High Performance Linpack

Christian Külker

2023-01-25

Contents

1 HPL 2.3 On Raspberry Pi . . . 3
1.1 Preparation . . . 3
1.2 Package Openmpi & Openblas-pthread . . . 3
1.3 HPL 2.3 with Openmpi & Openblas-pthread . . . 3

2 HPL 2.3 Debian 11 on Raspberry Pi 4 8GB . . . 7
2.1 mpich-4.1a1 atlas-3.10.3 . . . 7

3 Compile Linpack 2.1 on CentOS 7 . . . 9
3.1 Dependencies . . . 9
3.2 Download . . . 10
3.3 Compile . . . 10

4 Intel Optimized Linpack For Linux / Clusters . . . 10
4.1 Device Output Codes . . . 10
4.2 Variables . . . 10

5 Example Linpack Configuration . . . 11

6 History . . . 11

The High Performance Linpack (HPL) is the major benchmark to measure the performance of super computers. As there are different hardware architectures, different optimized version of the benchmark exist.

Running HPL is at least a multi dimensional problem, where the main **6 dimensions** are:

1. Architecture
2. Message Passing Interface (MPI) Library
3. Linear Algebra (LA) Library

4. Compiler/ Linker
5. HPL
6. Benchmark Problem Size

Unfortunately, these 6 dimensions are only partly dependent on each other, some are not. For example, the problem size depends on the hardware size (nodes, memory, cores), but usually not on the HPL version. While the `xhpl` binary depends on the HPL version, MPI, LA and the compiler/linker, but not on the hardware memory size (unless optimized). The problem size configuration (HPL.dat) is not dependent on MPI and vice versa. Therefore, a strictly hierarchical directory structure to run HPL does not seem possible. The following diagram shows roughly the dependencies in round brackets and is the proposed directory structure of described linpack runs. This is the same as for the Raspberry Pi (rpi) (for educational purposes).

The following directory structure is used throughout this guide, with dependency information in round brackets and dimension information in angle brackets.

```
opt
  hpc
     src
       hpl-2.3.tar.gz
     bin
       run-create
     rpi [=arch] <dim0>
       mpi [=sw:mpi] <dim1>
         openipmi (arch) -> deb
         mpich (arch) <- tar
       la [=sw:la] <dim2>
         openblas-pthread (arch) -> deb
         atlas (arch) <- tar
       compiler/linker <dim3>
          c
          c++
          f77
       cfg <dim4>
         1n [=infrastructure[=inf]:node]
           8gb [=hw:mem]
             4c [=hw:core]
               1 (arch,inf,mem,core)
                 HPL.dat (arch,inf,mem,core)
       openmpi
         openblas-pthread
           hpl (arch,mpi,la(blas|openblas|atlas) [=sw:bm] <dim5>
             2.3 [=sw:bm:ver])
               bld
```

```
29                  Make.rpi (arch,mpi,la,compiler)
30                  bin/rpi/xhpl (arch,mpi,la,compiler,sw:bm:ver)
31                run
32                  1  (arch,mpi,la,compiler,inf,mem,core)
33                    run
34                    xhpl     (arch.mpi,la,compiler)
35                    HPL.dat  (arch,inf,mem,core)
36                    Make.rpi (arch,mpi,la,compiler)
```

1 HPL 2.3 On Raspberry Pi

1.1 Preparation

As root

```
mkdir -p /opt/hpc/hpl/src
mkdir -p /opt/hpc/hpl/2.3/openmpi/openblas-pthread/r/
chown -R $USER.$USER /opt/hpc/
```

As user `$USER`:

```
cd /opt/hpc/hpl/src
wget http://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz
```

1.2 Package Openmpi & Openblas-pthread

Installing `openmpi` and `openblas-pthread` from packages has the advantage of a fast installation. However, the result is usually not a very good performance. It is usually possible to get better performance with self-compiled components.

```
aptitude install openmpi-bin libopenblas-dev openmpi-common libopenmpi-dev
```

This will mostly install to `/usr/lib/aarch64-linux-gnu/`.

1.3 HPL 2.3 with Openmpi & Openblas-pthread

```
aptitude install automake
cd /opt/hpc/hpl/2.3/openmpi/openblas-pthread/
tar xvzf /opt/hpc/src/hpl-2.3.tar.gz
cd hpl-2.3/setup
sh make_generic
cp Make.UNKNOWN ../Make.rpi
```

Change the following values in /opt/hpc/hpl/2.3/openmpi/openblas-pthread/hpl-2.3/Make.rpi:

```
ARCH         = rpi

TOPdir       = /opt/hpc/hpl-2.3-openmpi-openblas

# /usr/lib/aarch64-linux-gnu/openmpi
MPdir        = /usr/lib/aarch64-linux-gnu/openmpi
# Search for include in openmpi
MPinc        = -I $(MPdir)/include
# Serach for libmpi.a or libmpi.so
# /usr/lib/aarch64-linux-gnu/openmpi/lib/libmpi.so ../../libmpi.so.40
MPlib        = $(MPdir)/lib/libmpi.so

LAdir        =
# 3. /usr/lib/aarch64-linux-gnu/libblas.a
LAinc        = /usr/include/aarch64-linux-gnu/
LAlib        = -lblas
# 2. /usr/lib/aarch64-linux-gnu/atlas/libblas.a
LAinc        = /usr/include/aarch64-linux-gnu/atlas
LAlib        = -lblas
# 1. /usr/lib/aarch64-linux-gnu/openblas-pthread/libblas.a
LAinc        = /usr/include/aarch64-linux-gnu/openblas-pthread
LAlib        = -lblas
```

```
make arch=rpi
...
make[1]: Leaving directory
 ↪  '/opt/hpc/hpl/2.3/openmpi/openblas-pthread/hpl-2.3'
```

Next, it is possible to run HPL on a single node. Prepare some variables and a `hostfile`. Run this manually or create a startup script. If the binary is not too large, it may also be advisable to copy the `xhpl` binary and the `Make.rpi` file to the run directory to rebuild or re-test the run on different hardware. An alternative is to make a note of the binary path and never recompile the archive under that directory …

This is an example of a simple startup script:

```
#!/usr/bin/zsh
export ARCH=rpi
export DIR=/opt/hpc/hpl/2.3/openmpi/openblas-pthread/
export BDIR=$DIR/hpl-2.3/bin/$ARCH
export BIN=$BDIR/xhpl
export RUN=0001
```

```
export RDIR=$DIR/r/$RUN
export MPI=/usr/bin/mpiexec
export HF=rpi4-8gb-1n-localhost.nodes
mkdir -p $RDIR
if [ -f $HF ];then rm $HF;fi
touch $HF
for i in {1..4}; do echo localhost >> $HF;done
cp $BIN .
cp $DIR/hpl-2.3/Make.rpi .
```

If you run the binary the output may look like this:

```
cd $RDIR
$MPI --hostfile $HF $BIN
================================================================================
HPLinpack 2.3  --  High-Performance Linpack benchmark  --   December 2,
 ↪  2018
Written by A. Petitet and R. Clint Whaley,  Innovative Computing
 ↪  Laboratory, UTK
Modified by Piotr Luszczek, Innovative Computing Laboratory, UTK
Modified by Julien Langou, University of Colorado Denver
================================================================================

An explanation of the input/output parameters follows:
T/V    : Wall time / encoded variant.
N      : The order of the coefficient matrix A.
NB     : The partitioning blocking factor.
P      : The number of process rows.
Q      : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N      :    5120
NB     :     128
PMAP   : Row-major process mapping
P      :       2
Q      :       2
PFACT  :   Right
NBMIN  :       4
NDIV   :       2
RFACT  :   Crout
```

```
BCAST  :  1ringM
DEPTH  :       1
SWAP   : Mix (threshold = 64)
L1     : transposed form
U      : transposed form
EQUIL  : yes
ALIGN  : 8 double precision words

----------------------------------------------------------------------------
 ↪  -------

- The matrix A is randomly generated for each test.
- The following scaled residual check will be computed:
      ||Ax-b||_oo / ( eps * ( || x ||_oo * || A ||_oo + || b ||_oo ) * N )
- The relative machine precision (eps) is taken to be
 ↪  1.110223e-16
- Computational tests pass if scaled residuals are less than
 ↪  16.0

============================================================================
T/V              N    NB     P     Q               Time                 Gflops
----------------------------------------------------------------------------
 ↪  -------
WR11C2R4        5120   128     2     2              26.71
 ↪  3.3515e+00
HPL_pdgesv() start time Sun Jun 19 23:53:01 2022

HPL_pdgesv() end time   Sun Jun 19 23:53:27 2022

----------------------------------------------------------------------------
 ↪  -------
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=   4.41990053e-03 ......
 ↪  PASSED
============================================================================

Finished      1 tests with the following results:
              1 tests completed and passed residual checks,
              0 tests completed and failed residual checks,
              0 tests skipped because of illegal input values.
----------------------------------------------------------------------------
 ↪  -------
```

```
End of Tests.
================================================================================
```

2 HPL 2.3 Debian 11 on Raspberry Pi 4 8GB

This describes an unsuccessful attempt to run `HPL` 2.3 as of December 2, 2018 with `mpich` (mpich-4.1a1) and `atlas` (atlas-3.10.3). In this example OpenMPI and OpenBlas were **not** used.

2.1 mpich-4.1a1 atlas-3.10.3

This use of HPL on a Raspberry Pi 4 8GB is for educational purposes. It is assumed that mpich-4.1a1 and atlas-3.10.3 have been successfully installed.

```
mkdir -p /tmp/hpc # should be already done
cd /tmp/hpc
wget http://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz
tar xvzf hpl-2.3.tar.gz
cd hpl2.3/setup
sh make_generic
cp Make.UNKNOWN ../Make.rpi
cd ..
```

On my Debian 11 Bullseye installation the following links had been made as a workaround:

```
cd /usr/lib/aarch64-linux-gnu
ln -s libevent_pthreads-2.1.so.7.0.1 libevent_pthreads.so
ln -s libevent_core-2.1.so.7.0.1 libevent_core.so
ln -s libhwloc.so.15.4.1 libhwloc.so
ln -s libopen-pal.so.40.30.0 libopen-pal.so
ln -s libopen-rte.so.40.30.0 libopen-rte.so
ln -s libmpi_mpifh.so.40.30.0 libmpi_mpifh.so
ln -s libmpi_usempi_ignore_tkr.so.40.30.0 libmpi_usempi_ignore_tkr.so
ln -s libmpi_usempif08.so.40.30.0 libmpi_usempif08.so
```

Some values of Make.rpi have to change.

```
ARCH         = rpi
TOPdir       = /tmp/hpc/hpl-2.3
MPdir        = /tmp/hpc
MPinc        = -I /tmp/hpc/include
MPlib        = /tmp/hpc/lib/libmpich.so
```

```
LAdir        = /tmp/hpc/atlas-build
LAlib        = $(LAdir)/lib/libf77blas.a $(LAdir)/lib/libatalas.a
```

Run `make arch=rpi`. If in some cases the configuration is not OK, make sure you are running from a **clean** archive.

```
make arch=rpi
```

Create a HPL.dat file.

```
HPLinpack benchmark input file
Innovative Computing Laboratory, University of Tennessee
HPL.out      output file name (if any)
6            device out (6=stdout,7=stderr,file)
1            # of problems sizes (N)
5120         Ns
1            # of NBs
128          NBs
0            PMAP process mapping (0=Row-,1=Column-major)
1            # of process grids (P x Q)
2            Ps
2            Qs
16.0         threshold
1            # of panel fact
2            PFACTs (0=left, 1=Crout, 2=Right)
1            # of recursive stopping criterium
4            NBMINs (>= 1)
1            # of panels in recursion
2            NDIVs
1            # of recursive panel fact.
1            RFACTs (0=left, 1=Crout, 2=Right)
1            # of broadcast
1            BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)
1            # of lookahead depth
1            DEPTHs (>=0)
2            SWAP (0=bin-exch,1=long,2=mix)
64           swapping threshold
0            L1 in (0=transposed,1=no-transposed) form
0            U  in (0=transposed,1=no-transposed) form
1            Equilibration (0=no,1=yes)
8            memory alignment in double (> 0)
```

Make a link.

```
cd
ln -s libmpi.so.40.30.0 libmpi.so.0
```

The following is a `mpiexec` demonstration of HPL running without a startup script.

```
mkdir /opt/hpl/hpl-run/rpi8gb
cp HPL.dat /opt/hpl/hpl-run/rpi8gb
cd /opt/hpl/hpl-run/rpi8gb

mpiexec --host localhost /tmp/hpc/hpl-2.3/bin/rpi/xhpl
--------------------------------------------------------------------------
 ↪  -
Primary job  terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
--------------------------------------------------------------------------
 ↪  -
--------------------------------------------------------------------------
 ↪  -
mpiexec noticed that process rank 0 with PID 0 on node c3 exited on signal
 ↪  11 (Segmentation fault).
--------------------------------------------------------------------------
 ↪  -
```

The following is a `mpirun` demonstration of HPL running without a startup script.

```
mpirun -np 4 /tmp/hpc/hpl-2.3/bin/rpi/xhpl
--------------------------------------------------------------------------
 ↪  -
Primary job  terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
--------------------------------------------------------------------------
 ↪  -
--------------------------------------------------------------------------
 ↪  -
mpirun noticed that process rank 2 with PID 0 on node c3 exited on signal
 ↪  11 (Segmentation fault).
--------------------------------------------------------------------------
 ↪  -
```

3 Compile Linpack 2.1 on CentOS 7

3.1 Dependencies

```
yum install wget
```

3.2 Download

```
wget http://www.netlib.org/benchmark/hpl/hpl-2.1.tar.gz
```

3.3 Compile

```
tar xf hpl-2.1.tar.gz
cd hpl-2.1/setup
sh make_generic
cp Make.UNKNOWN ../Make.Linux
cd ..
```

4 Intel Optimized Linpack For Linux / Clusters

Unfortunately, as of 2023, the performance of free open source (FOSS) libraries has limitations. Non-free HPLs for specific hardware, such as Intel CPUs, still solve the same problem as the free versions, are faster, but cannot be downloaded in source code. Theuse of non-free-HPLs or libraries will not be discussed in detail here, but some links are provided for further reading.

The Intel optimized HPL (Intel Linpack) requires the Intel(R) Math Kernel Library (MKL) 10.3 update 4 or later for Linux.

- MKL Download
- Intel Linkpack Download
- MKL
- oneMKL

4.1 Device Output Codes

```
6=stdout
7=stderr
```

4.2 Variables

Variables needs to match hardware. Example:

```
N      :   36992
NB     :     128
```

```
PMAP   : Row-major process mapping
P      :       1                   - compute nodes
Q      :      16                   - cores per node
```

5 Example Linpack Configuration

Some Linpack binaries accept configuration in the form of a file. With a name like `cfg.dat` that needs to have a special format, like this for HPL.

```
HPL.out      output file name (if any)
6            device out (6=stdout,7=stderr,file)
3            # of problems sizes (N)
82688 82880 82720 Ns
3            # of NBs
128 160 176 NBs
0            PMAP process mapping (0=Row-,1=Column-major)
1            # of process grids (P x Q)
4        Ps
4        Qs
16.0         threshold
1            # of panel fact
2            PFACTs (0=left, 1=Crout, 2=Right)
1            # of recursive stopping criterium
4            NBMINs (>= 1)
1            # of panels in recursion
2            NDIVs
1            # of recursive panel fact.
1            RFACTs (0=left, 1=Cr
```

6 History

Version	Date	Notes
0.1.2	2023-01-25	Improve writing, cleanup
0.1.1	2022-06-18	History, shell->bash, Debian 11, hpl-2.3, mpich-4.1a1 and atlas-3.10.3 on AMD and Rpi4
0.1.0	2020-05-03	Initial release