Assignment 3

CPSC424

Jake Brawer

March 6, 2017

1 Software and Development Environment

All the programming for this assignment was done in vim. This document was made using emacs. The only modules used in this assignment were Langs/Intel/15 and MPI/OpenMPI/1.8.6-intel15.

1.1 How to run the code

To compile the code and load the appropriate modules do the following:

```
cd HW3
sh setup.sh
```

In order to run code associated with a task, do the following:

```
qsub build-run_task<n>.sh
```

where $<n>$ is the number of the task (1-3). NOTE: there are three files associated with task2. This was done to run the task2 code using different numbers of nodes.

2 Task 1

Matrix multiplication times:

N	TIME (SECS)
1000	0.3491
2000	3.2685
4000	26.9946
8000	215.3641
12000	724.8860

3 Task 2

3.1 1 Node per processes

N	p	T_{comp}	T_{comm}	T_{total}
1000	1	.214	.0	.213
1000	2	.201	.076	0.144
1000	4	.199	.137	0.099
1000	8	.200	.061	0.061
2000	1	2.397	.0	2.404
2000	2	2.003	0.939	1.520
2000	4	1.547	1.008	0.742
2000	8	1.543	1.517	0.4347
4000	1	20.206	0.0	20.258
4000	2	19.819	7.605	13.896
4000	4	18.447	13.147	9.248
4000	8	15.224	19.199	4.7912
8000	1	158.729	.0	158.962
8000	2	158.718	58.611	109.588
8000	4	157.849	96.720	73.135
8000	8	154.884	146.747	41.0987

3.2 N Nodes per process

Nodes	N	p	T_{comp}	T_{comm}	T_{total}
2	4000	8	18.323	21.407	5.551
2	8000	8	179.748	164.217	47.025
4	4000	8	15.338	19.327	4.816
4	8000	8	1555.653	146.617	41.142

3.3 Analysis

The performance of the algorithm clearly does not scale well with N. From the table, there is generally an order of magnitude difference in T_{comp} with each new N. The larger matrices are likely too large to fit in caches, resulting in cache misses, which would explain latency. However, we see that as p goes up, walltime drops significantly even though the computation time stays the same, which is to be expected of increased parallelization.

Given the nature of the problem. Load balance was really poor. Here is a break down of times for each processes when $N = 8000$ and $p = 8$:

Rank	T_{comp}	T_{comm}
0	2.6035552	32.5018320
1	9.6408641	26.6162312
2	15.5497413	21.6846383
3	20.0316899	18.1150072
4	23.6048787	15.4343460
5	26.3032980	13.6382785
6	28.1652679	12.6867340
7	28.9846859	6.0699725

The higher ranks take drastically longer than the lower ranks. This is because the bottom rows of a triangular matrix have more elements than the top rows, and thus have to do more computations. One possible way to improve raw performance would be to balance the load better such that each processes gets a similar number of elements. That way, nodes would have minimal downtime between matrix operations and transferring new data.

4 Task 3

4.1 N: 4000, p:8

Rank	T_{comp}	T_{comm}
0	0.3064010	3.4325607
1	0.7865582	3.0880861
2	1.2177820	2.6955748
3	1.7257719	2.4751112
4	2.3202884	1.8734610
5	2.7701874	2.0287280
6	3.0012357	1.5136244
7	3.1219509	0.5708749

total comp 15.2502 total comm: 17.6780 total time: 4.8620

4.2 N: 8000, p:8

Rank	T_{comp}	T_{comm}
0	2.5950162	28.1467261
1	9.6281633	21.5909376
2	15.5542970	15.7984033
3	20.0510950	11.4315715
4	23.6090293	11.7515347
5	26.2921932	8.3918016
6	28.1252441	8.5823791
7	29.0012889	1.4017930

total comp 154.8563 total comm: 107.0951 total time: 36.9494

4.3 Analysis

Load balance wise nothing has changed here; The lower ranks still have relatively few elements to multiply and the higher ranks relatively many. whats changed here is that the procs are allowed to perform computations while sending a message, cutting down on the time spent idling. Indeed, this is reflected in the decreased T_{comm} time across all ranks. Additionally, this resulted in faster walltimes.

5 Task 4

5.1 N: 8000, p:8

Rank	T_{comp}	T_{comm}
0	24.5154459	2.3789113
1	23.3038557	6.5741057
2	21.3334873	10.7067559
3	19.7638998	14.6329527
4	18.4170363	13.9024374
5	17.2320039	13.1099050
6	16.1882646	12.6264982
7	15.1973917	13.4514012

total comp 155.9514 total comm: 87.3830 total time: 34.6173

5.2 Analysis

Here the load was balanced across processes meaning each processes received relatively the same number of elements. While I assumed improper load balancing was a major bottleneck on performance, load balancing had only modest gains. The total communication time decreased drastically, but total walltime only decreased by a few seconds compared to the non load balanced approach. This implies that the biggest bottleneck is probably memory related.

6 Task 5: N: 7663, p:7 Nodes: 4

Rank	T_{comp}	T_{comm}
0	0.0165167	0.0019746
1	0.0134375	0.0073438
2	0.0121093	0.0105138
3	0.0111079	0.0096681
4	0.0101681	0.0090859
5	0.0096390	0.0084102
6	0.0089161	0.0107658

total comp 0.0819 total comm: 0.0578 total time: 0.0254

7 Env

```
mpicc -g -O3 -I/home/fas/cpsc424/jnb37/scratch/jnb37_ps3_cpsc424/ -c task4.c
mpicc -o task4 -g -O3 -I/home/fas/cpsc424/jnb37/scratch/jnb37_ps3_cpsc424/ task4.o m
[jnb37@compute-33-1 HW3]$ vim build-run_task4
build-run_task4     build-run_task4.sh
[jnb37@compute-33-1 HW3]$ vim build-run_task4
build-run_task4     build-run_task4.sh
[jnb37@compute-33-1 HW3]$ vim build-run_task4.sh
[jnb37@compute-33-1 HW3]$ less Brawer_Task_5.o5425430
[jnb37@compute-33-1 HW3]$ less task
task2    task2.c  task2.o  task3    task3.c  task3.o  task4    task4.c  task4.o
[jnb37@compute-33-1 HW3]$ less Brawer_Task2.o5425401
[jnb37@compute-33-1 HW3]$ less Brawer_Task
Brawer_Task2_2_Nodes.o5425388  Brawer_Task_3.o5425403
Brawer_Task2_2_Nodes.o5425417  Brawer_Task_4.o5425429
Brawer_Task2_4_Nodes.o5425402  Brawer_Task_4.o5425440
```

```
Brawer_Task2.o5425401          Brawer_Task_5.o5425430
[jnb37@compute-33-1 HW3]$ less Brawer_Task
Brawer_Task2_2_Nodes.o5425388  Brawer_Task_3.o5425403
Brawer_Task2_2_Nodes.o5425417  Brawer_Task_4.o5425429
Brawer_Task2_4_Nodes.o5425402  Brawer_Task_4.o5425440
Brawer_Task2.o5425401          Brawer_Task_5.o5425430
[jnb37@compute-33-1 HW3]$ less Brawer_Task_3.o5425403
[jnb37@compute-33-1 HW3]$ less Brawer_Task_4.o54254
Brawer_Task_4.o5425429  Brawer_Task_4.o5425440
[jnb37@compute-33-1 HW3]$ less Brawer_Task_4.o54254
Brawer_Task_4.o5425429  Brawer_Task_4.o5425440
[jnb37@compute-33-1 HW3]$ less Brawer_Task_4.o5425429
[jnb37@compute-33-1 HW3]$ less Brawer_Task_4.o5425440
[jnb37@compute-33-1 HW3]$ less Brawer_Task_5.o5425430
[jnb37@compute-33-1 HW3]$ env
MKLROOT=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/mkl
MANPATH=/usr/local/cluster/hpc/MPI/OpenMPI/1.8.6-intel15/share/man:/home/apps/fas/Langs
GDB_HOST=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugger/gdb
/intel64_mic/bin/gdb-ia-mic
HOSTNAME=compute-33-1.local
IPPROOT=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/ipp
INTEL_LICENSE_FILE=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/licen
TERM=xterm
SHELL=/bin/bash
HISTSIZE=1000
GDBSERVER_MIC=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugger/g
SSH_CLIENT=10.191.63.252 56266 22
LIBRARY_PATH=/usr/local/cluster/hpc/MPI/OpenMPI/1.8.6-intel15/lib:/home/apps/fas/Langs/
PERL5LIB=/opt/moab/lib/perl5
FPATH=/usr/local/cluster/hpc/MPI/OpenMPI/1.8.6-intel15/include:/home/apps/fas/Langs/Int
QTDIR=/usr/lib64/qt-3.3
F90=ifort
PWD=/home/fas/cpsc424/jnb37/scratch/jnb37_ps3_cpsc424/HW3
_LMFILES_=/home/apps/fas/Modules/Base/yale_hpc:/home/apps/fas/Modules/Langs/Intel/15:/h
YHPC_COMPILER_MAJOR=2
JAVA_HOME=/usr/java/latest
GDB_CROSS=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugger/gdb/i
DOMAIN=omega
LANG=en_US.iso885915
MODULEPATH=/home/apps/fas/Modules
```

```
MOABHOMEDIR=/opt/moab
YHPC_COMPILER_RELEASE=2015
LOADEDMODULES=Base/yale_hpc:Langs/Intel/15:MPI/OpenMPI/1.8.6-intel15
KDEDIRS=/usr
F77=ifort
MPM_LAUNCHER=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugger/mp
CXX=icpc
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HISTCONTROL=ignoredups
INTEL_PYTHONHOME=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugge
SHLVL=1
HOME=/home/fas/cpsc424/jnb37
FC=ifort
LOGNAME=jnb37
QTLIB=/usr/lib64/qt-3.3/lib
CVS_RSH=ssh
SSH_CONNECTION=10.191.63.252 56266 10.191.12.33 22
MODULESHOME=/usr/share/Modules
LESSOPEN=||/usr/bin/lesspipe.sh %s
arch=intel64
INFOPATH=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/debugger/gdb/in
CC=icc
INCLUDE=/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/mkl/include
MPI_PATH=/usr/local/cluster/hpc/MPI/OpenMPI/1.8.6-intel15
G_BROKEN_FILENAMES=1
BASH_FUNC_module()=() {  eval `/usr/bin/modulecmd bash $*`
}
_=/bin/env
```