Introduction to OpenMP

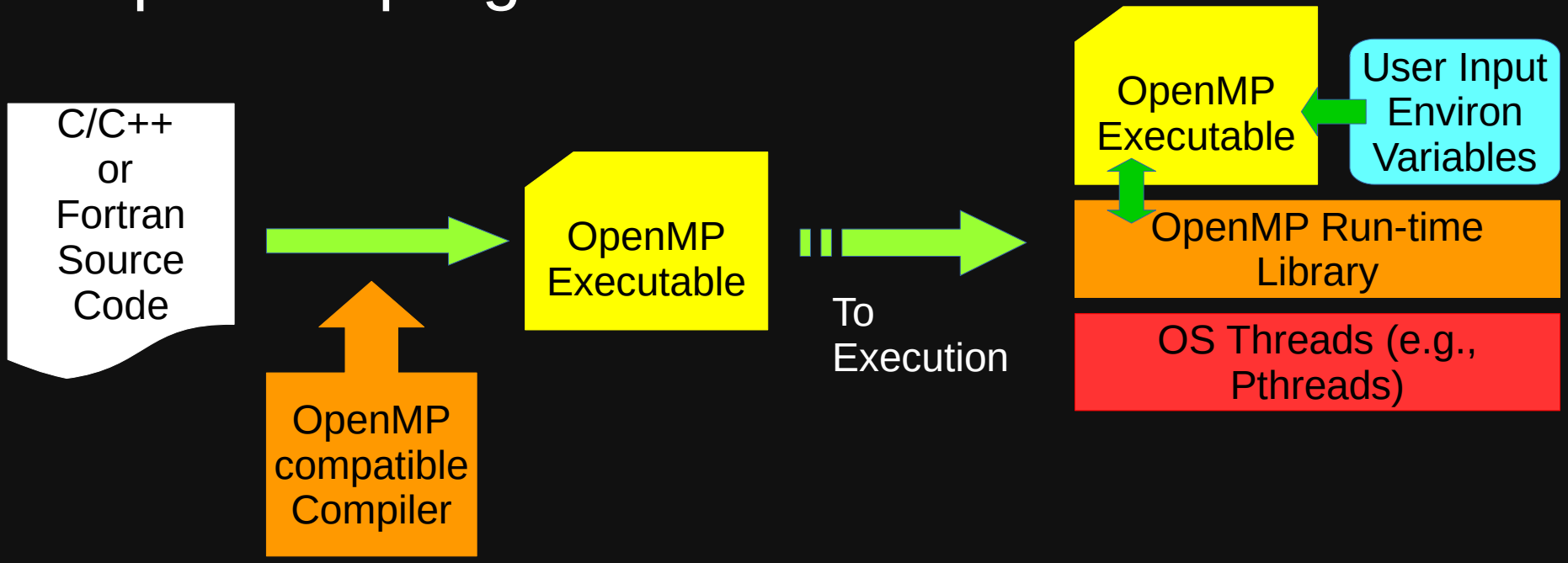
Wei Wang

What is OpenMP?

- Open specifications for Multi Processing
- OpenMP is an Application Program Interface (API) that may be used to explicitly direct multi-threaded, shared memory parallelism.
- Comprised of three primary API components:
  - Compiler Directives
    - for programming
  - Runtime Library Routines
    - to support parallel execution
  - Environment Variables
    - To control parallel execution

Compiling and Executing OpenMP Programs

- The general flow of compiling and executing OpenMP programs



OpenMP Is Not:

- for distributed memory parallel systems
- implemented identically across vendor and/or compilers
- Guaranteed to be efficient
- Required to check for parallel bugs, such as,
  - Data dependencies, data conflicts, race conditions, deadlocks
  - The programmer should ensure there are no such bugs
- Designed to handle parallel I/O. The programmer is responsible for synchronizing input and output.

The Goals of OpenMP

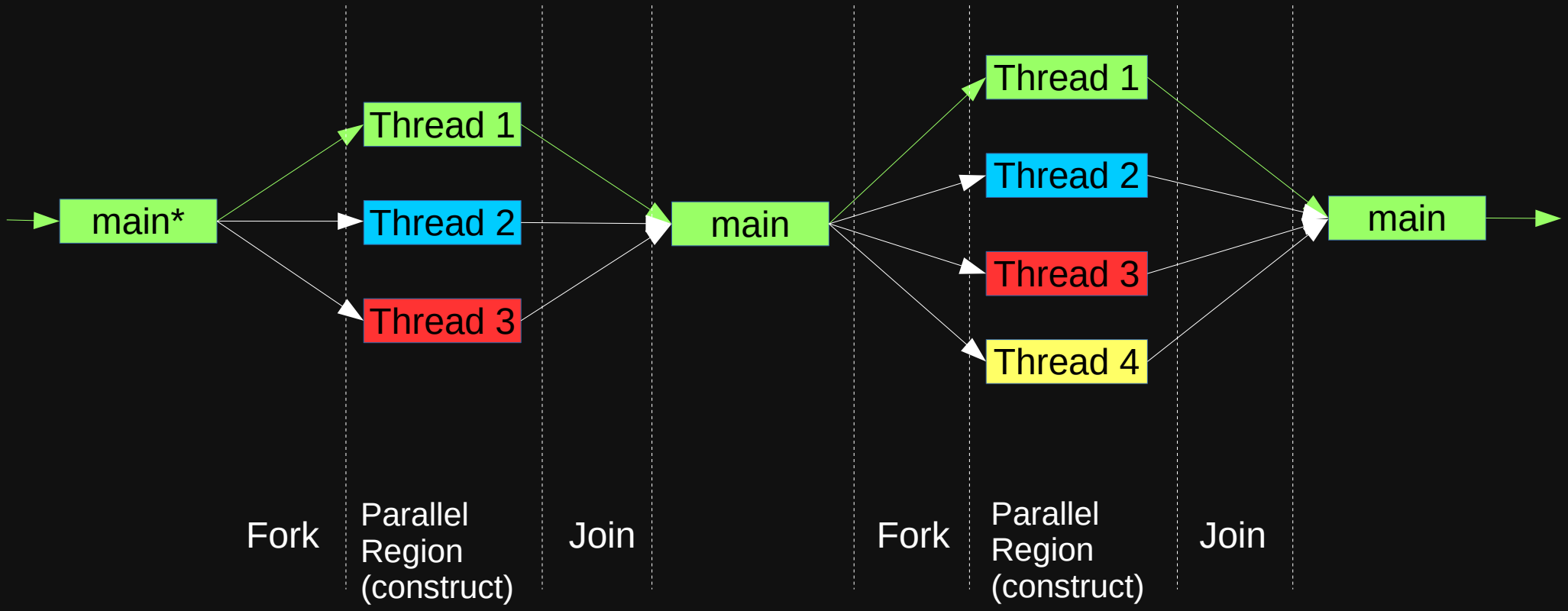
- Ease of use
  - Mostly targeting non-CS major scientists
  - Aims at converting sequential programs to parallel programs with minimal changes
  - Can handle simple decomposition and mapping automatically
- Standardization
  - Provides a standard for a variety of shared-memory architectures
- Portability
  - The API is specified for C/C++ and Fortran
  - Most major platforms have been implemented including Unix/Linux platforms and Windows

History of OpenMP

- First attempt at a standard was the draft for ANSI X3H5 in 1994. It was never adopted, largely due to waning interest as distributed memory machines became popular.
- However, not long after this, newer shared memory machine architectures started to become prevalent, and interest resumed.
- The OpenMP standard specification started in the spring of 1997, taking over where ANSI X3H5 had left off.
- Led by the OpenMP Architecture Review Board (ARB).
- Version 1.0, Oct 1997
- Version 4.5, Nov 2015

The Fork-Join Model

- OpenMP employs a Fork-Join Model:



*In implementation, main thread and thread1 are usually the same thread

Compiling a OpenMP Program

- GCC implements OpenMP standard
- To compile a C/C++ OpenMP program, use command line option “-fopenmp”
  - Example: gcc -fopenmp openmp_c_code.c -o openmp_executable
- To execute a OpenMP program, just run the executable normally
  - OpenMP will automatically create threads for parallel regions
  - By default, the number of threads is the same as the physical processors
  - Users can specify thread number with environment variable OMP_NUM_THREADS. For example, one can run an OpenMP executable with 4 threads using the following command in Bash:
    - OMP_NUM_THREADS=4 ./openmp_executable

Textbook and Auxiliary Materials

- Introduction to Parallel Computing, Chapter 7.10
- You can also use online tutorials:
  - https://hpc-tutorials.llnl.gov/openmp/
  - https://www.youtube.com/playlist?list=PLLbPZJxtMs4ZHSamRRYCtvowRS0qlwC-I
    (thanks to Farhan Tajwar Romit)
  - These tutorials may be slightly different from slides, because of the implementation differences between GCC and ICC. For exams, please use our slides as the standard in case of conflicts.