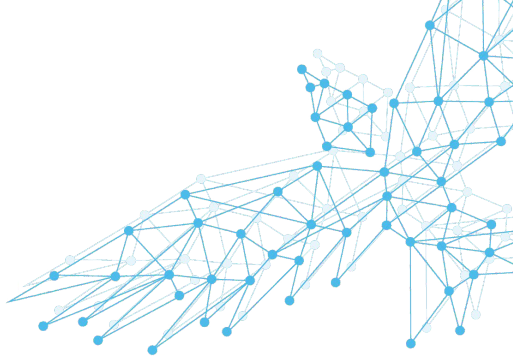
university of groningen / center for information technology

CIT Academy

Using your own source code on Peregrine

Outline



1. Transferring your software
2. Compiler environment
3. Compiling your software
4. Makefile

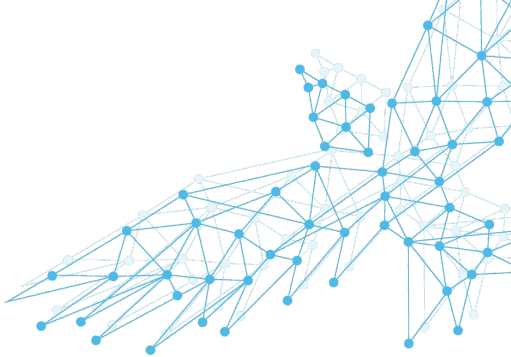
Transferring your software

Multiple approaches:

- ~~Use sftp/scp (e.g. MobaXterm file transfer)~~
- Use version control system, e.g. git

Unfortunately an explanation of git does not fit in current time slot

Compiler environment



- Compilers organised in toolchains, including:
  - MPI (Parallel library)
  - FFTW (Fourier transforms)
  - BLAS, LAPACK (linear algebra)
- Three versions available
  - foss: Free Open Source Software
    GNU compilers
    Open source!
  - intel: Intel compiler
    Sometimes better CPU optimization
  - PGI: Portland group compiler
    Supports OpenACC for GPU
    compiler only

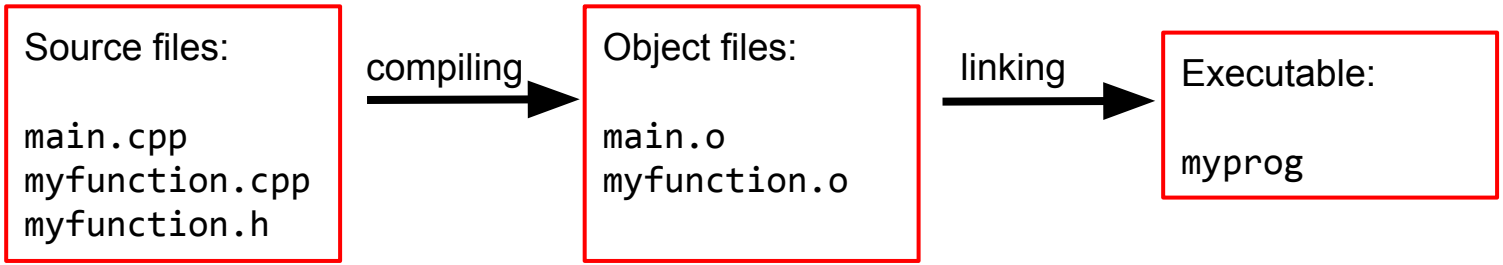
Common toolchain modules

- foss/2018a
- intel/2018a (2016a versions also still available)

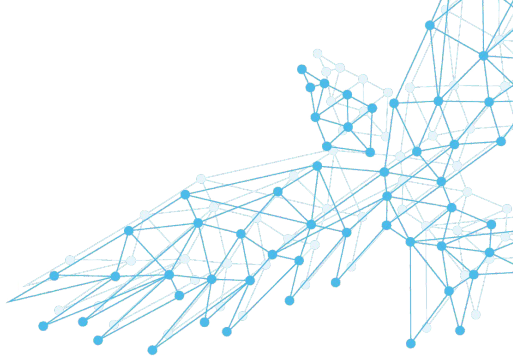
module load foss/2018a

Compiling from source

Two step process:



Command line:



- Two steps:
  - Compiling:

```
g++ -c -O2 myfunction.cpp
g++ -c -O2 main.cpp
```

  - Linking

```
g++ myfunction.o main.o -o myprog
```

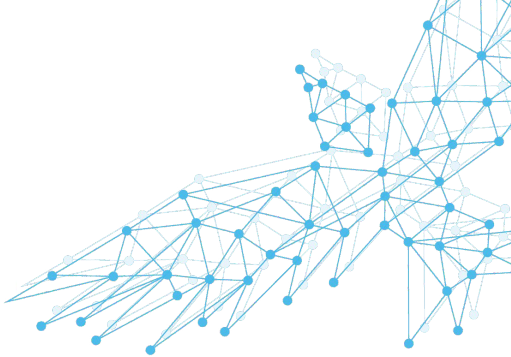
  - Flags:

`-c`: Skip linking

`-O2`: Optimization level

`-o`: Name of resulting output file

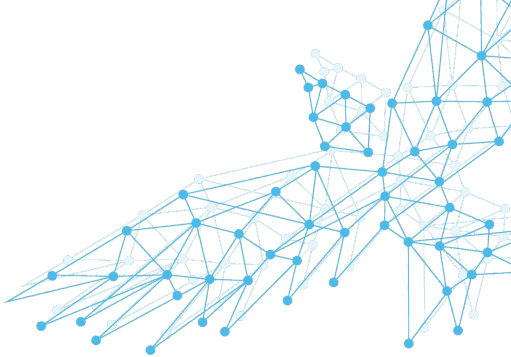
Command line:



Single line:

```
g++ -O2 main.cpp myfunction.cpp -o myprog
```

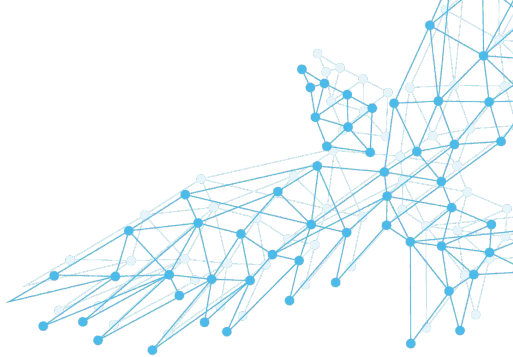
Makefiles



- Recipe for building your program
- Prevents typing a lot of commands
- Skips parts which do not need to be recompiled
- Tutorial:
  http://mrbook.org/blog/tutorials/make/

- Example provided in course material

Makefile



```
CXX=g++
CXXFLAGS=-c -O3 -Wall
LDFLAGS=
SOURCES=$(wildcard *.cpp)
OBJECTS=$(SOURCES:.cpp=.o)
EXECUTABLE=hello

all: $(SOURCES) $(EXECUTABLE)

$(EXECUTABLE): $(OBJECTS)
<  tab  >$(CXX) $(LDFLAGS) $(OBJECTS) -o $@

%.o: %.cpp
<  tab  >$(CXX) $(CXXFLAGS) $< -o $@

clean:
<  tab  >rm $(OBJECTS) $(EXECUTABLE)
```