Lecture 17: HPL

Software Management with Spack

- Yum installations are precompiled binaries compatible with the core compiler.
- They are usually well tested and stable (good for regular computing)
- The core compiler for a Linux distribution changes only with the Linux version
- Login to Wheeler, Xena, Hopper, and your server and report the core compiler versions for each: gcc --version

Software Management with Spack

- In High Performance Computing we usually want to compile our own code:

1. As we have seen when you compile code for the particular CPU architecture you are using you get performance improvements
2. We want the freedom to compile with different compilers. You saw that sometimes there is a difference in performance between Intel and GCC compilers.
3. We want to be able to compile code with the latest fastest compilers.
4. We might want to modify the source code for programs we deploy
5. We might want to modify the libraries programs use

Software Management with Spack

- Spack is distributed as a collection of build recopies for thousands of different software packages
- Community based – anyone can submit a new build package
- The packages are written in python so they are easy to modify

Clone the Spack Git Repo into Home Dir

```
[matthew@moonshine spack]$ git clone --depth=100 --
branch=releases/v0.21 https://github.com/spack/spack.git ~/spack
```

```
[matthew@moonshine spack]$ cd spack
```

```
[matthew@moonshine spack]$ . share/spack/setup-env.sh
```

These are the compilers to which spack has access

```
[matthew@moonshine spack]$ spack compilers
[matthew@moonshine spack]$ spack compiler find
==> Added 1 new compiler to
/home/matthew/.spack/linux/compilers.yaml
    gcc@11.4.1
==> Compilers are defined in the following files:
    /home/matthew/.spack/linux/compilers.yaml
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack list gcc
armpl-gcc  gcc  gccmakedep  gccxml
==> 4 packages
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack list gcc
armpl-gcc  gcc  gccmakedep  gccxml
==> 4 packages

[matthew@moonshine spack]$ spack info gcc

Description:
    The GNU Compiler Collection includes front ends for C, C++, Objective-C,
    Fortran, Ada, and Go, as well as libraries for these languages.

Homepage: https://gcc.gnu.org
Preferred version:
    13.2.0    https://ftpmirror.gnu.org/gcc/gcc-13.2.0/gcc-13.2.0.tar.xz
```

Take a look at the gcc package

```
[matthew@moonshine spack]$ spack edit gcc
```

One of the most useful things spack does is handle the many dependencies most HPC software needs.

This is done through simple conditional logic in python.

```
        "RelWithDebInfo: -O2 -g; MinSizeRel: -Os",
    )
    variant(
        "profiled",
        default=False,
        description="Use Profile Guided Optimization",
        when="+bootstrap %gcc",
    )

    depends_on("flex", type="build", when="@master")

    # https://gcc.gnu.org/install/prerequisites.html
    depends_on("gmp@4.3.2:")
    # mawk is not sufficient for go support
    depends_on("gawk@3.1.5:", type="build")
    depends_on("texinfo@4.7:", type="build")
    depends_on("libtool", type="build")
    # dependencies required for git versions
    depends_on("m4@1.4.6:", when="@master", type="build")
    depends_on("automake@1.15.1:", when="@master", type="build")
    depends_on("autoconf@2.69:", when="@master", type="build")

    depends_on("gmake@3.80:", type="build")
    depends_on("perl@5", type="build")

    # GCC 7.3 does not compile with newer releases on some platforms, see
    #   https://github.com/spack/spack/issues/6902#issuecomment-433030376
    depends_on("mpfr@2.4.2:3.1.6", when="@:9.9")
    depends_on("mpfr@3.1.0:", when="@10:")
    depends_on("mpc@1.0.1:", when="@4.5:")
    # Already released GCC versions do not support any newer version of ISL
    #   GCC 5.4 https://github.com/spack/spack/issues/6902#issuecomment-433072097
    #   GCC 7.3 https://github.com/spack/spack/issues/6902#issuecomment-433030376
    #   GCC 9+  https://gcc.gnu.org/bugzilla/show_bug.cgi?id=86724
                                                          172,1         13%
```

Spack generates a dependency tree

```
[matthew@moonshine spack]$ spack spec gcc
```

```
Concretized
--------------------------------
 -   gcc@13.2.0%gcc@12.3.0~binutils+bootstrap~graphite~nvptx~piclibs~profiled~strip build_system=autotools build_type=RelWithDebInfo languages=c,c++,f
andybridge
[+]      ^diffutils@3.9%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]          ^libiconv@1.17%gcc@12.3.0 build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
 -       ^gawk@5.2.2%gcc@12.3.0~nls build_system=autotools arch=linux-rocky9-sandybridge
[+]          ^libsigsegv@2.14%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]          ^readline@8.2%gcc@12.3.0 build_system=autotools patches=bbf97f1 arch=linux-rocky9-sandybridge
[+]      ^gmake@4.4.1%gcc@12.3.0~guile build_system=generic arch=linux-rocky9-sandybridge
 -       ^gmp@6.2.1%gcc@12.3.0+cxx build_system=autotools libs=shared,static patches=69ad2e2 arch=linux-rocky9-sandybridge
[+]          ^autoconf@2.69%gcc@12.3.0 build_system=autotools patches=35c4492,7793209,a49dd5b arch=linux-rocky9-sandybridge
[+]          ^automake@1.16.5%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]          ^m4@1.4.19%gcc@12.3.0+sigsegv build_system=autotools patches=9dc5fbd,bfdffa7 arch=linux-rocky9-sandybridge
[+]      ^libtool@2.4.7%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
 -       ^mpc@1.3.1%gcc@12.3.0 build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
 -       ^mpfr@4.2.0%gcc@12.3.0 build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
 -           ^autoconf-archive@2023.02.20%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]      ^perl@5.38.0%gcc@12.3.0+cpanm+opcode+open+shared+threads build_system=generic patches=714e4d1 arch=linux-rocky9-sandybridge
[+]          ^berkeley-db@18.1.40%gcc@12.3.0+cxx~docs+stl build_system=autotools patches=26090f4,b231fcc arch=linux-rocky9-sandybridge
[+]          ^bzip2@1.0.8%gcc@12.3.0~debug~pic+shared build_system=generic arch=linux-rocky9-sandybridge
[+]          ^gdbm@1.23%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
 -       ^texinfo@7.0.3%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]          ^gettext@0.22.3%gcc@12.3.0+bzip2+curses+git~libunistring+libxml2+pic+shared+tar+xz build_system=autotools arch=linux-rocky9-sandybridge
[+]              ^libxml2@2.10.3%gcc@12.3.0+pic~python+shared build_system=autotools arch=linux-rocky9-sandybridge
[+]              ^tar@1.34%gcc@12.3.0 build_system=autotools zip=pigz arch=linux-rocky9-sandybridge
[+]                  ^pigz@2.7%gcc@12.3.0 build_system=makefile arch=linux-rocky9-sandybridge
[+]              ^xz@5.4.1%gcc@12.3.0~pic build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
[+]          ^ncurses@6.4%gcc@12.3.0~symlinks+termlib abi=none build_system=autotools arch=linux-rocky9-sandybridge
[+]              ^pkgconf@1.9.5%gcc@12.3.0 build_system=autotools arch=linux-rocky9-sandybridge
[+]      ^zlib-ng@2.1.4%gcc@12.3.0+compat+opt build_system=autotools arch=linux-rocky9-sandybridge
[+]      ^zstd@1.5.5%gcc@12.3.0+programs build_system=makefile compression=none libs=shared,static arch=linux-rocky9-sandybridge
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack install gcc@13
/home/matthew/spack/opt/spack/linux-rocky9-sandybridge/

[matthew@moonshine spack]$ spack find gcc

[matthew@moonshine spack]$ spack load gcc@13

[matthew@moonshine spack]$ spack compiler find

[matthew@moonshine spack]$ spack compilers
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack find gcc
-- linux-rocky9-sandybridge / gcc@12.3.0 ------------------------
gcc@13.2.0

[matthew@moonshine spack]$ spack load gcc@13
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack compiler find
==> Added 1 new compiler to
/home/matthew/.spack/linux/compilers.yaml
    gcc@13.2.0
==> Compilers are defined in the following files:
    /home/matthew/.spack/linux/compilers.yaml
```

We are going to install a new compiler with Spack

```
[matthew@moonshine spack]$ spack compilers
==> Available compilers
-- gcc rocky9-x86_64 ----------------------
gcc@13.2.0  gcc@11.4.1
```

TOP 500
The List.

acm TURING AWARD SINCE 1966
CELEBRATING 50 YEARS
OF COMPUTING'S GREATEST ACHIEVEMENTS

Jack Dongarra
(Far left)

New Mexico

Clev Moler's PhD
Student

netlib.org

Basic Installation Tutorial — Sp... | Installing Lua and Lmod — Lmo... | Installing Lua and Lmod — Lmo... | Lmod download | SourceForge.... | Lmod: A New Environment Mo... | Getting Started — Spack 0.22.... | https://www.netlib.org/utk/peo...

The LINPACK Benchmark: Past, Present, and Future*

Jack J. Dongarra†, Piotr Luszczek†, and Antoine Petitet‡

July 2002

Abstract

This paper describes the LINPACK Benchmark [41] and some of its variations commonly used to assess performance of computer systems. Aside from the LINPACK benchmark suite, the TOP500 [43], and the HPL [48] code are presented. The latter is frequently used to obtained results for TOP500 submissions. Information is also given on how to interpret results of the benchmark and how the results fit into performance evaluation process.
Keywords: BLAS, benchmarking, high performance computing, HPL, linear algebra, LINPACK, TOP500

1 Introduction

The original LINPACK Benchmark is, in some sense, an accident. It was originally designed to assist users of the LINPACK package [15] by providing information on execution times required to solve a system of linear equations. The first "LINPACK Benchmark" report appeared as an appendix in the LINPACK Users' Guide [15] in 1979. The appendix comprised of data for one commonly used path in the LINPACK software package. Results were provided for a matrix problem of size 100, on a collection of widely used computers (23 computers in all). This was done so users could estimate the time required to solve their matrix problem by extrapolation.

TOP500 The List.

acm TURING AWARD SINCE 1966

CELEBRATING 50 YEARS OF COMPUTING'S GREATEST ACHIEVEMENTS

Present, and Future*

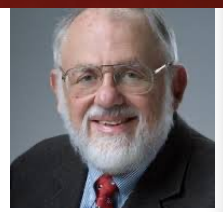
d Antoine Petitet‡

s variations commonly used to assess perfor-
ite, the TOP500 [43], and the HPL [48] code
500 submissions. Information is also given on
how to interpret results of the benchmark and how the results fit into performance evaluation process.
Keywords: BLAS, benchmarking, high performance computing, HPL, linear algebra, LINPACK, TOP500

1 Introduction

The original LINPACK Benchmark is, in some sense, an accident. It was originally designed to assist users of the LINPACK package [15] by providing information on execution times required to solve a system of linear equations. The first "LINPACK Benchmark" report appeared as an appendix in the LINPACK Users' Guide [15] in 1979. The appendix comprised of data for one commonly used path in the LINPACK software package. Results were provided for a matrix problem of size 100, on a collection of widely used computers (23 computers in all). This was done so

Jack Dong (Far left)



New Mexico

Clev Moler's PhD Student

Linear Algebra Libraries

- BLAS (Basic Linear Algebra Subprograms)
  - Provides basic operations (dot product, multiplication, etc)

  (Jack Donngara and Charles Lawson)
- LAPACK (Linear Algebra Package)
  - Higher level operations like solving systems of equations, eignevalues, decompositions, etc.
- ScaLAPACK (Scalable LAPack)
  - Uses MPI to make LAPack scale across nodes
  - There are other scalable implementations such as PBLAS
- HPL uses ScalLAPACK to benchmark supercomputers
- MKL is Intel’s optimized linear algebra package.

Installing the High Performance Linpack Benchmark

```
[matthew@moonshine spack]$ spack info hpl
AutotoolsPackage:   hpl

Description:
    HPL is a software package that solves a (random) dense linear system in
    double precision (64 bits) arithmetic on distributed-memory computers.
    It can thus be regarded as a portable as well as freely available
    implementation of the High Performance Computing Linpack Benchmark.

Homepage: https://www.netlib.org/benchmark/hpl/

Preferred version:
    2.3    https://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz
```

Installing the High Performance Linpack Benchmark

```
[matthew@moonshine spack]$ spack spec hpl %gcc@13
```

```
Concretized
--------------------------------
 -   hpl@2.3%gcc@13.2.0~openmp build_system=autotools arch=linux-rocky9-sandybridge
 -       ^gmake@4.4.1%gcc@13.2.0~guile build_system=generic arch=linux-rocky9-sandybridge
 -       ^openblas@0.3.24%gcc@13.2.0~bignuma~consistent_fpcsr+fortran~ilp64+locking+pic+shared build_system=makefile symbol_suffix=none threads=none arch=linux-rocky9-sandybrid
ge
 -           ^perl@5.38.0%gcc@13.2.0+cpanm+opcode+open+shared+threads build_system=generic patches=714e4d1 arch=linux-rocky9-sandybridge
 -               ^berkeley-db@18.1.40%gcc@13.2.0+cxx~docs+stl build_system=autotools patches=26090f4,b231fcc arch=linux-rocky9-sandybridge
 -               ^bzip2@1.0.8%gcc@13.2.0~debug~pic+shared build_system=generic arch=linux-rocky9-sandybridge
 -                   ^diffutils@3.9%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -               ^gdbm@1.23%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -                   ^readline@8.2%gcc@13.2.0 build_system=autotools patches=bbf97f1 arch=linux-rocky9-sandybridge
 -       ^openmpi@4.1.6%gcc@13.2.0~atomics~cuda~cxx~cxx_exceptions~gpfs~internal-hwloc~internal-pmix~java~legacylaunchers~lustre~memchecker~openshmem~orterunprefix+romio+rsh~si
ngularity+static+vt+wrapper-rpath build_system=autotools fabrics=none schedulers=none arch=linux-rocky9-sandybridge
 -           ^hwloc@2.9.1%gcc@13.2.0~cairo~cuda~gl~libudev+libxml2~netloc~nvml~oneapi-level-zero~opencl+pci~rocm build_system=autotools libs=shared,static arch=linux-rocky9-san
dybridge
 -               ^libpciaccess@0.17%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -                   ^util-macros@1.19.3%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -               ^libxml2@2.10.3%gcc@13.2.0+pic~python+shared build_system=autotools arch=linux-rocky9-sandybridge
 -                   ^libiconv@1.17%gcc@13.2.0 build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
 -                   ^xz@5.4.1%gcc@13.2.0~pic build_system=autotools libs=shared,static arch=linux-rocky9-sandybridge
 -               ^ncurses@6.4%gcc@13.2.0~symlinks+termlib abi=none build_system=autotools arch=linux-rocky9-sandybridge
 -           ^numactl@2.0.14%gcc@13.2.0 build_system=autotools patches=4e1d78c,62fc8a8,ff37630 arch=linux-rocky9-sandybridge
 -               ^autoconf@2.69%gcc@13.2.0 build_system=autotools patches=35c4492,7793209,a49dd5b arch=linux-rocky9-sandybridge
 -               ^automake@1.16.5%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -               ^libtool@2.4.7%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -               ^m4@1.4.19%gcc@13.2.0+sigsegv build_system=autotools patches=9dc5fbd,bfdffa7 arch=linux-rocky9-sandybridge
 -                   ^libsigsegv@2.14%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -           ^openssh@9.5p1%gcc@13.2.0+gssapi build_system=autotools arch=linux-rocky9-sandybridge
 -               ^krb5@1.20.1%gcc@13.2.0+shared build_system=autotools arch=linux-rocky9-sandybridge
 -                   ^bison@3.8.2%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -                   ^findutils@4.9.0%gcc@13.2.0 build_system=autotools patches=440b954 arch=linux-rocky9-sandybridge
 -                   ^gettext@0.22.3%gcc@13.2.0+bzip2+curses+git~libunistring+libxml2+pic+shared+tar+xz build_system=autotools arch=linux-rocky9-sandybridge
 -                       ^tar@1.34%gcc@13.2.0 build_system=autotools zip=pigz arch=linux-rocky9-sandybridge
 -                           ^pigz@2.7%gcc@13.2.0 build_system=makefile arch=linux-rocky9-sandybridge
 -                           ^zstd@1.5.5%gcc@13.2.0+programs build_system=makefile compression=none libs=shared,static arch=linux-rocky9-sandybridge
 -               ^libedit@3.1-20210216%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -               ^libxcrypt@4.4.35%gcc@13.2.0~obsolete_api build_system=autotools patches=4885da3 arch=linux-rocky9-sandybridge
 -               ^openssl@3.1.3%gcc@13.2.0~docs+shared build_system=generic certs=mozilla arch=linux-rocky9-sandybridge
 -                   ^ca-certificates-mozilla@2023-05-30%gcc@13.2.0 build_system=generic arch=linux-rocky9-sandybridge
 -           ^pkgconf@1.9.5%gcc@13.2.0 build_system=autotools arch=linux-rocky9-sandybridge
 -           ^pmix@5.0.1%gcc@13.2.0~docs+pmi_backwards_compatibility~python~restful build_system=autotools arch=linux-rocky9-sandybridge
 -               ^libevent@2.1.12%gcc@13.2.0+openssl build_system=autotools arch=linux-rocky9-sandybridge
 -           ^zlib-ng@2.1.4%gcc@13.2.0+compat+opt build_system=autotools arch=linux-rocky9-sandybridge
```

Installing the High Performance Linpack Benchmark

```
[matthew@moonshine spack]$ spack install hpl %gcc@13

==> Installing gmake-4.4.1-ufqig5vxmfp4j7cdu3trou5sxsnk3ire [1/39]
==> No binary for gmake-4.4.1-ufqig5vxmfp4j7cdu3trou5sxsnk3ire found: installing from source
==> Using cached archive: /home/matthew/spack/var/spack/cache/_source-cache/archive/dd/dd16fb1d67bfab79a72f5e8390735c49e3e8e70b4945a15ab1f81ddb78658fb3.tar.gz
==> No patches needed for gmake
==> gmake: Executing phase: 'install'
==> gmake: Successfully installed gmake-4.4.1-ufqig5vxmfp4j7cdu3trou5sxsnk3ire
  Stage: 0.09s.  Install: 17.88s.  Post-install: 0.03s.  Total: 18.03s
[+] /home/matthew/spack/opt/spack/linux-rocky9-sandybridge/gcc-13.2.0/gmake-4.4.1-ufqig5vxmfp4j7cdu3trou5sxsnk3ire
==> Installing ca-certificates-mozilla-2023-05-30-i3umzs4ore6nnlyxhuezcebzcc5zo3ow [2/39]
==> No binary for ca-certificates-mozilla-2023-05-30-i3umzs4ore6nnlyxhuezcebzcc5zo3ow found: installing from source
==> Using cached archive: /home/matthew/spack/var/spack/cache/_source-cache/archive/5f/5fadcae90aa4ae041150f8e2d26c37d980522cdb49f923fc1e1b5eb8d74e71ad
==> No patches needed for ca-certificates-mozilla
==> ca-certificates-mozilla: Executing phase: 'install'
==> ca-certificates-mozilla: Successfully installed ca-certificates-mozilla-2023-05-30-i3umzs4ore6nnlyxhuezcebzcc5zo3ow
  Stage: 0.00s.  Install: 0.00s.  Post-install: 0.03s.  Total: 0.06s
```

• • •

Customising Spack Installations

- We can control the compilation of spack packages and their dependencies

Use spack info {package name} to get a list of options.

+ enable a true/false option

~ disable a true/false option

{variant name}={value} to set an option that takes a text value

% specify a compiler

^ specify dependency options, can nest the specifications

Lets install HPC with OpenMP Support

```
[matthew@moonshine ~]$ spack info hpl
AutotoolsPackage:   hpl


Description:
    HPL is a software package that solves a (random) dense linear system in
       …
    openmp [false]                  false, true
        Enable OpenMP support

Build Dependencies:
    blas  gmake  gnuconfig  mpi


Link Dependencies:
    blas  mpi
```

By default OpenMP is turned off

Lets install HPC with OpenMP Support

```
[matthew@moonshine ~]$ spack install hpl +openmp %gcc@13
==> Installing gmake-4.4.1-nv5smzt3nio42k6bnueylndwiap2tmdh
[1/39]
==> No binary for gmake-4.4.1-nv5smzt3nio42k6bnueylndwiap2tmdh
found: installing
```

If you look at the new specification you won't see any changes.

Recall that OpenMP is built in to most modern compilers. So +openmp just sets a compiler flag.

Finding out about the packages you installed

```
$ spack find -lv hpl
-- linux-rocky8-cascadelake / gcc@11.2.0 ------------------------
5vvaumo hpl@2.3~openmp

-- linux-rocky8-cascadelake / gcc@13.2.0 ------------------------
wgypugl hpl@2.3+openmp build_system=autotools

-- linux-rocky8-skylake_avx512 / intel@18.0.4 --------------------
ftf6drc hpl@2.3~openmp

-- linux-rocky8-skylake_avx512 / intel@19.0.5 --------------------
rfycnrc hpl@2.3~openmp
-- linux-rocky8-skylake_avx512 / intel@20.0.4 --------------------
iyvkyz2 hpl@2.3~openmp
==> 5 installed packages
```

Working with Hashes and Showing Loaded Packages

```
$ mfricke@hopper:~ $ spack load /wgypugl
mfricke@hopper:~ $ spack find --loaded
-- linux-rocky8-cascadelake / gcc@12.1.0 ------------------------
-
hpl@2.3
==> 1 loaded package
```

Compiling with OpenMPI Support

We need to setup the following:

Add the Slurm integrated (Process management interface) libraries

Tell spack to use the Slurm we built and installed manually

Compile against Slurm’s built in PMI

Specify that MPI should use the UCX (Unified Communication) library for communication. UCX can use a variety of transports (e.g. shared memory, the Infiniband fabric*, and ethernet).

Inspect the hardware interface

```
[matthew@moonshine1 ~]$ lspci -Qvv | grep Mellanox
41:00.0 Network controller: Mellanox Technologies
MT27500 Family [ConnectX-3]
Subsystem: Mellanox Technologies Device 0017
```

Unified Communication X (UCX)

- **UCX** enables high-performance communication in HPC systems.
- Supports **InfiniBand**, **Ethernet**, **Omni-Path**, and **shared memory**.
- Provides low-latency, high-throughput for **MPI** and **GPU** workloads.
- Optimized for **RDMA** and high-performance fabrics like **InfiniBand**.
- Used in major frameworks like **OpenMPI** and **TensorFlow**.

- Written by Mellanox Technologies (recall the vendor of your IB card)
- Mellanox was bought by Nvidia in 2020 for $7 billion
  (Why do you think Nvidia wanted Mellanox?)

Check Slurm MPI Capabilities

```
[matthew@moonshine1 ]$ srun --mpi list
MPI plugin types are...
none
cray_shasta
pmi2
```

PMI-2 is an API for managing parallel processes and job control in high-performance computing environments.

Install Slurm RPM packages

Provides the PMI library.

```
[matthew@moonshine x86_64]$ sudo yum localinstall
slurm-libpmi-23.11.5-1.el9.x86_64.rpm
```

Provides the pmi.h header file so we can compile against the PMI library

```
[matthew@moonshine x86_64]$ sudo yum localinstall
slurm-devel-23.11.5-1.el9.x86_64.rpm
```

(Don’t forget to install these on your compute nodes too)

Too see what RPM packages contain, we can use yum-utils

```
[matthew@moonshine x86_64]$ sudo yum install yum-utils

[matthew@moonshine x86_64]$ repoquery -l slurm-devel-23.11.5-1.el9.x86_64.rpm
Last metadata expiration check: 1:06:18 ago on Fri 08 Nov 2024 11:31:53 PM MST.
/usr/include/slurm
/usr/include/slurm/pmi.h
/usr/include/slurm/pmi2.h
/usr/include/slurm/slurm.h
/usr/include/slurm/slurm_errno.h
/usr/include/slurm/slurm_version.h
/usr/include/slurm/slurmdb.h
/usr/include/slurm/smd_ns.h
/usr/include/slurm/spank.h
/usr/lib64/pkgconfig/slurm.pc
```

Tell Spack to use our existing Slurm Installation

```
[matthew@moonshine1 gcc-11]$ cat ~/.spack/packages.yaml
packages:
  slurm:
    externals:
    - spec: slurm@23.11.5
      prefix: /usr
      sysconfdir: /etc/slurm
    buildable: False
  rdma-core:
    externals:
    - spec: rdma-core@48.0
      prefix: /usr
    buildable: False
```

We will use the Slurm and RDMA packages we installed with yum earlier.

Tell Spack to use our existing Slurm Installation

```
[matthew@moonshine1 gcc-11]$ spack install hpl +openmp
^openmpi +pmi fabrics=ucx schedulers=slurm ^ucx +verbs
```

Compile hpl with the openmp variant, and specify that openmpi should be compiled with support for PMI, use Slurm to provide that PMI, and use the UCX fabric, UCX is compiled with verbs support.

Load the HPL package and its dependencies

```
[matthew@moonshine1 ]$ spack load hpl
[matthew@moonshine1 ]$ spack find --loaded
-- linux-rocky9-sandybridge / gcc@13.2.0 ------------------------
hpl@2.3
```

A useful command to see how openmpi is configured

```
[matthew@moonshine1]$ ompi_info | grep pmi
  Configure command line: '--prefix=/home/matthew/spack/opt/spack/linux-rocky9-
sandybridge/gcc-13.2.0/openmpi-4.1.6-gk2awutuuobtflrth25qdfgpy4ktrmb5' '--enable-shared' '--
disable-silent-rules' '--disable-sphinx' '--disable-builtin-atomics'
'--with-pmi=/usr' '--enable-static' '--enable-mpi1-compatibility' '--without-ofi' '--without-fca' '--
without-psm' '--without-mxm' '--without-xpmem' '--without-knem' '--
```

Spack will search this directory and subdirectories for headers and libraries

With Rocky 9.4 Connect-X 3 Inifniband Cards need the driver to be installed after install

```
[matthew@moonshine ~]# sudo yum install rdma-core libibverbs-utils librdmacm
librdmacm-utils ibacm infiniband-diags opensm

Dependencies resolved.
================================================================================
 Package                 Architecture Version                 Repository      Size
================================================================================
Installing:
 ibacm                   x86_64       46.0-1.el9              baseos          88 k
 infiniband-diags        x86_64       46.0-1.el9              appstream      310 k
 libibv                                                                      67 k
 librdm                                                                      70 k
 librdm                                                                      90 k
 opensm                                                                     503 k
 rdma-c                                                                      51 k
Installing dependencies:
 libibumad               x86_64       46.0-1.el9              baseos          26 k
 opensm-libs             x86_64       3.3.24-2.el9            baseos          75 k
 pciutils                x86_64       3.7.0-5.el9             baseos          92 k
```

We did this previously in Lecture 7 HPC Networking

You will need to reboot if this wasn't already installed

With Rocky 9.4 Connect-X 3 Inifniband Cards need the driver to be installed after install

```
[matthew@moonshine ~]$ sudo yum install ucx ucx-ib
Last metadata expiration check: 2:19:57 ago on Wed 05 Nov 2025 08:14:20 AM MST.
Dependencies resolved.
================================================================================
 Package                 Architecture                Version
            Repository                      Size
================================================================================
Installing:
 ucx                     x86_64                      1.17.0-
2.el9                    appstream                      833 k
 ucx-ib                  x86_64                      1.17.0-
2.el9                    appstream                      223 k
```

Make sure ucx libraries are installed

Load the HPL package and its dependencies

```
[matthew@moonshine1 ~]$ ucx_info -v | grep verbs
# Configured with: --disable-logging --disable-debug --disable-
assertions --disable-params-check --
prefix=/home/matthew/spack/opt/spack/linux-rocky9-
sandybridge/gcc-13.2.0/ucx-1.14.1-
46geh2fjjg2pxv5xkf7esyjwu7s2ajq5 --without-go --disable-
doxygen-doc --enable-numa --disable-assertions --enable-
compiler-opt=3 --without-java --enable-shared --enable-static -
-disable-logging --enable-mt --with-openmp --enable-
optimizations --disable-params-check --disable-gtest --with-pic
--without-cuda --disable-cma --without-dc --without-dm --
without-gdrcopy --without-ib-hw-tm --without-knem --without-
mlx5-dv --without-rc --without-ud --without-xpmem --without-
fuse3 --without-bfd --without-rdmacm --with-
verbs=/home/matthew/spack/opt/spack/linux-rocky9-
sandybridge/gcc-13.2.0/rdma-core-41.0-
fxc4n65qkm65athvxuuapfridnxfjdex --with-avx --without-rocm
```

Spack will search this directory and subdirectories for headers and libraries

Inspect the hardware interface

```
[matthew@moonshine1 ~]$  ibv_devinfo -d mlx4_0
hca_id: mlx4_0
transport: InfiniBand (0)
fw_ver: 2.10.600
node_guid: 0002:c903:0035:8090
sys_image_guid: 0002:c903:0035:8093
vendor_id: 0x02c9
vendor_part_id: 4099
hw_ver: 0x0
board_id: MT_1060110018
phys_port_cnt: 1
port: 1
state: PORT_ACTIVE (4)
max_mtu: 4096 (5)
active_mtu: 4096 (5)
sm_lid: 7
port_lid: 7
port_lmc: 0x00
link_layer: InfiniBand
```

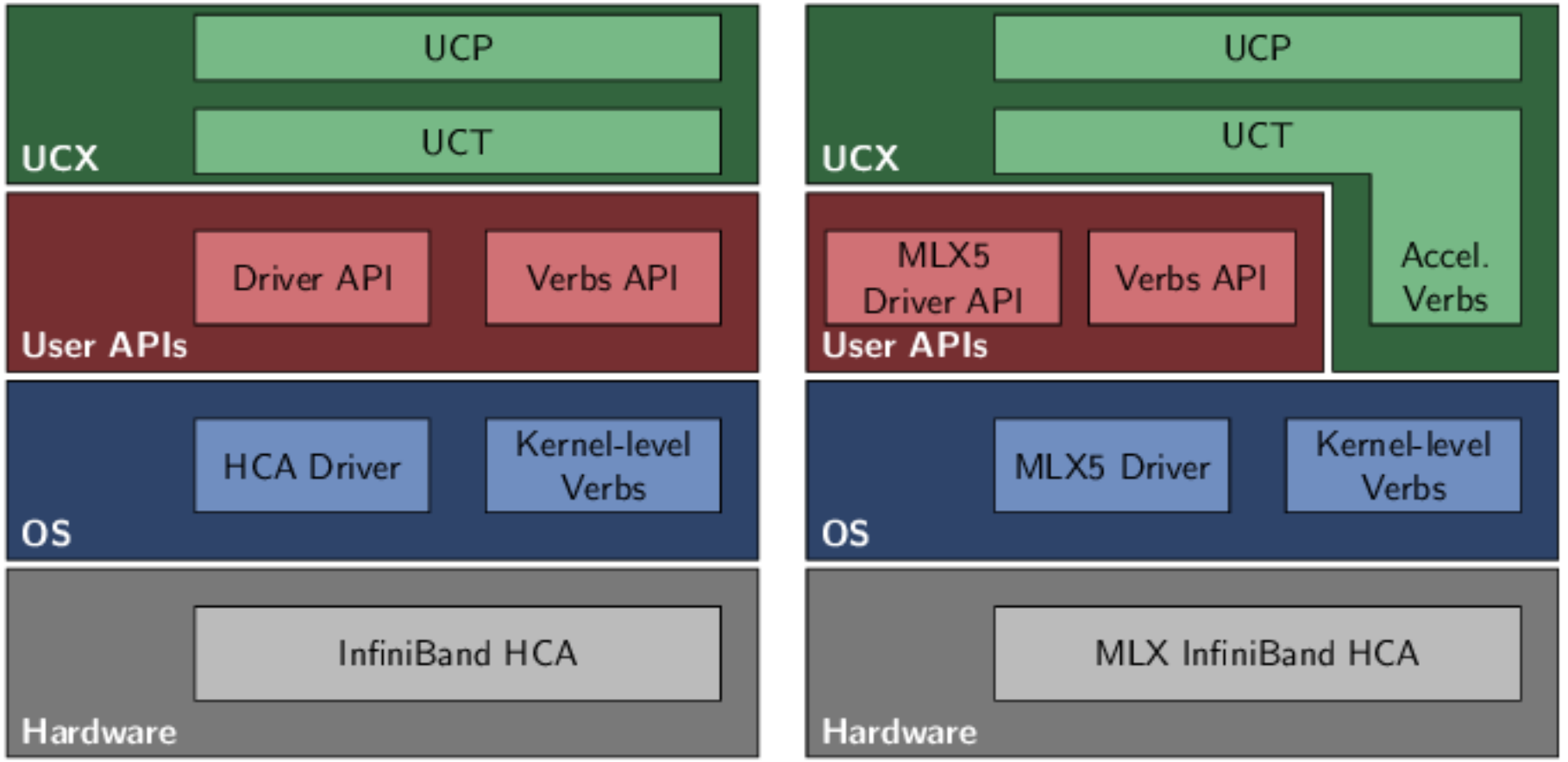
Now we will configure the Open MPI to use Infiniband with environment variables

Vader: shared memory communication

TCP: network communication

```
[matthew@moonshine1 ~]$ ompi_info
[moonshine1:1822684] mca: base: components_register: registering framework btl components
[moonshine1:1822684] mca: base: components_register: found loaded component self
[moonshine1:1822684] mca: base: components_register: component self register function successful
[moonshine1:1822684] mca: base: components_register: found loaded component sm
[moonshine1:1822684] mca: base: components_register: found loaded component tcp
[moonshine1:1822684] mca: base: components_register: component tcp register function successful
[moonshine1:1822684] mca: base: components_register: found loaded component vader
[moonshine1:1822684] mca: base: components_register: component vader register function successful
                 Package: Open MPI matthew@moonshine1 Distribution
                Open MPI: 4.1.6
```

Schematic of UCX, Verbs, and Infiniband



(a) Common InfiniBand OFED stack and UCX.

(b) MLX5 InfiniBand OFED stack and UCX with accelerated Verbs.

Papadopoulou, Nikela, Lena Oden, and Pavan Balaji. "A performance study of UCX over InfiniBand." *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 2017.

List available devices and transports:

```
[matthew@moonshine1 ~]$ ucx_info -d

# Memory domain: mlx4_0
#     Component: ib
#             register: unlimited, cost: 180 nsec
#           remote key: 8 bytes
#           local memory handle is required for zcopy
#         memory types: host (access,reg,cache)
#
#      Transport: rc_verbs
#         Device: mlx4_0:1
#           Type: network
#  System device: mlx4_0 (0)
#
#      capabilities:
#              bandwidth: 3438.60/ppn + 0.00 MB/sec
#                latency: 1300 + 1.000 * N nsec
#               overhead: 75 nsec
#              put_short: <= 88
#              put_bcopy: <= 8256
#              put_zcopy: <= 1G, up to 6 iov
#  put_opt_zcopy_align: <= 512
```

May need to be `ucx_info -u -d` if you have a later version of ucx

Let’s test all this to make sure we are using Infiniband for communication – first we will need a test program

```
// Minimal MPI program to demonstrate inter-rank
communication
#include <stdio.h>
#include <mpi.h>
int main(int argc, char *argv[]) {
    int rank, size; char message[20];
    // Initialize the MPI environment
    MPI_Init(&argc, &argv);
    // Get the number of processes
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    // Get the rank of the process
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    // Generate the message
    sprintf(message, "Hello from rank %d", rank);
```

Let’s test all this to make sure we are using Infiniband for communication – first we will need a test program

```
    // Send the message to the next rank (in a ring pattern)
 if (rank != size - 1) {
      MPI_Send(message, 20, MPI_CHAR, rank + 1, 0, MPI_COMM_WORLD);
            printf("Rank %d sent message: %s\n", rank, message); }

// Receive the message from the previous rank
if (rank != 0) {
     MPI_Recv(message, 20, MPI_CHAR, rank - 1, 0,
              MPI_COMM_WORLD, MPI_STATUS_IGNORE);
 printf("Rank %d received message: %s\n", rank, message);
    } else {
    // For rank 0, also print its own message
    printf("Rank %d received message: %s\n", rank, message); }

    // Finalize the MPI environment
    MPI_Finalize();
    return 0;
}
```

Use the program to test our network

We have access to mpi compilers because the HPL spack package is loaded

```
[matthew@moonshine1 ~]$ mpicxx mpihello.cpp -o mpihello
```

Enable verbose output so we can see what network is being used

```
[matthew@moonshine1 ~]$ export OMPI_MCA_btl_base_verbose=1000
```

Monitor TCP traffic over the Network interfaces with ifstat:

```
[matthew@moonshine1 ~]$ yum provides ifstat
Last metadata expiration check: 4 days, 18:09:28 ago on Fri 08 Nov 2024
04:41:22 PM MST.
ifstat-1.1-34.el9.x86_64 : Interface statistics
Repo        : epel
Matched from:
Provide    : ifstat = 1.1-34.el9


[matthew@moonshine1 ~]$ yum install ifstat
```

Monitor TCP traffic over the Network interfaces:

```
[matthew@moonshine1 ~]$ watch ifstat  --interval 2 ibp65s0 eno1 eno2

Every 2.0s: ifstat --interval 2 ibp65s0 eno1 eno2
moonshine1: Sat Nov  9 18:46:16 2024


#kernel
Interface        RX Pkts/Rate    TX Pkts/Rate    RX Data/Rate    TX Data/Rate
                 RX Errs/Drop    TX Errs/Drop    RX Over/Rate    TX Coll/Rate
eno1                   14 0            2 0          1436 0          276 0
                        0 2            0 0             0 0            0 0
eno2                    1 0            0 0            64 0            0 0
                        0 0            0 0             0 0            0 0
ibp65s0                 0 0            0 0             0 0            0 0
                        0 0            0 0             0 0            0 0
```

Depending on the ifstat version you end up with you might need to run :

```
ifstat -i eno1,eno2,ibp65s0
```

Use the program to test transport

```
[matthew@moonshine1 ~]$ srun --mpi=pmi2 --ntasks 16 ./mpihello
[moonshine1:2600569] mca: base: components_register: registering framework btl components
[moonshine1:2600569] mca: base: components_register: found loaded component self
[moonshine1:2600571] mca: base: components_register: registering framework btl components
[moonshine1:2600571] mca: base: components_register: found loaded component self
[moonshine1:2600569] mca: base: components_register: component self register function successful
[moonshine1:2600571] mca: base: components_register: component self register function successful
[moonshine1:2600569] mca: base: components_register: found loaded component sm
[moonshine1:2600569] mca: base: components_register: found loaded component tcp
[moonshine1:2600571] mca: base: components_register: found loaded component sm
[moonshine1:2600762] select: initializing btl component vader
[moonshine1:2600764] select: init of component vader returned success
[moonshine1:2600676] btl:tcp: examining interface ibp65s0
[moonshine1:2600676] btl:tcp: using ipv6 interface ibp65s0

Rank 1 sent message: Hello from rank 1
Rank 1 received message: Hello from rank 0
Rank 2 sent message: Hello from rank 2
Rank 2 received message: Hello from rank 1
Rank 10 sent message: Hello from rank 10
Rank 10 received message: Hello from rank 9
```

Message Transport Methods:

Vader: Shared Memory (intranode)

Self: Interprocess (intranode)

TCP: Network TCP Layer

Use the program to test transport

```
[matthew@moonshine1 ~]$ srun --mpi=pmi2 --ntasks 16 ./mpihello
[moonshine1:2600569] mca: base: components_register: registering framework btl components
[moonshine1:2600569] mca: base: components_register: found loaded component self
[moonshine1:2600571] mca: base: components_register: registering framework btl components
[moonshine1:2600571] mca: base: components_register: found loaded component self
[moonshine1:2600569] mca: base: components_register: component self register function successful
[moonshine1:2600571] mca: base: components_register: component self register function successful
[moonshine1:2600569] mca: base: components_register: found loaded component sm
[moonshine1:2600569] mca: base: components_register: found loaded component tcp
[moonshine1:2600571] mca: base: components_register: found loaded component sm
[moonshine1:2600762] select: initializing btl component vader
[moonshine1:2600764] select: init of component vader returned success
[moonshine1:2600676] btl:tcp: examining interface ibp65s0
[moonshine1:2600676] btl:tcp: using ipv6 interface ibp65s0

Rank 1 sent message: Hello from rank 1
Rank 1 received message: Hello from rank 0
Rank 2 sent message: Hello from rank 2
Rank 2 received message: Hello from rank 1
Rank 10 sent message: Hello from rank 10
Rank 10 received message: Hello from rank 9
```

TCP over infiniband

HPL UCX Tries to lock too much memory

```
[matthew@moonshine1 ~]$ srun --nodes 2 --ntasks-per-node 2 xhpl


[1731519936.439711]
[moonshine2:15321:0]          ib_log.c:246  UCX  ERROR
ibv_reg_mr(address=0x2200000, length=6291456, access=0xf)
failed: Cannot allocate memory. Please set max locked memory
(ulimit -l) to 'unlimited' (current: 8192 kbytes)
[1731519936.439717]
[moonshine2:15321:0]           mpool.c:269  UCX  ERROR Failed to
allocate memory pool (name=ud_tx_skb) chunk: Input/output error
```

TCP over infiniband

Linux Limits the User Resources

The /etc/security/limits.conf file administrators to set limits on resources such as the maximum number of open files, maximum processes per user, CPU usage, and memory allocation for users or groups.

The file supports **hard and soft limits**:

- **Soft limits**: can be temporarily exceeded by users.
- **Hard limits**: are strict caps that cannot be surpassed.

Users can set their resource limits (up to the limits defined in limits.conf)

```
[matthew@moonshine1 ~]$ ulimit -l unlimited
-bash: ulimit: max locked memory: cannot modify limit: Operation not permitted
```

Sets the amount of locked memory user “Matthew” can use.
UCX liked to lock a lot of memory for on node communication.

If you get an error set the limits in /etc/security/limits.conf

```
#<domain>        <type>  <item>  <value>

* soft memlock unlimited
* hard memlock unlimited
```

“*” Sets the limit for all users
Since they are both unlimited soft vs hard doesn’t matter

Let’s try again after relogging.

```
[matthew@moonshine1 ~]$ ulimit -l unlimited
[matthew@moonshine1 ~]$ spack load hpl
[matthew@moonshine1 ~]$ srun --nodes 2 --ntasks-per-node 2 xhpl
```

```
[matthew@moonshine1 ~]$ ifstat -i eno1,eno2,ibp65s0
       eno1                eno2               ibp65s0
 KB/s in  KB/s out   KB/s in  KB/s out   KB/s in  KB/s out
    0.89       0.97       0.06       0.00       0.00       0.00
    1.30       0.51       0.00       0.00       0.00       0.00
    1.40       0.83       0.15       0.00       0.00       0.00
    1.34       0.60       0.00       0.09       0.00       0.00
    0.74       1.12       0.06       0.00       0.00       0.00
    1.11       0.62       0.00       0.00       0.00       0.00
    1.29       0.85       0.00       0.00       0.00       0.00
    0.50       0.65       0.06       0.00       0.00       0.00
    1.45       0.82       0.00       0.00       0.00       0.00
    1.38       2.81      79.25     117.06       0.00       0.00
    1.33       0.87      22.90      36.49       0.00       0.00
```