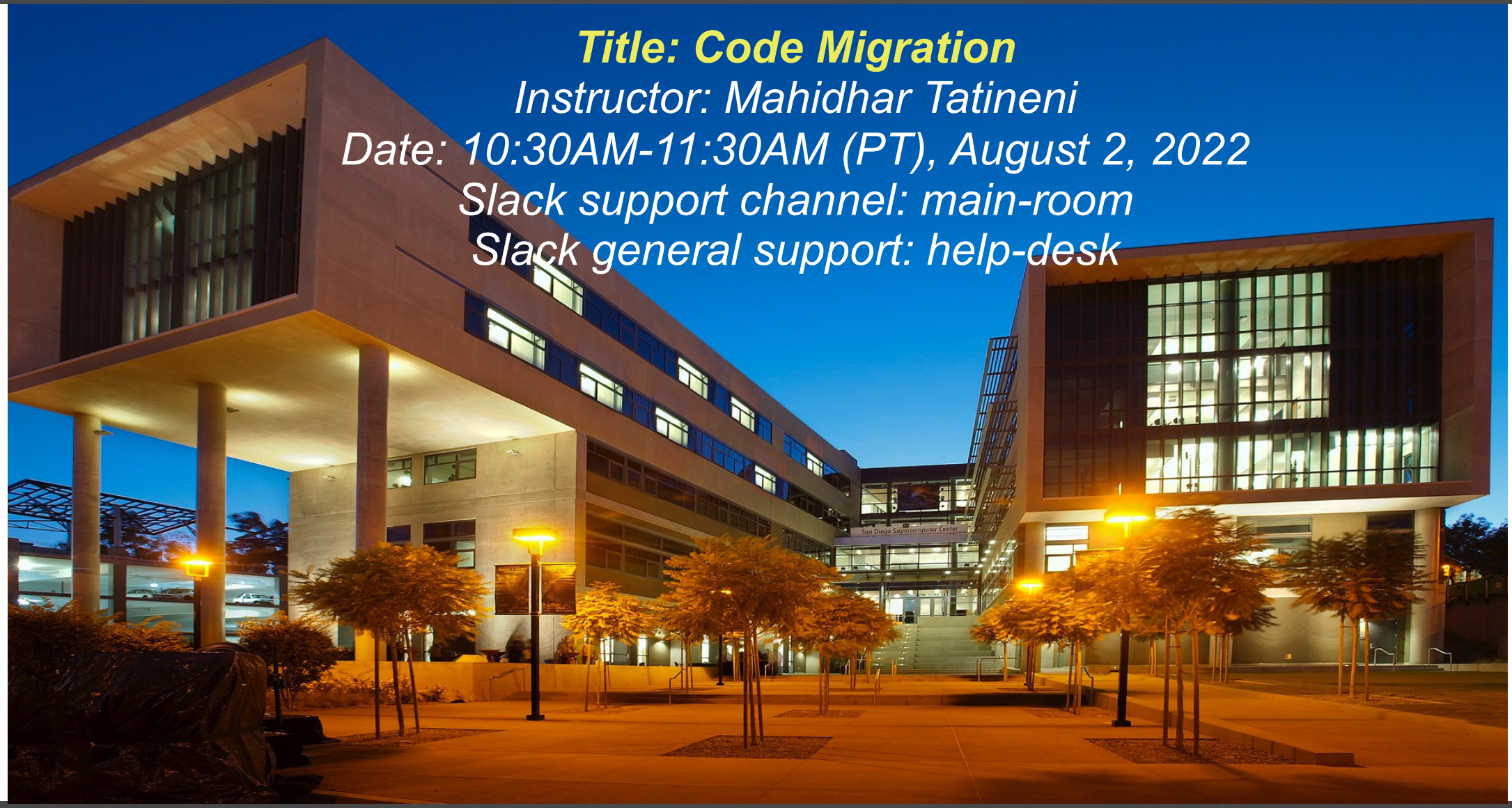
SDSC Summer Institute 2022



Title: Code Migration

Instructor: Mahidhar Tatineni

Date: 10:30AM-11:30AM (PT), August 2, 2022

Slack support channel: main-room

Slack general support: help-desk

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

Applications already available on Expanse

- SDSC staff have installed and made available a large suite of libraries and applications on Expanse. This includes commercially licensed software such as Q-Chem, Abaqus, and Gaussian.
- The primary approach is to make applications available using environment modules.
- Use “module spider” to find applications that are available via modules.
- Some applications/frameworks (e.g. PyTorch, TensorFlow) have been made available using singularity containers.

Applications via modules

- Use “module spider” to find applications.

```
[mahidhar@login02 ~]$ module spider matlab

----------------------------------------------------------------------------------------------------------------
  matlab:
----------------------------------------------------------------------------------------------------------------
     Versions:
        matlab/2020b
        matlab/2022a

----------------------------------------------------------------------------------------------------------------
  For detailed information about a specific "matlab" package (including how to load the modules) use the module's full n
ame. Note that names that have a trailing (E) are extensions provided by other modules.
  For example:

     $ module spider matlab/2022a
----------------------------------------------------------------------------------------------------------------



[mahidhar@login02 ~]$
```

Applications via modules

- Use “module spider” to find applications.

```
[mahidhar@login02 ~]$ module spider matlab/2020b

----------------------------------------------------------------------------------------------------------------
  matlab: matlab/2020b
----------------------------------------------------------------------------------------------------------------

    You will need to load all module(s) on any one of the lines below before the "matlab/2020b" module is available to l
oad.

      cpu/0.15.4
      gpu/0.15.4

    Help:
      MATLAB (MATrix LABoratory) is a multi-paradigm numerical computing
      environment and fourth-generation programming language. A proprietary
      programming language developed by MathWorks, MATLAB allows matrix
      manipulations, plotting of functions and data, implementation of
      algorithms, creation of user interfaces, and interfacing with programs
      written in other languages, including C, C++, C#, Java, Fortran and
      Python. Note: MATLAB is licensed software. You will need to create an
      account on the MathWorks homepage and download MATLAB yourself. Spack
      will search your current directory for the download file. Alternatively,
      add this file to a mirror so that Spack can find it. For instructions on
```

Applications via modules

```
[mahidhar@login02 ~]$ module spider fftw/3.3.8

----------------------------------------------------------------------------------------------------
  fftw: fftw/3.3.8
----------------------------------------------------------------------------------------------------

    You will need to load all module(s) on any one of the lines below before the "fftw/3.3.8" module is available to loa
d.

      cpu/0.15.4  gcc/10.2.0
      cpu/0.15.4  gcc/10.2.0  mvapich2/2.3.4
      cpu/0.15.4  gcc/10.2.0  openmpi/4.0.4
      cpu/0.15.4  gcc/10.2.0  openmpi/4.0.4-openib
      cpu/0.15.4  gcc/9.2.0  mvapich2/2.3.6
      cpu/0.15.4  gcc/9.2.0  openmpi/3.1.6
      cpu/0.15.4  gcc/9.2.0  openmpi/3.1.6-cxx
      cpu/0.15.4  intel/19.1.1.217  intel-mpi/2019.8.254
      cpu/0.15.4  intel/19.1.1.217  mvapich2/2.3.4
      cpu/0.15.4  intel/19.1.1.217  openmpi/4.0.4
      gpu/0.15.4

    Help:
      FFTW is a C subroutine library for computing the discrete Fourier
      transform (DFT) in one or more dimensions, of arbitrary input size, and
```

Applications via modules

```
[mahidhar@login02 ~]$ module reset
Resetting modules to system default. Reseting $MODULEPATH back to system default. All extra directories will be removed
from $MODULEPATH.
[mahidhar@login02 ~]$ clear

[mahidhar@login02 ~]$ module load cpu/0.15.4  gcc/10.2.0  mvapich2/2.3.4
[mahidhar@login02 ~]$ module load fftw/3.3.8
[mahidhar@login02 ~]$ module show fftw/3.3.8
-------------------------------------------------------------------------------------------------
   /cm/shared/apps/spack/cpu/lmod/linux-centos8-x86_64/mvapich2/2.3.4-szxlftd/gcc/10.2.0/fftw/3.3.8.lua:
-------------------------------------------------------------------------------------------------
whatis("Name : fftw")
whatis("Version : 3.3.8")
whatis("Target : zen2")
whatis("Short description : FFTW is a C subroutine library for computing the discrete Fourier transform (DFT) in one or
more dimensions, of arbitrary input size, and of both real and complex data (as well as of even/odd data, i.e. the discr
ete cosine/sine transforms or DCT/DST). We believe that FFTW, which is free software, should become the FFT library of c
hoice for most applications.")
help([[FFTW is a C subroutine library for computing the discrete Fourier
```

Applications via modules

- "module spider" also gives the loading information
- The CPU and GPU stacks are completely independent. Do *not* mix the two as something compiled for the GPU stack will not work on the CPU nodes (which have a different architecture) and vice versa
- Usage examples are provided in:

  /cm/shared/examples/sdsc

```
[mahidhar@login02 sdsc]$ cd /cm/shared/examples/sdsc/
[mahidhar@login02 sdsc]$ ls
abaqus     bintest  dftbplus  gromacs  localscratch       namd    openacc  pyscf    raxml  tensorflow  vasp-ase
abinit     ciml     excerpt   hadoop   matlab             neuron  openmp   pytorch  si     test        visit
alphafold  classes  gamess    hpl      mpi                nsight  orca     qchem    siesta trinity     wannier90
amber      cp2k     gaussian  lammps   mpi-openmp-hybrid  nwchem  paraview qe       spark  vasp        xpmem
```

Applications available via Singularity Containers

- **Some applications are easier to make available via Singularity containers.**
- **On Expanse the containers are at:**
  - /cm/shared/apps/containers/singularity

- **Applications available via Singularity include:**
  - TensorFlow, PyTorch, AlphaFold, Paraview, VisIt

- **The Singularity definition files are available* for users who wish to add to the containers/rebuild them.**

*** https://github.com/mkandes/naked-singularity/tree/master/definition-files**

Using Applications via Singularity Containers

```
#!/usr/bin/env bash
#SBATCH --job-name=pytorch-gpu-shared
#### Change account below
#SBATCH --account=XYZ123
#SBATCH --partition=gpu-shared
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=10
#SBATCH --cpus-per-task=1
#SBATCH --mem=90G
#SBATCH --gpus=1
#SBATCH --time=00:30:00
#SBATCH --output=pytorch-gpu-shared.o%j.%N
module reset
module load singularitypro
time -p singularity exec --bind /expanse,/scratch --nv
/cm/shared/apps/containers/singularity/pytorch/pytorch-1.5.0-gpu-20200511.simg  python3
/opt/pytorch-1.5.0/examples/mnist/main.py
```

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

Python applications/libraries via modules

- **Several python applications and libraries are available via modules**
- **Examples**
  - py-matplotlib, py-numpy, py-scipy
  - py-pysam, py-phonopy, py-htseq
- **Use “module spider” and “module show” to get more information**
- **Some of these modules are dependent on MPI like py-pyscf and py-mpi4py**
- **Use modules-based approach if all the packages needed are in modules.**
- **If further installs are needed, its best to go to the miniconda3 or Singularity approach.**
- **Don’t mix the python apps/libraries from the modules with ones from miniconda3 or a container. They will likely conflict/fail.**

Installs using miniconda3

- **Use miniconda3 if the software available on the system doesn't cover your needs. Specially if custom installs with both python and non-python dependencies are required.**
- **Install in your home directory. *Do not install miniconda3 into any Lustre location - can cause systemwide problems due to metadata loads.***
- **For GPU installs, make sure the packages chosen work with driver on Expanse. For example, the current versions are:**
  - NVIDIA-SMI 510.39.01 Driver Version: 510.39.01 CUDA Version: 11.6
  - Make sure any conda based installs are with a CUDA version that is 11.6 or older.

Installs using miniconda3

```
[xdtr112@login02 ~]$ srun --pty --nodes=1 --ntasks-per-node=1 --cpus-per-task=4 --gpus=1 --mem=32G --account=crl155 -t 0
0:30:00 -p gpu-debug --wait 0 /bin/bash
srun: job 14833460 queued and waiting for resources
srun: job 14833460 has been allocated resources
[xdtr112@exp-7-59 ~]$ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
--2022-08-01 20:47:10--  https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
Resolving repo.anaconda.com (repo.anaconda.com)... 104.16.131.3, 104.16.130.3, 2606:4700::6810:8203, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|104.16.131.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76607678 (73M) [application/x-sh]
Saving to: 'Miniconda3-latest-Linux-x86_64.sh'

Miniconda3-latest-Linux-x86_6 100%[=================================================>]  73.06M   110MB/s    in 0.7s

2022-08-01 20:47:11 (110 MB/s) - 'Miniconda3-latest-Linux-x86_64.sh' saved [76607678/76607678]

[xdtr112@exp-7-59 ~]$ sh Miniconda3-latest-Linux-x86_64.sh

Welcome to Miniconda3 py39_4.12.0

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>
```

Installs using miniconda3

```
(base) [xdtr112@exp-7-59 ~]$ conda search -c nvidia cuda
Loading channels: done
# Name                       Version           Build  Channel
cuda                          11.3.0      h3b286be_0  nvidia
cuda                          11.4.0      hf865f46_0  nvidia
cuda                          11.4.1      h2daf1ce_0  nvidia
cuda                          11.4.2      hbcc0205_0  nvidia
cuda                          11.5.0      hc28fa2a_0  nvidia
cuda                          11.5.1      hd4d9352_0  nvidia
cuda                          11.6.0      hde35cc3_0  nvidia
cuda                          11.6.1      h755e45f_0  nvidia
cuda                          11.6.2      h8144a35_0  nvidia
cuda                          11.7.0               0  nvidia
(base) [xdtr112@exp-7-59 ~]$ conda install -c nvidia cuda==11.6.0
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/xdtr112/miniconda3

  added / updated specs:
    - cuda==11.6.0
```

```
cuda-cudart-11.7.60  | 195 KB    | ######################################################################## | 100%
cuda-nvtx-11.7.50    | 58 KB     | ######################################################################## | 100%
cuda-nvprune-11.7.50 | 65 KB     | ######################################################################## | 100%
libcusolver-dev-11.3 | 62.2 MB   |                                                                        |   0%
libcusolver-dev-11.3 | 62.2 MB   | ######################################################################## | 100%
cuda-nvdisasm-11.7.5 | 31.5 MB   | ######################################################################## | 100%
cuda-runtime-11.7.0  | 1 KB      | ######################################################################## | 100%
cuda-toolkit-11.7.0  | 1 KB      | ######################################################################## | 100%
libnvjpeg-11.7.2.34  | 2.3 MB    | ######################################################################## | 100%
cuda-11.6.0          | 2 KB      | ######################################################################## | 100%
cuda-nvcc-11.7.64    | 42.7 MB   | ######################################################################## | 100%
libnvjpeg-dev-11.7.2 | 2.0 MB    | ######################################################################## | 100%
cuda-gdb-11.7.50     | 4.8 MB    | ######################################################################## | 100%
libcusparse-dev-11.7 | 301.2 MB  |                                                                        |   0%
libcusparse-dev-11.7 | 301.2 MB  | ######################################################################## | 100%
libnpp-11.7.3.21     | 118.5 MB  | ######################################################################## | 100%
libnpp-dev-11.7.3.21 | 115.7 MB  | ######################################################################## | 100%
cuda-libraries-11.7. | 1 KB      | ######################################################################## | 100%
libcublas-11.10.1.25 | 299.9 MB  | ######################################################################## | 100%
libcurand-dev-10.2.1 | 50.7 MB   | ######################################################################## | 100%
cuda-cuxxfilt-11.7.5 | 284 KB    | ######################################################################## | 100%
cuda-cudart-dev-11.7 | 1008 KB   | ######################################################################## | 100%
cuda-libraries-dev-1 | 1 KB      | ######################################################################## | 100%
cuda-tools-11.7.0    | 1 KB      | ######################################################################## | 100%
cuda-command-line-to | 1 KB      | ######################################################################## | 100%
libcufft-10.7.2.124  | 93.6 MB   | ######################################################################## | 100%
cuda-sanitizer-api-1 | 16.7 MB   | ######################################################################## | 100%
cuda-cuobjdump-11.7. | 159 KB    | ######################################################################## | 100%
libcusolver-11.3.5.5 | 89.2 MB   | ######################################################################## | 100%
cuda-cupti-11.7.50   | 22.9 MB   | ######################################################################## | 100%
cuda-nvvp-11.7.50    | 114.3 MB  | ######################################################################## | 100%
```

mpi4py

- mpi4py install needs to be consistent with the MPI being used. The system installed versions:

```
[xdtr112@login02 ~]$ module spider py-mpi4py/3.0.3

----------------------------------------------------------------------------------------------------------------------------------
  py-mpi4py: py-mpi4py/3.0.3
----------------------------------------------------------------------------------------------------------------------------------

    You will need to load all module(s) on any one of the lines below before the "py-mpi4py/3.0.3" module is available to load.

      cpu/0.15.4  gcc/10.2.0  mvapich2/2.3.6
      cpu/0.15.4  gcc/10.2.0  openmpi/4.0.4
      cpu/0.15.4  gcc/10.2.0  openmpi/4.0.4-openib
      gpu/0.15.4  openmpi/4.0.4

    Help:
      This package provides Python bindings for the Message Passing Interface
      (MPI) standard. It is implemented on top of the MPI-1/MPI-2
      specification and exposes an API which grounds on the standard MPI-2 C++
      bindings.
```

- miniconda3 install will use a conda based mpi install. This is ok for single node cases, but multi-node will not use the high-performance InfiniBand network.
- Use system installed MPI and combine with miniconda3 by building mpi4py from source.

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

R applications/libraries via modules

- **Several R applications and libraries are available via modules**
- **Use “module spider r-” to find all installed apps/libraries.**
- **Examples**
  - r-biobase, r-deseq2, r-doparallel, r-ggplot2
- **Use modules-based approach if all the packages needed are in modules.**
- **If further installs are needed, its best to go to the miniconda3 or Singularity approach.**
- **Don’t mix the R apps/libraries from the modules with ones from miniconda3 or a container. They will likely conflict/fail.**

Example of using R via Singularity

```
[xdtr112@exp-9-55 ~]$ export TMPDIR=/scratch/$USER/job_$SLURM_JOBID
[xdtr112@exp-9-55 ~]$ module load singularitypro
[xdtr112@exp-9-55 ~]$ singularity build excerpt.sif docker://rkitchen/excerpt
WARNING: 'nodev' mount option set on /scratch, it could be a source of failure during build process
INFO:    Starting build...
Getting image source signatures
Copying blob 5e35d10a3eba done
Copying blob cc17f052e960 done
Copying blob 6059ce0d04ff done
Copying blob be036d7c9474 done
Copying blob a59ce8f0c359 done
Copying blob 236bb8549592 done
Copying blob e6adcfd80d7c done
```

Example of using R via Singularity

```
#!/bin/bash
#SBATCH --job-name="excerpt-test"
#SBATCH --output="excerpt.%j.%N.out"
#SBATCH --partition=shared
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=8
#SBATCH --mem=16G
#SBATCH --account=XYZ123
#SBATCH -t 04:00:00

### Modules
module reset
module load singularitypro

### Run the job
singularity run --bind $SLURM_SUBMIT_DIR/input:/exceRptInput --bind $SLURM_SUBMIT_DIR/output:/exceRptOutput --
bind /expanse/projects/qstore/data/excerpt/hg38:/exceRpt_DB/hg38
/cm/shared/apps/containers/singularity/excerpt/excerpt.sif INPUT_FILE_PATH=/exceRptInput/SRR026761.sra
```

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

Installs from source, configure example

- **Some applications have configure scripts that pick up dependencies based on environment variables and options.**
- **Example: Quantum Espresso**
- **Build using a job script:**
  - /cm/shared/examples/sdsc/qe/build_scripts/qe_gcc92_openmpi.sh
- **Compile environment:**

```
module reset
module load gcc/9.2.0
module load openmpi/3.1.6
module load amdblis
module load amdlibflame
module load amdfftw
```

Installs from source, QE example continued

- **Compiler flags**

```
CC=gcc
CXX=g++
F77=gfortran
FC=gfortran
F90=gfortran
export CFLAGS="-O3 -march=core-avx2"
export CXXFLAGS="-O3 -march=core-avx2"
export FCFLAGS="-O3 -march=core-avx2  "
export F90FLAGS="-O3 -march=core-avx2 -cpp "
export F77FLAGS="-O3 -march=core-avx2  "
export FFLAGS="-O3 -march=core-avx2 "
```

Installs from source, QE example continued

- **Environment variables**

```
export FFT_LIBS="-L${AMDFFTWHOME}/lib -lfftw3"
export FFT_INCLUDE="-I{AMDFFTWHOME}/include"
export FFTW_INCLUDE="-I{AMDFFTWHOME}/include"
export BLAS_LIBS="-L${AMDBLISHOME}lib -lblis"
export LAPACK_LIBS="-L${AMDLIBFLAMEHOME}/lib -lflame"
export IFLAGS="-I../include -I${AMDFFTWHOME}/include -
I${AMDBLISHOME}/include -I${AMDLIBFLAMEHOME}/include"
export SCALAPACK_LIBS="-L${HOME}/scalapack/lib -lscalapack"
```

Installs from source, QE example continued

```
./configure \
    CC=$CC \
    CXX=$CXX \
    F77=$F77 \
    FC=$FC \
    F90=$FC \
    --prefix=$HOME/qe

cp make.sys make.sys.bak
make $ESPRESSO_TARGETS
make install
```

Installs from source, Makefile example

```
F90 = gfortran
# FFLAGS = -Wall -fbounds-check
# FFLAGS = -g -Wall -fcheck=all
FFLAGS = -O2
LIBFLAGS = -L/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen2/gcc-10.2.0/openblas-0.3.10-3lzjcwjsyu3qmott7k3k52mtzljioax3/lib -L
/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen2/gcc-10.2.0/fftw-3.3.8-zwmcmd2v5albxc2n5u5njzbou6r5vn3u/lib -lopenblas -lfftw3
INCFLAGS = -I/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen2/gcc-10.2.0/openblas-0.3.10-3lzjcwjsyu3qmott7k3k52mtzljioax3/includ
e -I/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen2/gcc-10.2.0/fftw-3.3.8-zwmcmd2v5albxc2n5u5njzbou6r5vn3u/include
PDFL = pdflatex -synctex=1

DRIVERS = driver1 driver2 driver3 driver4

default: oz.so

all : $(DRIVERS) fftw_test oz.so

docs : oz_doc.pdf

oz_doc.pdf : oz_doc.tex gofr.png sofk.png
        $(PDFL) oz_doc.tex
        $(PDFL) oz_doc.tex
        $(PDFL) oz_doc.tex

drivers : $(DRIVERS)

oz.so : oz_mod.f90
        f2py3 --overwrite-signature $< -m oz -h oz.pyf
        f2py3 -c $< oz.pyf $(INCFLAGS) $(LIBFLAGS)
```

Installs from source

- **All GPU compiles *must* be done on a GPU node. Also, don't mix installs from GPU and CPU stack**
- **Compiles done on the login node will fail on a GPU node**
- **CPU codes can be compiled on the login node as the processor matches**
- **Note on BLAS/LAPACK/SCALAPACK: there are several options**
  - OpenBLAS,
  - Netlib Scalapack
  - MKL
  - AOCL
- **Libraries will be in non-standard locations. So, make sure to use configure/cmake/makefile options to point build scripts to the right locations**
- **Intel compilers work fine on AMD nodes. Don't use "-xHOST", switch to "-march=core-avx2"**

Outline

- **Applications already available on Expanse**
  - modules
  - singularity containers
- **Python based applications/libraries**
  - SDSC installed and available via modules
  - miniconda3
  - mpi4py
- **R based applications**
  - SDSC installed and available via modules
  - miniconda3, Singularity approaches
- **Installing/building applications from source code**
- **User built containers**

Building Singularity containers

- **Useful if there are a lot of dependencies that cannot be easily installed in the regular Expanse environment**
- **For MPI based installs:**
  - The MPI in the container should match the external MPI version
  - Make sure the InfiniBand drivers are installed in the container
  - We have example definition files available
- **For GPU installs:**
  - Make sure packages installed are compatible with driver on our system
- **Builds from definition files need root access and cannot be done on Expanse. Build elsewhere - for example use your laptop/desktop OR a cloud resource (e.g. Jetstream2 on XSEDE)**
- **More details in Marty's talk on Thursday (Aug 4)**

Sample Singularity definition file

```
Bootstrap: shub
From: mkandes/naked-singularity:centos-7.9.2009

%labels

    APPLICATION_NAME centos + mvapich
    APPLICATION_VERSION 7.9.2009 + 2.3.2
    APPLICATION_URL https://mvapich.cse.ohio-state.edu

    AUTHOR_NAME Marty Kandes
    AUTHOR_EMAIL mkandes@sdsc.edu

    LAST_UPDATED 20201227

%setup

%environment

    # Set paths to MVAPICH2 binaries and libraries
    export PATH="/opt/mvapich2-2.3.2/bin:${PATH}"
    export LD_LIBRARY_PATH="/opt/mvapich2-2.3.2/lib:${LD_LIBRARY_PATH}"

%post -c /bin/bash

    # Set operating system mirror URL
    export MIRRORURL='http://mirror.centos.org/centos-7/7.9.2009/os/x86_64'

    # Set operating system version
    export OSVERSION='7'

    # Set system locale
    export LC_ALL=C

    # Update all software packages to their latest versions
    yum -y check-update && yum -y update

    # Install basic drivers for user space access to Ethernet, RDMA,
    # and Infiniband. See https://community.mellanox.com/docs/DOC-2431
    yum -y install dkms
    yum -y install infiniband-diags
    yum -y install infiniband-diags-devel
    yum -y install libibverbs
    yum -y install libibverbs-devel
    yum -y install ibacm
    yum -y install librdmacm
    yum -y install librdmacm-devel
    yum -y install libmlx4
    yum -y install libmlx5
    yum -y install mstflint
    yum -y install libibcm
    yum -y install libibmad
    yum -y install libibmad-devel
    yum -y install libibumad
    yum -y install libibumad-devel
    yum -y install opensm
    yum -y install srptools

    # Install additional tools
    yum -y install ibutils
    yum -y install libibverbs-utils
    yum -y install librdmacm-utils
    yum -y install perftest
    yum -y install numactl

    # Install libnl
    yum -y install libnl3
    yum -y install libnl3-devel

    # Install mvapich2 (build) dependencies
    yum -y install bison

    cd /tmp

    # Download, build, and install mvapich2
    wget http://mvapich.cse.ohio-state.edu/download/mvapich/mv2/mvapich2-2.3.2.tar.gz
    tar -xzvf mvapich2-2.3.2.tar.gz
    cd mvapich2-2.3.2
    ./configure --prefix=/opt/mvapich2-2.3.2
    make
    make install

    # Cleanup
    package-cleanup -q --leaves | xargs -l1 yum -y remove
    yum -y clean all
```

Summary

- **Several options to get applications working on Expanse**
- **Check if application is already installed - either via modules or in Singularity images**
- **Examples directory (/cm/shared/examples/sdsc)**
- **Several options for R and python: 1) installs available via modules; 2) miniconda3 installations in user directories; 3) containers**
- **Do not mix software installations on system with conda based installs - try to keep the entire application tree needed for a workflow in one environment.**
- **Can use docker images via Singularity**
- **Build your own containers from definition files**
  - keep InfiniBand stack in the container consistent with the one used on system
  - GPU application/library installs must be compatible with drivers on the system