How to install Firedrake

YANG Zongze

2023 年 5 月 7 日

目录

1 Installation of Firedake

To install firedrake, the computer should have access to the Internet. Otherwise, please refer to Section 2: Installation without Network

1.1 Ubuntu

The easiest way to intall firedrake is to download the installation script `firedrake-install` and run it using Python. This method will intall the real number version by defaults.

```
curl -O \
    https://raw.githubusercontent.com/firedrakeproject/firedrake/master/scripts/firedrake-install
python3 firedrake-install
```

If you need to know more installation options, please refer to the help documentation.

```
python3 firedrake-install -h
```

Remark: Sometimes there may be issues with accessing the `pip` source during installation, and error messages similar to the following may appear:

```
Starting new HTTPS connection (6): pypi.org:443
Could not fetch URL https://pypi.org/simple/pulp/: connection error:
 ↪  HTTPSConnectionPool(host='pypi.org', port=443): Max retries exceeded with url: /simple/pulp/
 ↪  (Caused by NewConnectionError('<pip._vendor.urllib3.connection.HTTPSConnection object at
 ↪  0x7f43dce52bc0>: Failed to establish a new connection: [Errno 101] Network is unreachable'))
 ↪  - skipping
Skipping link: not a file: https://pypi.org/simple/pulp/
Given no hashes to check 0 links for project 'pulp': discarding no candidates
ERROR: Could not find a version that satisfies the requirement PuLP (from versions: none)
ERROR: No matching distribution found for PuLP
```

This can be fixed by setting the source of `pip`, such as changing it to the source of USTC:

```
mkdir -p $HOME/.pip && \
cat > $HOME/.pip/pip.conf <<EOF
[global]
index-url = https://pypi.mirrors.ustc.edu.cn/simple
[install]
trusted-host=pypi.mirrors.ustc.edu.cn
EOF
```

1.1.1 Installation Examples (`real-int32` and `real-int32-debug`)

1. Download the installation script

```
curl -O \
https://raw.githubusercontent.com/firedrakeproject/firedrake/master/scripts/firedrake-install
```

2. Enable PETSc's debug option (optional)

```
DEBUG='-debug'
sed -i.bak -e 's/\(--with-debugging=\)0/\11/g' firedrake-install
```

3. Update the package of the system

```
sudo apt-get update
sudo apt-get install pkg-config # for p4est
```

4. Install

```
PETSC_CONFIGURE_OPTIONS=" \
    --download-fftw --download-mmg \
    --download-p4est --download-parmmg --download-triangle \
    --download-tetgen --download-ctetgen --download-hpddm --download-libpng \
    --download-slepc  --download-pragmatic" \
python3 firedrake-install --disable-ssh \
    --documentation-dependencies \
    --venv-name $HOME/firedrake/real-int32$DEBUG
```

1.1.2 Installation Examples (`complex-int64` and `complex-int64-debug`)

1. Download the installation script

```
curl -O \
https://raw.githubusercontent.com/firedrakeproject/firedrake/master/scripts/firedrake-install
```

2. Enable PETSc's debug option (optional)

```
DEBUG='-debug'
sed -i.bak -e 's/\(--with-debugging=\)0/\11/g' firedrake-install
```

3. Update the package of the system

```
sudo apt-get update
sudo apt-get install pkg-config
```

4. Install

```
PETSC_CONFIGURE_OPTIONS=" \
    --download-fftw --download-mmg \
    --download-p4est --download-parmmg --download-triangle \
    --download-tetgen --download-ctetgen --download-hpddm --download-libpng \
    --download-slepc --download-scalapack --download-mumps" \
python3 firedrake-install --disable-ssh \
    --documentation-dependencies  \
    --petsc-int-type int64 --complex \
    --venv-name $HOME/firedrake/complex-int64$DEBUG
```

Remark: `pragmatic` cannot be used with `int64`

1.1.3 Installation Example with MKL

1. Install mkl

   a. Add repo of mkl

   ```
   wget -O- https://apt.repos.intel.com/intel-gpg-keys/GPG-PUB-KEY-INTEL-SW-PRODUCTS.PUB \
   | gpg --dearmor | sudo tee /usr/share/keyrings/oneapi-archive-keyring.gpg > /dev/null

   echo "deb [signed-by=/usr/share/keyrings/oneapi-archive-keyring.gpg] \
   https://apt.repos.intel.com/oneapi all main" \
   | sudo tee /etc/apt/sources.list.d/oneAPI.list

   sudo apt update
   ```

   b. Install libs and headers of MKL

   ```
   # sudo apt install intel-basekit
   sudo apt install intel-oneapi-mkl
   sudo apt install intel-oneapi-mkl-devel
   ```

2. Update the packages of the system

   ```
   sudo apt-get update
   sudo apt-get install pkg-config # for p4est
   ```

3. Download the installation script and enable the debug option if necessary

   ```
   curl -O \
   https://raw.githubusercontent.com/firedrakeproject/firedrake/master/scripts/firedrake-install
   ```

   ```
   sed -i.bak -e 's/\(--with-debugging=\)0/\11/g' -e 's/\({0}\/lib\)/\1\/intel64/g' \
       -e 's/\(.*\)\(--C\)\(FLAGS=-I{}\/include\)\(.*\)/\1\2\3\4\n\1\2XX\3\4/' \
       firedrake-install
   ```

   a. `'s/\(--with-debugging=\)0/\11/g'` for petsc debug

   b. `'s/\({0}\/lib\)/\1\/intel64/g'` for mkl lib

   c. `'s/\(.*\)\(--C\)\(FLAGS=-I{}\/include\)\(.*\)/\1\2\3\4\n\1\2XX\3\4/'` for hpddm with mkl

4. Install Firedrake `real-int32`

```
time PETSC_CONFIGURE_OPTIONS="--download-fftw --download-mmg \
        --download-p4est --download-parmmg --download-triangle \
        --download-tetgen --download-ctetgen --download-hpddm --download-libpng \
        --download-slepc  --download-pragmatic \
        --with-mkl_pardiso-dir=/opt/intel/oneapi/mkl/latest \
        --with-mkl_cpardiso-dir=/opt/intel/oneapi/mkl/latest" \
python3 firedrake-install --disable-ssh --documentation-dependencies \
    --with-blas=/opt/intel/oneapi/mkl/latest \
    --venv-name firedrake/real-int32-mkl-debug
```

5. Fix the error on `mkl_cpardiso`

   If you run the following test, there will be an error:

```
$ cd petsc/src/binding/petsc4py/demo/kspsolve
$ python test_mat_ksp.py -pc_type lu -pc_factor_mat_solver_type mkl_cpardiso -ksp_view
Intel MKL FATAL ERROR: Cannot load symbol MKLMPI_Get_wrappers.
```

   a. Patch `petsc4py`

```
$ git diff
diff --git a/src/binding/petsc4py/conf/confpetsc.py
 ↪  b/src/binding/petsc4py/conf/confpetsc.py
index 5801b146ff..b00fab2d32 100644
--- a/src/binding/petsc4py/conf/confpetsc.py
+++ b/src/binding/petsc4py/conf/confpetsc.py
@@ -319,6 +319,11 @@ class PetscConfig:
         self._configure_ext(extension, petsc_inc, preppend=True)
         self._configure_ext(extension, petsc_lib)

+        blas_lib = flaglist(self['BLASLAPACK_LIB'])
+        blas_inc = flaglist(self['BLASLAPACK_INCLUDE'])
+        self._configure_ext(extension, blas_inc, preppend=True)
+        self._configure_ext(extension, blas_lib)
+
     def configure_compiler(self, compiler):
         if compiler.compiler_type != 'unix': return
         getenv = os.environ.get
```

   The value of BLASLAPACK_LIB is

```
BLASLAPACK_LIB="-Wl,-rpath,/opt/intel/oneapi/mkl/latest/lib/intel64 \
    -L/opt/intel/oneapi/mkl/latest/lib/intel64 \
    -lmkl_intel_lp64 -lmkl_core -lmkl_gnu_thread \
    -lmkl_blacs_intelmpi_lp64 -lgomp -ldl -lpthread"
```

   The other way to fix this is modifing the file `firedrake-install` by adding the following content to `blas["LDFLAGS"]`

```
blas["LDFLAGS"] = "-Wl,-rpath,/opt/intel/oneapi/mkl/latest/lib/intel64 \
    -L/opt/intel/oneapi/mkl/latest/lib/intel64 \
    -lmkl_intel_lp64 -lmkl_core -lmkl_gnu_thread \
    -lmkl_blacs_intelmpi_lp64 -lgomp -ldl -lpthread"
```

   b. Recompile and install petsc4py (in the activated Firedrake environment)

```
export PETSC_DIR=$(readlink -f $(dirname `which python`)/../src/petsc)
export PETSC_ARCH=default
cd $PETSC_DIR/src/binding/petsc4py
make clean
python -m pip install --no-build-isolation --no-binary mpi4py,randomgen,islpy,numpy \
    --no-deps -vvv --ignore-installed .
```

6. Install `slepc4py`

```
export SLEPC_DIR="$(find $PETSC_DIR/$PETSC_ARCH/externalpackages \
                    -maxdepth 1 -name '*slepc*')"
python -m pip install --no-build-isolation --no-binary mpi4py,randomgen,islpy,numpy \
    --no-deps -vvv --ignore-installed $SLEPC_DIR/src/binding/slepc4py
```

The complex version complex-int32 can be installed using the following command. If you encounter the same error of solver `mkl_cpardiso`, you can fix it by using the same method as before.

```
PETSC_CONFIGURE_OPTIONS=" \
    --download-fftw --download-mmg --download-pragmatic \
    --download-p4est --download-parmmg --download-triangle \
    --download-tetgen --download-ctetgen --download-hpddm --download-libpng \
    --download-slepc --download-scalapack --download-mumps \
    --with-mkl_pardiso-dir=/opt/intel/oneapi/mkl/latest \
    --with-mkl_cpardiso-dir=/opt/intel/oneapi/mkl/latest" \
python3 firedrake-install --disable-ssh \
    --documentation-dependencies  \
    --with-blas=/opt/intel/oneapi/mkl/latest --complex \
    --venv-name firedrake/complex-int32-mkl-debug
```

1.1.4 Some notes on petsc

PETSc with X

1. Install `libx11-dev`

```
sudo apt install libx11-dev
```

2. Add `--with-x=1` to `PETSC_CONFIGURE_OPTIONS`, and then follow the installation command of the previous section.

Add petsc bin to path We can define command `add-petsc-bin`. Executing it in activated firedrake env will add the petsc/bin to PATH

```
alias add-petsc-bin='export \
    PATH=$PATH:$(dirname $(which python))/../src/petsc/lib/petsc/bin:$(\
    dirname $(which python))/../src/petsc/default/bin'

alias firedrake-mkl="export OMP_NUM_THREADS=1 && \
    source ~/firedrake/real-int32-mkl-debug/bin/activate && add-petsc-bin"
```

Download package for petsc Sometimes, some of the packages that petsc depends on cannot be downloaded automatically. We can add the option

```
--with-packages-download-dir=<path/to/petsc/packages>
```

to obtain the list of required packages, and then download these packages manually and put them into the path. Afterwards, configure it again with the above option.

The following python script can be used to download multiple packages. Please modify the corresponding commands according to your needs.

```
packages = {
# "scalapack": ['git://https://github.com/Reference-ScaLAPACK/scalapack',
#
 ↪  'https://github.com/Reference-ScaLAPACK/scalapack/archive/5bad7487f496c81119233464 0ce4d3fc5f88144b.tar.gz'],
"pastix": ['http://ftp.mcs.anl.gov/pub/petsc/externalpackages/pastix_5.2.3.tar.bz2'],
}
fail = {}
for name, paths in packages.items():
    print(name)
    flag = False
    for path in paths:
        print(f'try path: {path}')
        if path.startswith('git'):
            ret = os.system(f'git clone {path[6:]}')
        else:
            ret = os.system(f'curl -L -x socks5h://localhost:5000 -O {path}')
        if ret == 0:
            flag = True
            break

    if flag == False:
        fail[name] = paths
        print(f'Fail to download {name}: {paths}')

print('packages failed to download:')
print(fail)
```

1.1.5 Test

```
source firedrake/bin/activate
cd $VIRTUAL_ENV/src/firedrake
pytest tests/regression/ -k "poisson_strong or stokes_mini or dg_advection"
```

1.1.6 Install Jupyter-lab

1. Install `jupyterlab`

   ```
   python3 -m pip install jupyterlab
   ```

   Maybe you need add `$HOME/.local/bin` to environment variable `PATH`:

   ```
   export PATH=$PATH:$HOME/.local/bin
   ```

2. Configure jupyterlab

   Generate config file:

```
jupyter notebook --generate-config
```

Set `use_redirect_file` to `False` in file `~/.jupyter/jupyter_notebook_config.py`

```
c.NotebookApp.use_redirect_file = False
```

3. Configure Browser

   In wsl-ubuntu, configure the browser like this:

   ```
   export BROWSER="/path/to/chrome/or/firefox"
   ```

   An example of chrome:

   ```
   export BROWSER='/mnt/c/Program Files/Google/Chrome/Application/chrome.exe'
   ```

   Now, you can type `jupyter-lab` to start jupyter. You will see jupyter in browser.

4. Configure kernels

   1. Activate env:

      ```
      $ source /your/env/path/activate
      ```

   2. Add kernels:

      ```
      (your-venv)$ ipython kernel install --name "local-venv" --user
      ```

      Now you need check the python path in `kernel.json`. Make sure it is the python in your env. Otherwise, correct it.

   3. Add environment variables to `kernel.json`:

      Ref: https://jupyter-client.readthedocs.io/en/stable/kernels.html

      An exmaple of `kernel.json`:

      ```
      {
       "argv": [
        "/home/yzz/firedrake/real-int32-debug/bin/python",
        "-m",
        "ipykernel_launcher",
        "-f",
        "{connection_file}"
       ],
       "env": {
        "OMP_NUM_THREADS": "1",
        "PATH": "/home/yzz/firedrake/real-int32-debug/bin:${PATH}"
       },
       "display_name": "firedrake-real-int32",
       "language": "python",
       "metadata": {
        "debugger": true
       }
      }
      ```

1.1.7 Update

Generally, you can simply run `firedrake-update` in the activated environment to update firedrake.

If you want to rebuild PETSc (i.e., using the `--rebuild` option) and you have used `PETSC_CONFIGURE_OPTIONS` and `--with-blas` during the installation, you also need to use these two options when updating.

In addition, if you installed MKL using the aforementioned method, you may need to modify the firedrake-update script.

The example for `firedrake/complex-int32-mkl-debug` is as follows:

1. Modify `firedrake-update`

```
sed -i.bak -e 's/\(--with-debugging=\)0/\11/g' -e 's/\({0}\/lib\)/\1\/intel64/g' \
    -e 's/\(.*\)\(--C\)\(FLAGS=-I{}\/include\)\(.*\)/\1\2\3\4\n\1\2XX\3\4/' \
    firedrake-update
```

2. Update

```
PETSC_CONFIGURE_OPTIONS=" \
    --download-fftw --download-mmg --download-pragmatic \
    --download-p4est --download-parmmg --download-triangle \
    --download-tetgen --download-ctetgen --download-hpddm --download-libpng \
    --download-slepc --download-scalapack --download-mumps \
    --with-mkl_pardiso-dir=/opt/intel/oneapi/mkl/latest \
    --with-mkl_cpardiso-dir=/opt/intel/oneapi/mkl/latest" \
firedrake-update --rebuild --no-update-script --with-blas=/opt/intel/oneapi/mkl/latest
```

1.2 Windows

Install WSL (Windows Subsystem for Linux) on Windows (the system installed is `Ubuntu` by default) and then install Firedrake as before.

1.2.1 Install WSL

https://docs.microsoft.com/zh-cn/windows/wsl/install

1.2.2 Install Firedrake

Follow the installation method for Ubuntu.

1.3 MacOS

First, install Homebrew (https://brew.sh/), and then use Homebrew to install python3. After that, install Firedrake directly, similar to Ubuntu.

1.4 Linux Server

If the server cannot access the network, please refer to the next section: **Installation without Network**. The method is based on the following method.

The Firedrake team provides a way to install Firedrake based on Spack, a package manager for HPC.

Ref: https://github.com/firedrakeproject/firedrake-spack

1. Download spack

```
mkdir -p $HOME/opt
cd $HOME/opt && \
git clone -c feature.manyFiles=true https://github.com/lrtfm/spack.git && \
pushd spack
git checkout lrtfm/develop
popd
source $HOME/opt/spack/share/spack/setup-env.sh
```

Remark 1: Add the following command to the file $HOME/.bashrc to add shell support for spack.

```
source $HOME/opt/spack/share/spack/setup-env.sh
```

Remark 2: On some workstations, the content of the /tmp directory may not have execution permissions. You need to change the spack build directory as follows.

```
mkdir -p $HOME/.spack && \
cat > $HOME/.spack/config.yaml <<EOF
config:
  build_stage:
    - \$user_cache_path/stage
EOF
```

2. Download firedrake-spack

```
cd $HOME/opt && \
git clone https://github.com/lrtfm/firedrake-spack.git && \
pushd firedrake-spack && \
git checkout lrtfm/air-gapped-install && \
popd
```

Note: The current version of petsc in firedrakeproject will break when using some compilers: https://lists.mcs.anl.gov/pipermail/petsc-users/2023-April/048482.html. Patch has been added branch `lrtfm/air-gapped-install` of `firedrake-spack`.

3. Create spack env and add packages

- complex-int32

  a. Create spack env

  ```
  cd $HOME/opt && \
  FIREDRAKE_ENV_NAME=firedrake-complex-int32 && \
  spack env create -d $FIREDRAKE_ENV_NAME && \
  spack env activate -p $FIREDRAKE_ENV_NAME && \
  spack -e $SPACK_ENV config add concretizer:unify:true
  ```

  b. Add firedrake repo

  We add the firedrake repo to the created space env

```
cd $HOME/opt && \
spack repo add firedrake-spack
```

c. Add packages

```
spack add py-firedrake@develop%gcc +complex ^mpich ^openblas ^slepc+hpddm \
    ^petsc+libpng+libyaml+parmmg+mmg+hpddm+tetgen+valgrind \
    ^hypre+superlu-dist ^vtk@9.0.3 && \
spack add gmsh py-meshio py-tqdm py-pyyaml py-memory-profiler
```

- real-int32

a. Create env

```
cd $HOME/opt && \
FIREDRAKE_ENV_NAME=firedrake-real-int32 && \
spack env create -d $FIREDRAKE_ENV_NAME && \
spack env activate -p $FIREDRAKE_ENV_NAME && \
spack -e $SPACK_ENV config add concretizer:unify:true
```

b. Add firedrake repo

```
cd $HOME/opt && \
spack repo add firedrake-spack
```

c. Add packages

Copy the follwing command into bash will raise error. Please copy line by line.

```
spack add py-firedrake@develop%gcc ^mpich ^openblas ^slepc+hpddm \
    ^petsc+libpng+libyaml+parmmg+mmg+hpddm+tetgen+valgrind \
    ^hypre+superlu-dist ^vtk@9.0.3 && \
spack add gmsh py-meshio py-tqdm py-pyyaml py-memory-profiler
```

Remark 3: Installing `vtk@8.x.x` and `vtk@9.2.2`(on some hosts) in spack will fail. We use `vtk@9.0.3`.(2023-04-30)

4. Make some packages as develop

This step can be skiped. With this step, we can update firedrake easily in spack.

```
spack develop py-firedrake@develop && \
spack develop libsupermesh@develop && \
spack develop petsc@develop && \
spack develop slepc@develop && \
spack develop py-fiat@develop && \
spack develop py-finat@develop && \
spack develop py-islpy@develop && \
spack develop py-petsc4py@develop && \
spack develop py-slepc4py@develop && \
spack develop py-pyadjoint@develop && \
spack develop py-pyop2@develop && \
spack develop py-coffee@develop && \
spack develop py-loopy@develop && \
spack develop py-cgen@develop && \
spack develop py-codepy@develop && \
```

```
spack develop py-genpy@develop && \
spack develop py-tsfc@develop && \
spack develop py-ufl@develop && \
spack develop chaco@petsc
```

Remark: We do not need the following package when install `int64` version:

```
spack develop chaco@petsc
```

5. Concretize and install

```
spack concretize -f 2>&1 | tee $SPACK_ENV/spack-firedrake-develop.log && \
    time spack install --fail-fast --show-log-on-error \
        --log-file $SPACK_ENV/spack-firedrake-install.log --log-format cdash
```

1.4.1 Docker

1. firedrake team:

   https://hub.docker.com/u/firedrakeproject.

2. lrtfm/firedrake:

   https://hub.docker.com/r/lrtfm/firedrake

TODO: Trimming the Docker image The Docker image is too large, so we can consider deleting some unnecessary files.

```
firedrake=$HOME/firedrake
rm -rf $HOME/.cache/pip
find $firedrake -name ".git" | xargs rm -rf
find $firedrake -name "*.o" | xargs rm

rm -rf $firedrake/src/{libspatialindex,libsupermesh}

rm -rf $firedrake/src/{petsc,slepc}/src

find $firedrake -name "doc" | xargs rm -rf
find $firedrake -name "docs" | xargs rm -rf
```

```
docker export
docker import
```

2 Installation without Network

If you need to install Firedrake on some HPC without internet access, you can use the source mirror feature of spack. A mirror is a URL that points to a directory, either on the local filesystem or on some server, containing tarballs for all of Spack's packages.

Assume the local host can access the network (github, etc.).

If the login node can access the network, the operations on the local host can be executed on the login node. Generally, HPC uses shared storage, so there is no need to arhive and upload the downloaded packages.

In the following, we will install spack and firedrake in directory `$HOME/opt`.

Note that the multi-line commands are connected by "`&& \`". You can copy and paste the multi-line command blocks into the terminal and run it.

Reference:

1. spack install:
   - https://spack.readthedocs.io/en/latest/getting_started.html#installation
2. spack mirror:
   - https://spack.readthedocs.io/en/latest/bootstrapping.html#creating-a-mirror-for-air-gapped-systems
   - https://spack.readthedocs.io/en/latest/mirrors.html#mirror-environment
   - https://spack.readthedocs.io/en/latest/mirrors.html#mirror-files
3. firedrake spack:
   - https://github.com/firedrakeproject/firedrake-spack
   - https://hackmd.io/@TzVnFeL0TMCb3FaAi9qYBA/ByaRskMQ5

2.1 Local host (with internet access)

2.1.1 Create installation directory

```
mkdir -p $HOME/opt
```

2.1.2 Clone spack

```
cd $HOME/opt && \
git clone -c feature.manyFiles=true https://github.com/lrtfm/spack.git && \
pushd spack && \
git checkout lrtfm/develop && \
popd && \
source $HOME/opt/spack/share/spack/setup-env.sh
```

Remark 1: Here, I clone spack from `https://github.com/lrtfm/spack.git`, a fork of spack, and use the branch `lrtfm/develop`, which may have some patchs I added. You can clone spack from the offical source `https://github.com/spack/spack.git`.

Remark 2: Add the following command to `$HOME/.bashrc` to enable the shell support of spack.

```
source $HOME/opt/spack/share/spack/setup-env.sh
```

2.1.3 Create mirror for bootstrap

```
spack bootstrap mirror --binary-packages $HOME/opt/bootstrap
```

The output looks like:

```
==> Adding "clingo-bootstrap@spack+python %gcc target=x86_64" and dependencies to the mirror at
 ↪  /home/xyz/opt/bootstrap/bootstrap_cache

==> Adding "gnupg@2.3: %gcc target=x86_64" and dependencies to the mirror at
 ↪  /home/xyz/opt/bootstrap/bootstrap_cache
==> Adding "patchelf@0.13.1: %gcc target=x86_64" and dependencies to the mirror at
 ↪  /home/xyz/opt/bootstrap/bootstrap_cache
==> Adding "gnuconfig" and dependencies to the mirror at /home/xyz/opt/bootstrap/bootstrap_cache
==> Adding binary packages from
 ↪  "https://github.com/spack/spack-bootstrap-mirrors/releases/download/v0.4/bootstrap-buildcache.tar.gz"
 ↪  to the mirro
r at /home/xyz/opt/bootstrap/bootstrap_cache

To register the mirror on the platform where it's supposed to be used, move
 ↪  "/home/xyz/opt/bootstrap" to its final location and run the following
command(s):

  % spack bootstrap add --trust local-sources <final-path>/metadata/sources
  % spack bootstrap add --trust local-binaries <final-path>/metadata/binaries
```

2.1.4 Pack the spack source and the mirror of bootstrap

```
tar -czvf spack.tar.gz spack
tar -czvf bootstrap.tar.gz bootstrap
```

2.1.5 Clone firedrake-spack

```
cd $HOME/opt && \
git clone https://github.com/lrtfm/firedrake-spack.git && \
pushd firedrake-spack && \
git checkout lrtfm/air-gapped-install && \
popd
```

Remark 1: Here, I clone firedrake-spack from `https://github.com/lrtfm/firedrake-spack.git`, which have some patchs I added. You can clone firedrake-spack from the offical source `https://github.com/firedrakeproject/firedrake-spack.git`.

2.1.6 Pack the source of firedrake-spack

```
tar -czvf firedrake-spack.tar.gz firedrake-spack
```

2.1.7 Add repo to spack (TODO: may be run in an spack env which will be created below)

```
spack repo add firedrake-spack
```

2.1.8 Check the installation of spack

The command `spack info py-firedrake` should have the following output

```
$ spack info py-firedrake
PythonPackage:   py-firedrake
```

```
Description:
    Firedrake is an automated system for the portable solution of partial
    differential equations using the finite element method (FEM)

Homepage: https://firedrakeproject.org

Preferred version:
    develop    [git] https://github.com/firedrakeproject/firedrake.git on branch master

Safe versions:
    develop    [git] https://github.com/firedrakeproject/firedrake.git on branch master

Deprecated versions:
    None

Variants:
    Name [Default]               When    Allowed values    Description
    ========================    ====    ==============
 ↪  ============================================

    64-bit-indices [off]         --      on, off           Install PETSc using 64bit indices
    build_system [python_pip]    --      python_pip        Build systems supported by the package
    complex [off]                --      on, off           Install Firedrake in complex mode
    minimal-petsc [off]          --      on, off           Build PETSc with minimal packages for
     ↪  Firedrake
    slepc [off]                  --      on, off           Install SLEPc and slepc4py

Build Dependencies:
    eigen            mpi            py-cython  py-h5py        py-mpi4py    py-pip
     ↪  py-pyadjoint  py-scipy       py-sympy   py-vtk    slepc
    libspatialindex  petsc          py-fiat    py-islpy       py-numpy     py-pkgconfig  py-pyop2
     ↪  py-setuptools  py-tsfc   py-wheel
    libsupermesh     py-cachetools  py-finat   py-matplotlib  py-petsc4py  py-progress
     ↪  py-requests   py-slepc4py    py-ufl    python

Link Dependencies:
    eigen  libspatialindex  libsupermesh  mpi  petsc  python  slepc

Run Dependencies:
    eigen            petsc          py-finat       py-mpi4py    py-pip        py-pyop2
     ↪  py-scipy       py-tsfc  slepc
    libspatialindex  py-cachetools  py-h5py        py-nbval     py-pkgconfig  py-pytest
     ↪  py-setuptools  py-ufl
    libsupermesh     py-cython      py-islpy       py-numpy     py-progress   py-pytest-xdist
     ↪  py-slepc4py    py-vtk
    mpi              py-fiat        py-matplotlib  py-petsc4py  py-pyadjoint  py-requests
     ↪  py-sympy       python
```

Now, the contents of `$HOME/opt` should looks like this:

```
$ ls -lha
total 214M
drwxrwxr-x  4 z2yang z2yang  112 Oct 30 15:17 .
drwxrwxr-x  3 z2yang z2yang   47 Oct 30 15:02 ..
drwxrwxr-x  5 z2yang z2yang  204 Oct 30 15:16 firedrake-spack
-rw-rw-r--  1 z2yang z2yang 211K Oct 30 15:17 firedrake-spack.tar.gz
drwxrwxr-x 10 z2yang z2yang 4.0K Oct 30 15:17 spack
-rw-rw-r--  1 z2yang z2yang 214M Oct 30 15:15 spack.tar.gz
```

2.2 Remote host (compute nodes which do not have access to internet)

1. Installation commands should be run in compute nodes (Is this true?). In HPCs using `slurm`, you can use `srun` to start an interactive terminal:

   ```
   srun -p xahctest --pty --export=ALL -N 1 -n 64 --exclusive /bin/bash
   ```

   or use `salloc` first and then login to the nodes by using `ssh`:

   ```
   salloc -p xahctest -N 1 -n 4
   ```

2. You should add `slurm` as external package of spack on system using `slurm` to submit jobs:

   ```
   spack external find slurm
   ```

3. Requirements on compiler:

   1. As compilling `openblas@0.3.12` using `gcc@7.3.1` will result in error, we use `gcc@9.4.0`. Because Amazon Linux GCC 7.3.1 has the patch gcc-bug-87467, `spack` change the conflict rule for `openblas` when it is compiled by `gcc@7` spack-pr-3443. However, GCC 7.3.1 on some hosts do not have this patch, which will result in error.

   2. The current version of petsc in firedrakeproject will break when using some compilers: https://lists.mcs.anl.gov/pipermail/petsc-users/2023-April/048482.html. Patch has been added branch `lrtfm/air-gapped-install` of `firedrake-spack`.

2.2.1 Install spack

1. Create installation directory

   ```
   mkdir -p $HOME/opt
   ```

2. Upload files

   Upload `firedrake-spack.gz`, `spack.tar.gz`, and `bootstrap.tar.gz` to directory `$HOME/opt` on the server.

3. Unpack the files and install spack

   ```
   cd $HOME/opt && \
   tar -zxf spack.tar.gz && \
   tar -zxf bootstrap.tar.gz && \
   source $HOME/opt/spack/share/spack/setup-env.sh && \
   spack bootstrap add --trust local-sources $HOME/opt/bootstrap/metadata/sources && \
   spack bootstrap add --trust local-binaries $HOME/opt/bootstrap/metadata/binaries
   ```

   Remark 1: Add the following command to the file $HOME/.bashrc to add shell support for spack.

   ```
   source $HOME/opt/spack/share/spack/setup-env.sh
   ```

   Remark 2: On some workstations, the content of the /tmp directory may not have execution permissions. You need to change the spack build directory as follows.

```
mkdir -p $HOME/.spack && \
cat > $HOME/.spack/config.yaml <<EOF
config:
  build_stage:
    - \$user_cache_path/stage
EOF
```

4. Install the `firedrake-spack` repo

```
cd $HOME/opt && \
tar -zxf firedrake-spack.tar.gz && \
spack repo add firedrake-spack       # Can be run after the creation of the env
```

2.2.2 Create spack env to install `firedrake`

1. Create spack env

```
FIREDRAKE_ENV_NAME=firedrake-complex-int64 && \
spack env create -d $FIREDRAKE_ENV_NAME && \
spack env activate -p $FIREDRAKE_ENV_NAME && \
spack -e $SPACK_ENV config add concretizer:unify:true
```

2. Add packages to the env

   You can add or delete some packages here. We will take the `complex+int64` version as an example.

```
spack add python py-firedrake@develop%gcc +64-bit-indices+complex ^mpich ^openblas \
    ^petsc+mumps+scalapack+int64+complex+libyaml+parmmg+mmg ^llvm@12.0.1 \
    ^hypre+complex+int64+superlu-dist && \
spack add py-pygmsh py-meshio py-tqdm py-pyyaml
```

3. Run `spack concretize`

```
spack concretize -f 2>&1 | tee $SPACK_ENV/spack-firedrake-concretize.log
```

4. Check the directory `$SPACK_ENV`

```
$ ls -la $SPACK_ENV
total 620
drwxrwxr-x 3 z2yang z2yang    118 Oct 30 16:01 .
drwxrwxr-x 5 z2yang z2yang    147 Oct 30 15:33 ..
drwxrwxr-x 4 z2yang z2yang     89 Oct 30 16:01 .spack-env
-rw-rw-r-- 1 z2yang z2yang  54343 Oct 30 16:01 spack-firedrake-concretize.log
-rw-rw-r-- 1 z2yang z2yang 572917 Oct 30 16:01 spack.lock
-rw-rw-r-- 1 z2yang z2yang    457 Oct 30 16:01 spack.yaml
```

   We will need the `spack.lock` file to create mirror in local host.

2.2.3 Create mirror on local host

The following commands run on **local host**.

We will create a firedrake env on local host by using the file `spack.lock`. And then create mirror for the env. After that, we upload the mirror to remote host.

1. Download `spack.lock` from remote host to directory `$HOME/opt` on local host.

2. Create mirror (May take 10 mins)

```
cd $HOME/opt && \
spack env create -d firedrake-mirror-env spack.lock && \
spack env activate -p ./firedrake-mirror-env && \
time spack mirror create -a -d spack-firedrake-mirror 2>&1 | tee creat-mirror.logs
```

   The above command should have the following output:

```
==> Summary for mirror in file:///home/z2yang/z2yang/local/opt/spack-firedrake-mirror
==> Archive stats:
    0    already present
    244  added
    0    failed to fetch.

real    10m56.048s
user    1m1.559s
sys     0m13.604s
```

   If there are some fails failed to fetch, you can clean the cache first and then create the mirror

```
spack clean -ds && \
time spack mirror create -a -d spack-firedrake-mirror 2>&1 | tee creat-mirror.logs
```

3. Pack the mirror

```
tar -czvf spack-firedrake-mirror.tar.gz spack-firedrake-mirror
```

2.2.4 Add mirror

The following commands run on remote host

1. Upload mirror

   Upload `spack-firedrake-mirror.tar.gz` to directory `$HOME/opt` on the remote host.

2. Unpack the mirrors

```
cd $HOME/opt && \
tar -xzvf spack-firedrake-mirror.tar.gz
```

3. Add the mirror to spack

```
cat > $HOME/.spack/mirrors.yaml <<EOF
mirrors:
  local_filesystem: file://$HOME/opt/spack-firedrake-mirror
EOF
```

4. Check the mirror.

   The output of `spack mirror lsit` should looks like:

```
$ spack mirror list
local_filesystem    file://<your-home-path>/opt/spack-firedrake-mirror
spack-public        https://mirror.spack.io
```

2.2.5 Install Firedrake

1. Run `spack develop` to avoid some errors

```
spack develop py-firedrake@develop && \
spack develop libsupermesh@develop && \
spack develop petsc@develop && \
spack develop slepc@develop && \
spack develop py-fiat@develop && \
spack develop py-finat@develop && \
spack develop py-islpy@develop && \
spack develop py-petsc4py@develop && \
spack develop py-slepc4py@develop && \
spack develop py-pyadjoint@develop && \
spack develop py-pyop2@develop && \
spack develop py-coffee@develop && \
spack develop py-loopy@develop && \
spack develop py-cgen@develop && \
spack develop py-codepy@develop && \
spack develop py-genpy@develop && \
spack develop py-tsfc@develop && \
spack develop py-ufl@develop
```

Remark 1: The `int32` version needs the following command: `bash  spack develop chaco@petsc`

2. Install

   Run the following command to install (It will take 1-2 hours for the first time depending on the system). It may be failed. Good Luck!

```
spack concretize -f 2>&1 | tee $SPACK_ENV/spack-firedrake-develop.log && \
time spack install --fail-fast --show-log-on-error \
    --log-file $SPACK_ENV/spack-firedrake-install.log --log-format cdash
```

   The last lines of the output:

```
[+]
 ↪  /home/z2yang/z2yang/server2/opt/spack/opt/spack/linux-ubuntu22.04-cascadelake/gcc-11.3.0/py-firedrake-dev
==> Updating view at /home/z2yang/z2yang/server2/opt/firedrake-complex-int64/.spack-env/view

real    184m44.242s
user    836m56.348s
sys     97m42.901s
```

3. Deactivate the env

```
despacktivate
```

   Feel free to ignore the following warnings

```
$ despacktivate
==> Warning: Skipping reversal of unreversable operation<class
 ↪  'spack.util.environment.UnsetEnv'> PETSC_ARCH
==> Warning: Skipping reversal of unreversable operation<class
 ↪  'spack.util.environment.UnsetEnv'> PETSC_ARCH
==> Warning: Skipping reversal of unreversable operation<class
 ↪  'spack.util.environment.UnsetEnv'> PETSC_ARCH
==> Warning: Skipping reversal of unreversable operation<class
 ↪  'spack.util.environment.UnsetEnv'> PETSC_ARCH
```

2.2.6 Usage

1. Activate the env

```
cd $HOME/opt && \
spack env activate -p $FIREDRAKE_ENV_NAME
```

2. Test

```
cd $SPACK_ENV/py-firedrake && \
pytest tests/regression/ -k "poisson_strong or stokes_mini or dg_advection"
```

3 Linux Notes

1. Enable proxy through dynamic port forwarding in ssh (the sockets proxy port is 5000)

```
ssh -vv -N -D 5000 user@hostname
```

2. apt proxy

```
sudo apt -o Acquire::http::proxy="socks5h://127.0.0.1:5000" update
```

3. curl proxy

```
curl -x socks5h://localhost:5000 -O https://url/to/you/file
```

4 Try Firedrake on Colab

Colab is short for Colaboratory (which can be considered as an online version of Jupyter, allowing you to write and execute Python code in your browser).

FEM on Colab supports the installation of FEniCS, FEniCSx, Firedrake, NGSolve, gmsh on colab

4.1 Import package

4.1.1 Firedrake

About 3 minutes

```
try:
    import firedrake
except ImportError:
    !wget "https://fem-on-colab.github.io/releases/firedrake-install-real.sh" \
        -O "/tmp/firedrake-install.sh" && bash "/tmp/firedrake-install.sh"
    import firedrake
```

4.1.2 Gmsh

```
try:
    import gmsh
except ImportError:
    !wget "https://fem-on-colab.github.io/releases/gmsh-install.sh" \
        -O "/tmp/gmsh-install.sh" && bash "/tmp/gmsh-install.sh"
    import gmsh
```

4.2 Examples

https://colab.research.google.com/drive/1gM3zMWTskH7XyDi1yJL76BPFnOJjSdYh?usp=sharing

5 Ask for help

5.1 spack

1. https://spackpm.slack.com/
2. https://groups.google.com/g/spack

5.2 firedrake

Documentation:

- https://www.firedrakeproject.org/documentation.html

Github (issues and discussions):

- https://github.com/firedrakeproject/firedrake
- https://github.com/firedrakeproject/firedrake/issues
- https://github.com/firedrakeproject/firedrake/discussions

Slack:

- https://firedrakeproject.slack.com

Mail list:

- https://mailman.ic.ac.uk/mailman/listinfo/firedrake

6 Other FEM library/softwares

- FEniCSx
- NgSolve
- deal.II
- libMesh
- FreeFEM
- Dune