

HPL (High Performance Linpack)- Execution Guidelines for running applications in aggregated environment using ScaleMP's vSMP Foundation

Overview

HPL is a multi-process benchmark that uses MPI for inter-process communication. It is recommended to run HPL using MPICH2 tuned for vSMP Foundation in order to obtain best performance.

In addition, configuring the size (N) of the problem is highly recommended in order to achieve best performance.

It is possible to receive a packaged version of HPL from ScaleMP (with the original code, and with recommended build and run scripts).

Building HPL

Requirements

1. Intel compilers: If different than version 11.0.074, modify the script intel.sh
2. MPICH2 tuned for vSMP Foundation

It is possible to build and run the benchmark with gcc instead of the Intel compiler. However, the HPL benchmark runs faster when compiled with the Intel compiler.

To build with gcc just run: ./rebuild

To build with Intel compiler run: ./rebuild intel

Configuring HPL

The input files are named HPL.dat.<#cpus> and reside in the benchmark folder.

To change the size of the problem (See more details below), or other HPL modifiers, modify the input file(s).

To run with a different number of CPUs, create a new HPL.dat.<#cpus> and make sure that the multiple of 'P' & 'Q' in the new input file is the number of CPUs you desire. Modify the run script to add the new cpu count to the list.

Selecting the problem size

Problem size (N): Your problem size should be the largest to fit in the memory to get best performance. For e.g.: If you have 10 nodes with 1 GB RAM, total memory is 10GB. i.e. nearly 1342 M double precision elements. Square root of that number is 36635. You need to leave some memory for Operating System and other things. As a rule of thumb, 80% of the total memory will be a starting point for problem size (So, in this case, say, 33000). If the problem size is too large, it is swapped out, and the performance will degrade.

$$N = \text{SQRT}(\{\text{Available RAM in Bytes}\} / \text{sizeof(double)})$$

Example 1: A system with 32GB RAM / Board:

Physical RAM = 32GB -> Available with vSMP Foundation ~ 28GB -> $N = \text{SQRT}(28 \times 75\% \times 1024^3 / 8)$
= 53,000

->

For 8 Cores -> N=53,000

For 16 Cores -> $N = 53,000 \times \text{SQRT}(2)$ -> 75,000

For 32 Cores -> $N = 73,000 \times \text{SQRT}(2)$ -> 106,000

For 64 Cores -> $N=106,000 \times \text{SQRT}(2) \rightarrow 150,000$
Example 2: A system with 24GB RAM / Board:

Physical RAM = 24GB -> Available with vSMP Foundation ~ 21GB -> $N = \text{SQRT}(21 \times 75\% \times 1024^3 / 8)$
= 46,000 -> 46,000

->

For 8 Cores -> $N=46,000$

For 16 Cores -> $N=46,000 \times \text{SQRT}(2) \rightarrow 65,000$

For 32 Cores -> $N=65,000 \times \text{SQRT}(2) \rightarrow 92,000$

For 64 Cores -> $N=92,000 \times \text{SQRT}(2) \rightarrow 130,000$

Sample build script

```
#!/bin/sh

rm -rf hpl-2.0
tar xzfp hpl-2.0.tgz
if [[ "$1" -eq "_intel" ]] ; then
    . ./intel.sh
    cp -p Make.Linux_CBLAS.icc hpl-2.0/Make.Linux_CBLAS
fi
cd hpl-2.0
export BASE_DIR=`pwd`

make arch=Linux_CBLAS
# make arch=Linux_FBLAS
```

Sample run script

```
#!/bin/sh

. ./intel.sh

BASE=`pwd`
export PATH=${BASE}/hpl-2.0/bin/Linux_CBLAS:$PATH

export MPI_ROOT=/opt/ScaleMP/mpich2/1.0.8
export PATH=$MPI_ROOT/bin:$PATH
export VSMP_PLACEMENT=PACKED
export VSMP_VERBOSE=YES
export VSMP_MEM_PIN=YES

sudo su -c "echo 1 > /proc/sys/vm/overcommit_memory"

ulimit -s unlimited

tag=`date +%m%d-%H%M`

cd benchmark

for NPROC in 8 16 32 ; do
    echo "Running HPL with $NPROC CPUs"
    vsmputil --unpinall
    cp -p HPL.dat.$NPROC HPL.dat
    /usr/bin/time mpiexec -np $NPROC xhpl > output-$NPROC-$tag.txt 2>&1
    cp HPL.out HPL-$NPROC-$tag.out
done
```