CPSC424 Parallel Programming

Lab3

Bo Song

Env

|  |
| --- |
| MKLROOT=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl  MANPATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/share/man:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/man/en\_US:/usr/share/man:/opt/moab/share/man:  MKL\_LINK=-L/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/intel64 -Wl,--start-group -lmkl\_intel\_lp64 -lmkl\_intel\_thread -lmkl\_core -Wl,--end-group -lpthread -lm  HOSTNAME=compute-14-1.local  IPPROOT=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp  INTEL\_LICENSE\_FILE=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/licenses:/opt/intel/licenses:/home/apps/fas/Licenses/intel\_site.lic  TERM=xterm  SHELL=/bin/bash  HISTSIZE=1000  SSH\_CLIENT=10.191.63.253 51916 22  LIBRARY\_PATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/lib:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/../compiler/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/compiler/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb/lib/intel64/gcc4.4  PERL5LIB=/opt/moab/lib/perl5  FPATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/include/intel64/lp64  QTDIR=/usr/lib64/qt-3.3  QTINC=/usr/lib64/qt-3.3/include  MIC\_LD\_LIBRARY\_PATH=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mpirt/lib/mic:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/compiler/lib/mic:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/mic:/opt/intel/mic/coi/device-linux-release/lib:/opt/intel/mic/myo/lib:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb/lib/mic  SSH\_TTY=/dev/pts/7  ANT\_HOME=/opt/rocks  USER=bs744  LD\_LIBRARY\_PATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/lib:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mpirt/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/../compiler/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/lib/intel64:/opt/intel/mic/coi/host-linux-release/lib:/opt/intel/mic/myo/lib:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/compiler/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb/lib/intel64/gcc4.4  MIC\_LIBRARY\_PATH=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb/lib/mic  ROCKS\_ROOT=/opt/rocks  CPATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb/include  YHPC\_COMPILER=Intel  OMPI\_MCA\_orte\_precondition\_transports=f20cd2d28f432704-15e3f8c3bb8e89d6  NLSPATH=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/compiler/lib/intel64/locale/%l\_%t/%N:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/ipp/lib/intel64/locale/%l\_%t/%N:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/intel64/locale/%l\_%t/%N:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gdb/intel64\_mic/py26/share/locale/%l\_%t/%N:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gdb/intel64/py26/share/locale/%l\_%t/%N:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/intel64/locale/%l\_%t/%N  MAIL=/var/spool/mail/bs744  PATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel/bin:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mpirt/bin/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gdb/intel64\_mic/py26/bin:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gdb/intel64/py26/bin:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/bin/intel64:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/bin/intel64\_mic:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gui/intel64:/home/apps/fas/Modules:/usr/lib64/qt-3.3/bin:/opt/moab/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/java/latest/bin:/opt/rocks/bin:/opt/rocks/sbin:/home/apps/bin:/home/fas/cpsc424/bs744/bin  YHPC\_COMPILER\_MINOR=2  TBBROOT=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/tbb  F90=ifort  PWD=/home/fas/cpsc424/bs744/cs424/BoSong\_PS3\_CPSC424  \_LMFILES\_=/home/apps/fas/Modules/Base/yale\_hpc:/home/apps/fas/Modules/Langs/Intel/14:/home/apps/fas/Modules/MPI/OpenMPI/1.6.5  YHPC\_COMPILER\_MAJOR=0  JAVA\_HOME=/usr/java/latest  IDB\_HOME=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/bin/intel64  GDB\_CROSS=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/debugger/gdb/intel64\_mic/py26/bin/gdb-mic  DOMAIN=omega  LANG=en\_US.iso885915  MODULEPATH=/home/apps/fas/Modules  MOABHOMEDIR=/opt/moab  YHPC\_COMPILER\_RELEASE=14  LOADEDMODULES=Base/yale\_hpc:Langs/Intel/14:MPI/OpenMPI/1.6.5  KDEDIRS=/usr  F77=ifort  CXX=icpc  SSH\_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass  HISTCONTROL=ignoredups  SHLVL=1  HOME=/home/fas/cpsc424/bs744  MKL\_LINK\_SEQUENTIAL=-L/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/lib/intel64 -Wl,--start-group -lmkl\_intel\_lp64 -lmkl\_sequential -lmkl\_core -Wl,--end-group -lpthread  FC=ifort  LOGNAME=bs744  QTLIB=/usr/lib64/qt-3.3/lib  CVS\_RSH=ssh  SSH\_CONNECTION=10.191.63.253 51916 10.191.10.209 22  MODULESHOME=/usr/share/Modules  LESSOPEN=||/usr/bin/lesspipe.sh %s  arch=intel64  CC=icc  DISPLAY=localhost:12.0  INCLUDE=/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/include:/home/apps/fas/Langs/Intel/icsxe/2013.1.046/composer\_xe\_2013\_sp1.2.144/mkl/include/intel64/lp64  MPI\_PATH=/home/apps/fas/MPI/OpenMPI/1.6.5-intel  G\_BROKEN\_FILENAMES=1  BASH\_FUNC\_module()=() { eval `/usr/bin/modulecmd bash $\*`  }  \_=/bin/env  OLDPWD=/home/fas/cpsc424/bs744 |

A make file is included in solution package.

**Task1 Serial program performance**

N TIME (secs)

----- -------------

1000 0.1797

2000 2.2810

4000 19.3756

8000 153.9208

12000 518.4189

**Task 2 Parallel program performance**

Raw performance table, p = 8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 node | | 2 nodes | | 4 nodes | |
| #Proc |  | N = 8000 | N=12000 | N = 8000 | N= 12000 | N=8000 | N=12000 |
| 0 | Comm time | 47.728039 | 168.069686 | 51.669500 | 139.064581 | 50.161624 | 133.829175 |
| Comp time | 6.981728 | 23.841349 | 3.781539 | 15.188161 | 2.404897 | 11.083201 |
| 1 | Comm time | 30.928616 | 114.894897 | 36.895063 | 99.661204 | 35.560373 | 96.545476 |
| Comp time | 18.261596 | 55.297461 | 10.245202 | 36.412500 | 9.167001 | 33.146939 |
| 2 | Comm time | 24.660388 | 85.390493 | 30.303248 | 78.393062 | 30.570016 | 61.168537 |
| Comp time | 26.266741 | 89.514275 | 18.113911 | 61.868680 | 15.172678 | 75.852079 |
| 3 | Comm time | 21.454271 | 84.306025 | 26.883242 | 66.318063 | 26.997465 | 44.599300 |
| Comp time | 30.268887 | 96.750693 | 22.730531 | 77.779222 | 19.668534 | 97.079598 |
| 4 | Comm time | 17.583461 | 56.590899 | 21.647295 | 55.586626 | 24.451651 | 62.652676 |
| Comp time | 35.948178 | 126.443870 | 29.806568 | 92.655340 | 23.132160 | 79.088234 |
| 5 | Comm time | 15.946033 | 47.610591 | 19.981328 | 47.045172 | 22.606396 | 53.896293 |
| Comp time | 38.721292 | 143.549835 | 32.386376 | 101.427900 | 25.926298 | 88.140114 |
| 6 | Comm time | 15.024184 | 66.171662 | 8.929359 | 47.225279 | 14.028019 | 50.843508 |
| Comp time | 39.668731 | 125.682471 | 46.499145 | 106.968397 | 38.516393 | 94.009908 |
| 7 | Comm time | 15.380558 | 50.892414 | 17.340718 | 43.223844 | 17.292582 | 48.203713 |
| Comp time | 39.329221 | 141.018594 | 38.110261 | 111.028759 | 35.273926 | 96.708664 |
| Total | Real time | 1m0.112s | 3m17.247s | 0m58.270s | 2m38.940s | 0m56.204s | 2m30.506s |

The table above shows the raw data of the performance under 3 configurations. Discussions are as below,

1. Raw performance – for the raw performance, it seems 1,2,4 nodes performs almost the same.
2. Load balance – Load balance is not good. Proc 0 always do the least computation work, while proc 7 always do the heaviest one. The computation time among different processes vary a lot.
3. Scalability – This algorithm is not scalable because the parallel efficiency decreases when p increases.

Suggestions,

1. Using no-blocking send/recv function– For most processes, a lot of time is used on communication parts. One reason is we are using blocking send/recv functions, which is safe but not efficient. Switching to non-blocking calls will improve the performance.
2. Allocate flexible number of rows – allocate more rows to one process when the number of elements in these rows is relatively small comparing to other rows. Therefore each process can compute approximately the same number of elements and achieve a load balance.

Conclusion from the set of results with p = 8:

Performance can’t be improved by allocating more nodes for the task. We should focus on optimizing algorithm.

**Task3 Non-blocking Communication**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 node | | 2 nodes | | 4 nodes | |
| #Proc |  | N = 8000 | N=12000 | N = 8000 | N= 12000 | N=8000 | N=12000 |
| 0 | Comm time | 40.147919 | 180.841691 | 34.780641 | 133.241363 | 31.903955 | 150.206143 |
| Comp time | 5.713917 | 21.163299 | 3.543888 | 14.932478 | 2.154690 | 10.852372 |
| 1 | Comm time | 20.680917 | 105.873109 | 22.103681 | 80.179828 | 19.610803 | 96.206326 |
| Comp time | 18.119205 | 59.589226 | 10.880871 | 50.288290 | 9.195730 | 36.843318 |
| 2 | Comm time | 14.253266 | 66.640101 | 17.204825 | 46.990116 | 13.579088 | 78.545088 |
| Comp time | 24.534594 | 98.866857 | 15.784186 | 83.458723 | 15.226347 | 54.695498 |
| 3 | Comm time | 5.602875 | 59.478495 | 12.263480 | 58.844335 | 9.291479 | 29.303046 |
| Comp time | 33.197228 | 106.031108 | 20.721139 | 71.759000 | 19.536430 | 103.993059 |
| 4 | Comm time | 7.468858 | 23.594143 | 5.853379 | 23.907193 | 6.650383 | 54.655600 |
| Comp time | 34.632653 | 156.685365 | 27.132669 | 106.705502 | 23.176314 | 78.700284 |
| 5 | Comm time | 5.918056 | 30.401165 | 7.611315 | 28.871405 | 8.239846 | 48.826690 |
| Comp time | 39.943796 | 171.603678 | 30.130560 | 113.998935 | 25.818788 | 87.590192 |
| 6 | Comm time | 2.166395 | 21.292271 | 5.716210 | 18.996820 | 2.555779 | 26.987083 |
| Comp time | 43.637253 | 164.828519 | 32.608326 | 129.176970 | 31.199644 | 134.071434 |
| 7 | Comm time | 0.194748 | 0.516925 | 0.218334 | 0.489364 | 0.227734 | 0.929506 |
| Comp time | 39.876985 | 170.502835 | 34.091956 | 133.305453 | 29.179112 | 138.361423 |
| Total | Real time | 0m49.768s | 3m28.185s | 0m42.134s | 2m34.305s | 0m37.564s | 2m46.786s |

Performance: No-blocking version performs better on small data set (N = 8000), but it doesn’t improve performance when N = 12000.

Scalability: It’s scalability is not strong. When node increases, the running time does not decreases much.

Load balance: the computation time varies a lot, process 0 runs much less time than process 7. It does not have load balance.

**Task4 Load balance**

Algorithm: Instead of assigning the same number of rows to each process, we assign nearly same number of elements in row vector to each process.

For example, p = 5, N = 10. Each process will be assigned to (1+10) \* 10 / 2 / 5 = 11 elements for its permanent row vector.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 node | | 2 nodes | | 4 nodes | |
| #Proc |  | N = 8000 | N=12000 | N = 8000 | N= 12000 | N=8000 | N=12000 |
| 0 | Comm time | 0.196519 | 0.428695 | 1.273652 | 0.611802 | 0.813960 | 2.729722 |
| Comp time | 35.814230 | 124.612697 | 25.425006 | 90.961243 | 21.090369 | 76.023772 |
| 1 | Comm time | 3.999923 | 19.809957 | 3.519772 | 11.014565 | 2.603908 | 12.813473 |
| Comp time | 32.010813 | 105.231392 | 23.156928 | 80.402356 | 19.230632 | 65.778477 |
| 2 | Comm time | 7.048572 | 22.460125 | 5.333823 | 18.124944 | 4.585132 | 19.663213 |
| Comp time | 28.944405 | 102.539182 | 21.335805 | 73.276381 | 17.241855 | 58.911472 |
| 3 | Comm time | 7.199054 | 35.103928 | 7.509963 | 23.483392 | 6.050212 | 24.905482 |
| Comp time | 28.801376 | 89.914006 | 19.145289 | 67.912339 | 15.796463 | 53.713203 |
| 4 | Comm time | 9.302175 | 36.233780 | 8.351240 | 34.204869 | 6.807402 | 10.926520 |
| Comp time | 26.695225 | 88.756054 | 18.306229 | 57.272002 | 15.054875 | 67.731194 |
| 5 | Comm time | 10.968989 | 45.122707 | 9.842251 | 39.877871 | 4.153613 | 17.048088 |
| Comp time | 25.026487 | 79.890252 | 16.830690 | 51.634965 | 17.750698 | 61.643371 |
| 6 | Comm time | 13.913002 | 55.073530 | 5.915458 | 43.846692 | 4.945262 | 36.938336 |
| Comp time | 22.070625 | 69.929332 | 20.770545 | 47.698672 | 16.930925 | 41.784970 |
| 7 | Comm time | 14.675121 | 52.340765 | 10.910636 | 45.047446 | 6.199962 | 39.665753 |
| Comp time | 21.307677 | 72.641300 | 15.788015 | 46.525616 | 15.689325 | 39.087747 |
| Total | Real time | 0m40.561s | 2m10.642s | 0m31.149s | 1m36.447s | 0m25.841s | 1m23.266s |

Row performance: The performance is best among these 3 tasks.

Scalability: As node increases, run time decreases a lot, It has some scalability but not that strong.

Load balance: Good load balance. All process spent nearly the same seconds.

**Task5 Generalization**

Padding N to p’s multiple, do task4 algorithm. Then extract useful part of the result.