Related Programming Models
OpenACC

Wen-mei Hwu – University of Illinois at Urbana-Champaign
Objective

- To understand the OpenACC programming model
  - basic concepts and pragma types
  - simple examples
The OpenACC Application Programming Interface provides a set of

- compiler directives (pragmas)
- library routines and
- environment variables

that can be used to write data parallel FORTRAN, C and C++ programs that run on accelerator devices including GPUs and CPUs
OpenACC Pragmas

• In C and C++, the #pragma directive is the method to provide to the compiler information that is not specified in the standard language.
Simple Matrix-Matrix Multiplication in OpenACC

1. void computeAcc(float *P, const float *M, const float *N, int Mh, int Mw, int Nw)

2   {
3     #pragma acc parallel loop copyin(M[0:Mh*Mw])
4           copyin(N[0:Mw*Nw]) copyout(P[0:Mh*Nw])
5     for (int i=0; i<Mh; i++) {
6         #pragma acc loop
7         for (int j=0; j<Nw; j++) {
8             float sum = 0;
9             for (int k=0; k<Mw; k++) {
10                float a = M[i*Mw+k];
11                float b = N[k*Nw+j];
12                sum += a*b;
13            }  
14            P[i*Nw+j] = sum;
15        }
16    }
17}
Some Observations

• The code is almost identical to the sequential version, except for the two lines with #pragma at line 3 and line 5.

• OpenACC uses the compiler directive mechanism to extend the base language.
  • #pragma at line 3 tells the compiler to generate code for the ‘i’ loop at line 4 through 15 so that the loop iterations are executed at the first level of parallelism on the accelerator.
  • The copyin() clause and the copyout() clause specify how the matrix data should be transferred between the host and the accelerator.
  • The #pragma at line 5 instructs the compiler to map the inner ‘j’ loop to the second level of parallelism on the accelerator.
• OpenACC programmers can often start with writing a sequential version and then annotate their sequential program with OpenACC directives.
  • leave most of the details in generating a kernel, memory allocation, and data transfers to the OpenACC compiler.

• OpenACC code can be compiled by non-OpenACC compilers by ignoring the pragmas.
Frequently Encountered Issues

- Some OpenACC pragmas are hints to the OpenACC compiler, which may or may not be able to act accordingly
  - The performance of an OpenACC program depends heavily on the quality of the compiler.
  - It may be hard to figure out why the compiler cannot act according to your hints
  - Much less so for CUDA or OpenCL programs
Currently OpenACC does not allow synchronization across threads.
OpenACC Execution Model
TO LEARN MORE, FINISH CHAPTER 15

Heterogeneous Parallel Programming