is downloading a big file.) Don't worry about small fluctuation in your plot. Or, submit multiple plots, with explanations.

```
[anakano@discovery ~]$ ping hpc-transfer
time=0.130 ms
time=0.090 ms
time=0.090 ms
time=0.113 ms
```

“See” network interference
cf. LA traffic

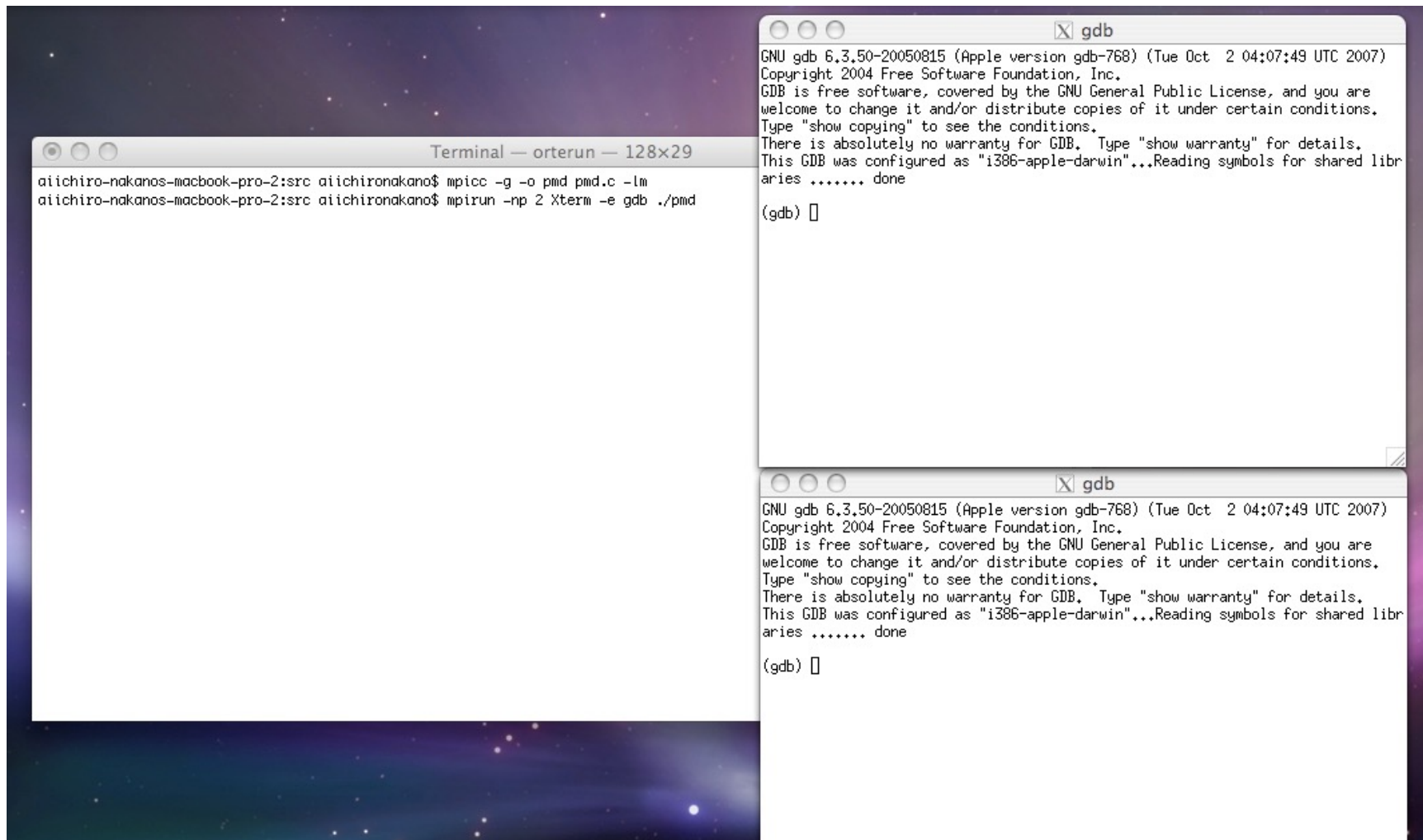


And you are sharing the nodes with other users.

How to debug MPI programs?

Different MPI ranks are different processes running on different computers, thus not executing in lockstep. This makes debugging MPI programs rather difficult. People usually insert **MPI_Barrier()** and **printf()** statements to locate the specific line where one or more ranks are crashing. Some systems allow MPI to work with debuggers like GDB, but I have not used them personally.

GNU C compiler-based MPI implementation (not on Discovery)



```
Terminal — orterun — 128x29
aiichiro-nakanos-macbook-pro-2:src aiichironakano$ mpicc -g -o pmd pmd.c -lm
aiichiro-nakanos-macbook-pro-2:src aiichironakano$ mpirun -np 2 Xterm -e gdb ./pmd

gdb
GNU gdb 6.3.50-20050815 (Apple version gdb-768) (Tue Oct 2 04:07:49 UTC 2007)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "i386-apple-darwin"...Reading symbols for shared libr
aries ..... done
(gdb) []

gdb
GNU gdb 6.3.50-20050815 (Apple version gdb-768) (Tue Oct 2 04:07:49 UTC 2007)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "i386-apple-darwin"...Reading symbols for shared libr
aries ..... done
(gdb) []
```

Finally: Please do not use “mpirun” on discovery

The login node, `discovery.usc.edu`, is shared by hundreds of users, and you are not supposed to run any serious programs on it. Please always use `sbatch` (in batch mode) or `salloc` (interactively) to run any MPI program, so that your program will run on dedicated computing nodes instead.

```
[anakano@discovery ~]$ ps -al
F S  UID      PID      PPID  C  PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  600118 133743   133453 3   80   0 - 524624 futex_ pts/6     00:14:32 viddy
0 S  354380 135459   135081 0   80   0 - 32268 poll_s pts/37    00:00:00 tmux: clie
1 S  354380 139661   3585060 0   80   0 - 28800 do_wai pts/57    00:00:00 bash
0 S  600493 626548   626491 0   80   0 - 83498 ep_pol pts/23    00:00:42 jupyter-no
0 S  323474 1154053 1133677 0   80   0 - 25299 do_wai pts/0     00:00:00 salloc
0 S  323474 1154202 1154053 0   80   0 - 83412 futex_ pts/0     00:00:00 srun
0 S  600773 1540702 1539154 0   80   0 - 28357 do_wai pts/25    00:00:00 bash
0 S  350473 1625067 1533720 0   80   0 - 39577 hrtime pts/13    00:00:05 watch
0 S  331977 1676488 1665045 0   80   0 - 32245 sys_pa pts/32    00:00:00 screen
0 S  352098 1680441 1661650 0   80   0 - 2082 n_tty_ pts/24    00:00:00 less
0 R   55322 1728179 1727860 0   80   0 - 38341 -      pts/46    00:00:00 ps
0 S  299827 2729297 2729150 0   80   0 - 37700 poll_s pts/19    00:00:00 vim
0 T  326739 2852294   60416 0   80   0 - 32348 do_sig pts/4     00:00:00 vim
1 S  354380 2885468   62782 0   80   0 - 28798 do_wai pts/14    00:00:00 bash
0 S  354380 2885544 2885468 0   80   0 - 395384 futex_ pts/14    00:00:00 srun
...
```

Also, don't wait till the last night to submit jobs!