Running LINPACK benchmarks on Linux on Power

Alexander Bokovoy  
Dr. Guanshan Tong  
IBM Linux Technology Center  
abokovoy@ru.ibm.com  
gtong@us.ibm.com
LINPACK and Top500

- Top 500 is a list of fastest computer systems in the world, updated twice a year
- LINPACK performance is used to do the ranking
- POWER is #1 performance architecture of Top500
Understanding Linpack HPC performance results

- **System description**
  - IBM eServer BladeCenter JS20+ (2-way 2.2GHz PowerPC970 with Myrinet)
  - IBM eSeries OpenPower 720 (2-way 1.6GHz POWER5 with Myrinet)

- $R_{\text{max}}$
  - Performance in Gflop/s for the largest problem run on the computer

- $N_{\text{max}}$
  - Problem size used to achieve $R_{\text{max}}$

- $N_{1/2}$
  - Problem size where half of the $R_{\text{max}}$ execution rate is achieved

- $R_{\text{peak}}$
  - Theoretical peak performance in Gflop/s for the machine
LINPACK typical result output

<table>
<thead>
<tr>
<th>T/V</th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
<th>Time</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>W03R2L4</td>
<td>200000</td>
<td>152</td>
<td>32</td>
<td>150</td>
<td>344.69</td>
<td>1.547e+04</td>
</tr>
</tbody>
</table>

||Ax-b||_oo / ( eps * ||A||_1 * N ) = 0.0121272 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1 * ||x||_1 ) = 0.0022087 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) = 0.0004098 ...... PASSED

<table>
<thead>
<tr>
<th>T/V</th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
<th>Time</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>W03R2L4</td>
<td>977816</td>
<td>152</td>
<td>32</td>
<td>150</td>
<td>22331.19</td>
<td>2.791e+04</td>
</tr>
</tbody>
</table>

||Ax-b||_oo / ( eps * ||A||_1 * N ) = 0.0014032 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_1 * ||x||_1 ) = 0.0008878 ...... PASSED
||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo ) = 0.0001331 ...... PASSED

\[ R_{\text{max}} = 27.91\text{TF}, \quad R_{\text{peak}} = 42.24\text{TF} \]
What is needed to run LINPACK on POWER?

- LINPACK HPL
- BLAS (Basic Linear Algebra Subprogram) or VSIPL (Vector Signal Image Processing Library)
- MPI (Message Passing Interface) library
- IBM Linux on POWER compilers
- GNU/Linux distribution (RHEL, SLES)
- Large pages support
Source code

- HPL, version 1.0a
  [http://www.netlib.org/benchmark/hpl](http://www.netlib.org/benchmark/hpl)
- BLAS ([http://www.netlib.org/blas/](http://www.netlib.org/blas/)):
  - IBM ESSL (binary-only)
  - Goto BLAS (binary-only)
    [http://www.tacc.utexas.edu/~kgoto/](http://www.tacc.utexas.edu/~kgoto/)
  - ATLAS
- Large pages support patch – more on this later
BLAS Libraries

• IBM ESSL and parallel ESSL
  http://www-03.ibm.com/systems/p/software/essl.html

• Implements very efficient DGEMM subroutine

• Allows for Myrinet-2000 communication using MPICH-GM for distributed DGEMM calls under Linux on POWER
BLAS Libraries (cont.)

- Goto BLAS
  http://www.tacc.utexas.edu/~kgoto/
  - Highly optimized BLAS implementation by Kazushige Goto of University of Texas, Austin
  - Available for POWER5 and PowerPC 970 (pSeries p5 systems, IBM BladeCenter JS20)
  - Available without charge to anyone for academic, research, experimental, or personal use
  - Contact Mr. Goto for special-kind versions – more on this later
Message Passing Interface

- MPI standard: http://www.mpi-forum.org/
- Some implementations for Linux on POWER:
  - MPICH – over Ethernet
  - MPICH-GM and MX – over Myrinet-2000
  - LAM/MPI and OpenMPI (Ethernet and Myrinet-2000)
  - IBM POE (experimental, in works)
- Good list of MPI implementations: http://www.lam-mpi.org/mpi/implementations/
Message Passing Interface (cont.)

- Pre-built MPI versions for Linux on POWER: http://ppclinux.ncsa.uiuc.edu/

- Simple self-contained LINPACK/MPI build environment will be available within IBM Redpaper “Running LINPACK benchmarks on 64-bit GNU/Linux” to be published December 2005
  http://www.redbooks.ibm.com
Large Pages

- A feature since 2.6 Linux kernel
- Documentation/vm/hugetlbpage.txt
- Linux on POWER supports 16Mb large pages
- Helps to minimize the size of page tables and TLB misses

- Substantially improves LINPACK performance on Linux on POWER, usually ~10% compared to 4Kb pages
Large pages setup

sysctl -w sys.vm.nr_hugepages=200

Before
$ cat /proc/meminfo
::
::
HugePages_Total: 0
HugePages_Free: 0
Hugepagesize: 16384 kB
::
::

After
$ cat /proc/meminfo
::
::
HugePages_Total: 200
HugePages_Free: 200
Hugepagesize: 16384 kB
::
::
Large pages (cont.)

● We aim for AIX-like setup:
  ● No need to additional system configuration
  ● Just link application with -blpdata
    NOT IMPLEMENTED YET!!

● Therefore:
  ● Modify LINPACK to allocate on large pages
  ● Modify BLAS library to use large pages
  ● Experimental Goto BLAS with large pages
  ● Experimental ESSL with large pages
Compiling and linking options

• For LINPACK HPL following options preffered when IBM compilers used on JS20:

```
CC           =mpicc -cc=xlc -q64
CCFLAGS      =$\text{\textdollar}(\text{HPL\_DEFS}) \ -O5 \ \ \\
              -qtune=ppc970 \ -qarch=ppc970 \ -DUSE\_LP
LINKER       =$\text{\textdollar}(\text{CC})
```

• For POWER5 change -qtune/-qarch to pwr5
LINPACK problem parameters

- Divide et impera: Proper system's division is a key to success
- How many MPI tasks?
- How many threads per each MPI task?

# MPI tasks × # Threads/task = # CPUs
LINPACK problem parameters (cont.)

- For example:

  8 2-way OpenPower P710 (8Gb RAM, Goto BLAS)
  
  8*2 = 16 CPUs (no SMT enabled)
  
  1 thread per task => 16 MPI tasks
  2 thread per task => 8 MPI tasks

- export GOTO_NUM_THREADS=1
- mpirun -np 16 -machinefile host.list ./xhpl
LINPACK problem parameters (cont.)

- Another key factor: process grid dimensions (P\times Q)

- P\times Q = \# CPUs

- P : Q = 1 : 4 usually gives better performance

- Therefore, better to use perfect square \#CPUs

\[
P = \sqrt{\frac{\text{number of CPUs}}{2}}
\]
LINPACK problem parameters (cont.)

- Problem size N depends on:
  - ... total memory available
  - ... number of large pages available
  - ... number of MPI tasks
  - ... interconnect library overhead
  - ... system I/O buffering

- General formula:
  memory size = N x N x 8 bytes
Problem size

- Common approaches:
  - The larger $N$, the better performance
  - Choose $N$ as large as possible
  - $N \times N \times 8 < \text{total memory size}$
  - Keep swapping below zero

$$N \times N \times 8 = 16 \times 8192 \text{ Mb} \Rightarrow N = 131072$$
Problem size and large pages

- When large pages are used, the amount of memory available as large pages is used to estimate N.

- How many large pages to allocate on each system?

hpc2:~ # cat /proc/meminfo
MemTotal: 7864320 kB
MemFree: 7023508 kB

HugePages_Total: 0
HugePages_Free: 0
Hugepagesize: 16384 kB

Total free memory = 6858 MB
Allocate 428 LPs = 428 x 16MB = 6848 MB
Problem size and large pages (cont.)

- Reserve large pages for Goto BLAS or ESSL – usually 10-20 large pages per node
- Reserve some memory for system I/O buffers and network driver buffers – up to 10 large pages per node
- Reserve some memory for Myrinet-2000 infrastructure – up to 10 large pages
- Usual reserve is about 10% of large pages in total

\[ N \times N \times 8 = \# \text{ nodes} \times 428 \times 0.9 \times 16384 \times 1024 \]

\[ N = 80390 \text{ (for 8 nodes)} \]
System configuration

- `/etc/sysctl.conf`:
  
  ```
  ...
  sys.vm.nr_hugepages=# Large Pages
  sys.vm.disable_cap_mlock=1
  kernel.shmmmax=NxNx8
  kernel.shmall=NxNx8
  kernel.vm.swappiness=10
  ...
  ```

- `sysctl -p`
LINPACK problem parameters

- Block size: **NB**
- Empirically selected:

<table>
<thead>
<tr>
<th></th>
<th>Best NB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goto for POWER5</td>
<td>256</td>
</tr>
<tr>
<td>Goto for JS20 Blade</td>
<td>256</td>
</tr>
<tr>
<td><strong>Goto for large JS20 cluster</strong></td>
<td>152</td>
</tr>
<tr>
<td>ESSL for POWER5</td>
<td>400</td>
</tr>
<tr>
<td>ESSL for JS20</td>
<td>200</td>
</tr>
</tbody>
</table>
LINPACK benchmark in brief

- Please read TUNING file in HPL source code distribution carefully
- Put appropriate N, P, Q, NB to HPL.dat
- Multiple combinations could be put, all to try in the same session
- Put xhpl, host.file, and HPL.dat on a shared disk
- Use mpirun to kick off the benchmark
- Collect results
LINPACK typical result output

<table>
<thead>
<tr>
<th>T/V</th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
<th>Time</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>W03R2L4</td>
<td>200000</td>
<td>152</td>
<td>32</td>
<td>150</td>
<td>344.69</td>
<td>1.547e+04</td>
</tr>
</tbody>
</table>

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{1} \cdot N)} = 0.0121272 \ldots \text{PASSED}
\]

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{1} \cdot \|x\|_{1})} = 0.0022087 \ldots \text{PASSED}
\]

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{\infty} \cdot \|x\|_{\infty})} = 0.0004098 \ldots \text{PASSED}
\]

<table>
<thead>
<tr>
<th>T/V</th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
<th>Time</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>W03R2L4</td>
<td>977816</td>
<td>152</td>
<td>32</td>
<td>150</td>
<td>22331.19</td>
<td>2.791e+04</td>
</tr>
</tbody>
</table>

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{1} \cdot N)} = 0.0014032 \ldots \text{PASSED}
\]

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{1} \cdot \|x\|_{1})} = 0.0008878 \ldots \text{PASSED}
\]

\[
\frac{\|Ax-b\|_{\infty}}{(\epsilon \cdot \|A\|_{\infty} \cdot \|x\|_{\infty})} = 0.0001331 \ldots \text{PASSED}
\]

\[R_{\text{max}} = 27.91\text{TF},\quad R_{\text{peak}} = 42.24\text{TF}\]
Thanks!

Linux Technology Center
IBM Corporation

http://www-1.ibm.com/linux/ltc/technology.shtml
http://www.ibm.com/ru/linuxcenter/

linux@ru.ibm.com