

Assignment 3

Frequently Asked Questions

Big picture: Coding for scalability tests

global_pi.c \leftarrow **global_avg.c** + **pi.c** (fixed problem-size scaling)
global_pi_iso.c $\xleftarrow{\text{a few lines change}}$ **global_pi.c** (isogranular scaling)

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: Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz
```

...

Discovery Compute Nodes

<https://carc.usc.edu/user-information/user-guides/hpc-basics/discovery-resources>

Partition	CPU model	CPU frequency	CPUs/node	GPU model	GPUs/node	Memory/node	Nodes
main	xeon-2640v3	2.60 GHz	16	---	---	59 GB	81
main	xeon-2640v4	2.40 GHz	20	---	---	59 GB	51
main	xeon-4116	2.10 GHz	24	---	---	89 GB	39
main	xeon-4116	2.10 GHz	24	---	---	184 GB	41
main	xeon-2640v3	2.60 GHz	16	K40	2	59 GB	17
main	xeon-2640v4	2.40 GHz	20	K40	2	59 GB	40
epyc-64	epyc-7542	2.90 GHz	64	---	---	248 GB	32

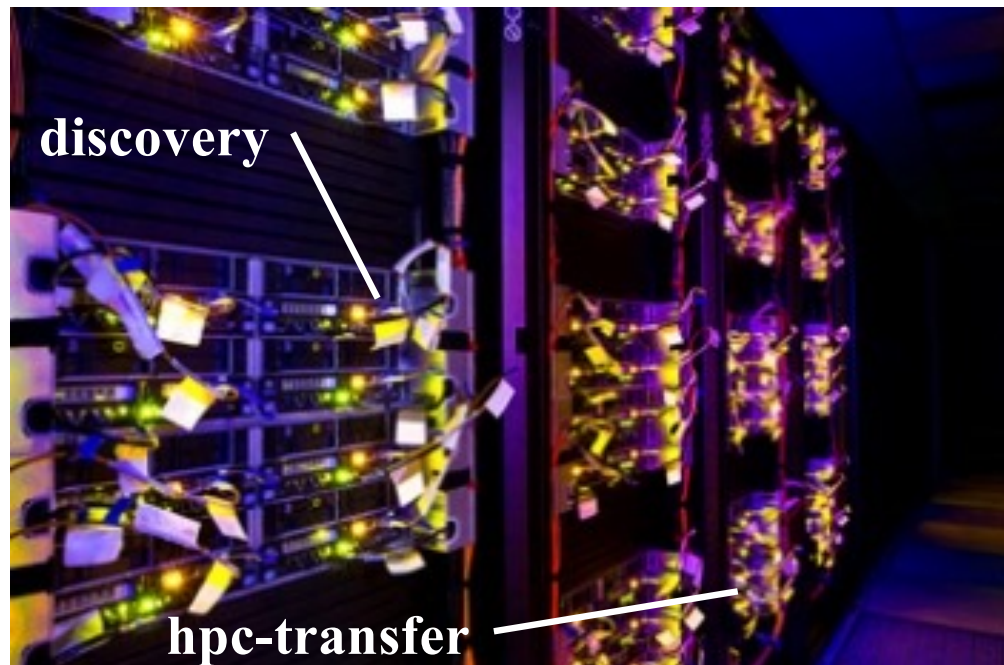
Also, optimization flag for compiler matters: `mpicc -O -o global_pi global_pi.c -lm`

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

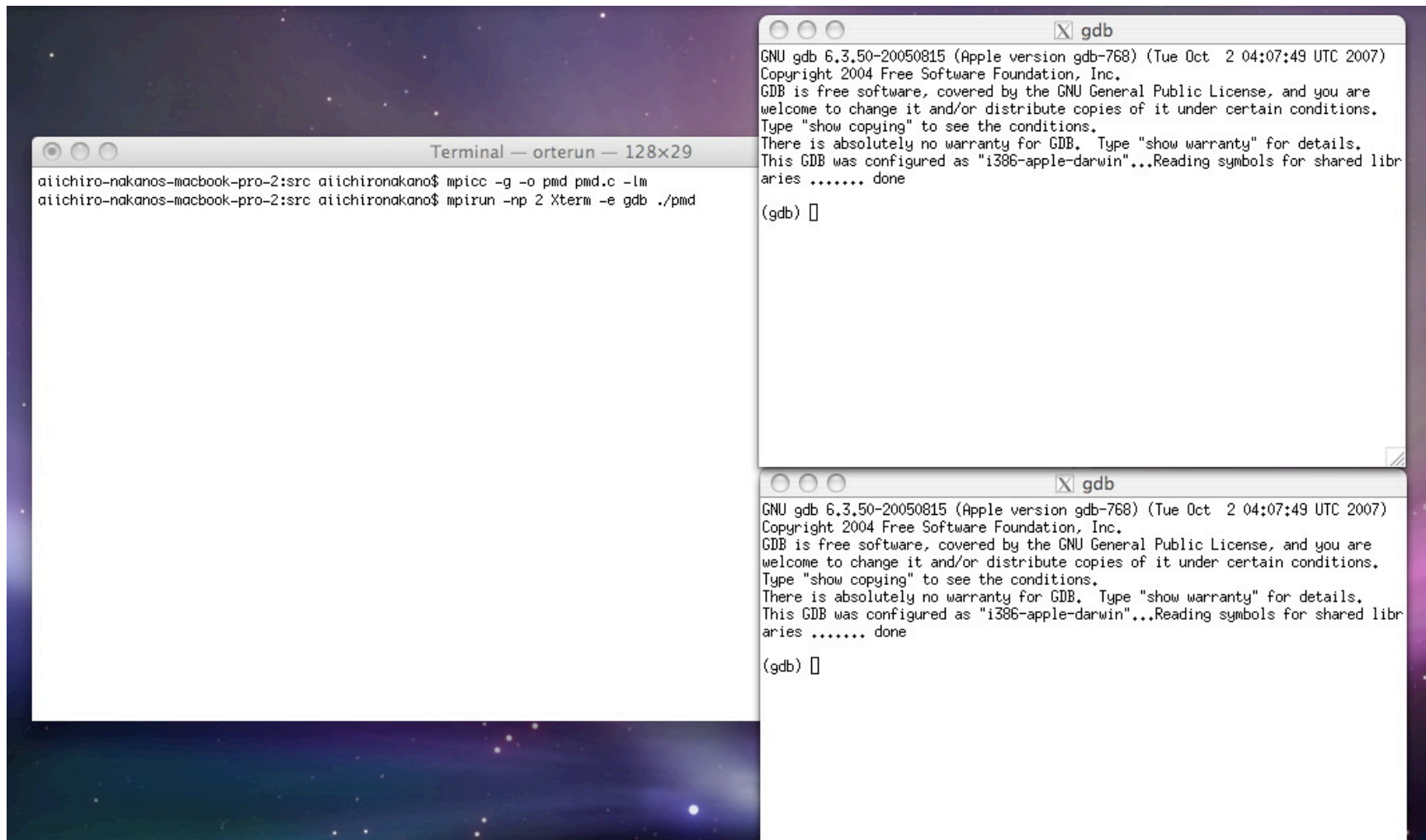


And you are sharing the nodes with other users.

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 a login node 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!