INTRODUCTION TO OPENACC

Bharat Kumar, Andreas Herten

OpenACC
More Science, Less Programming

NVIDIA aws Linux Academy

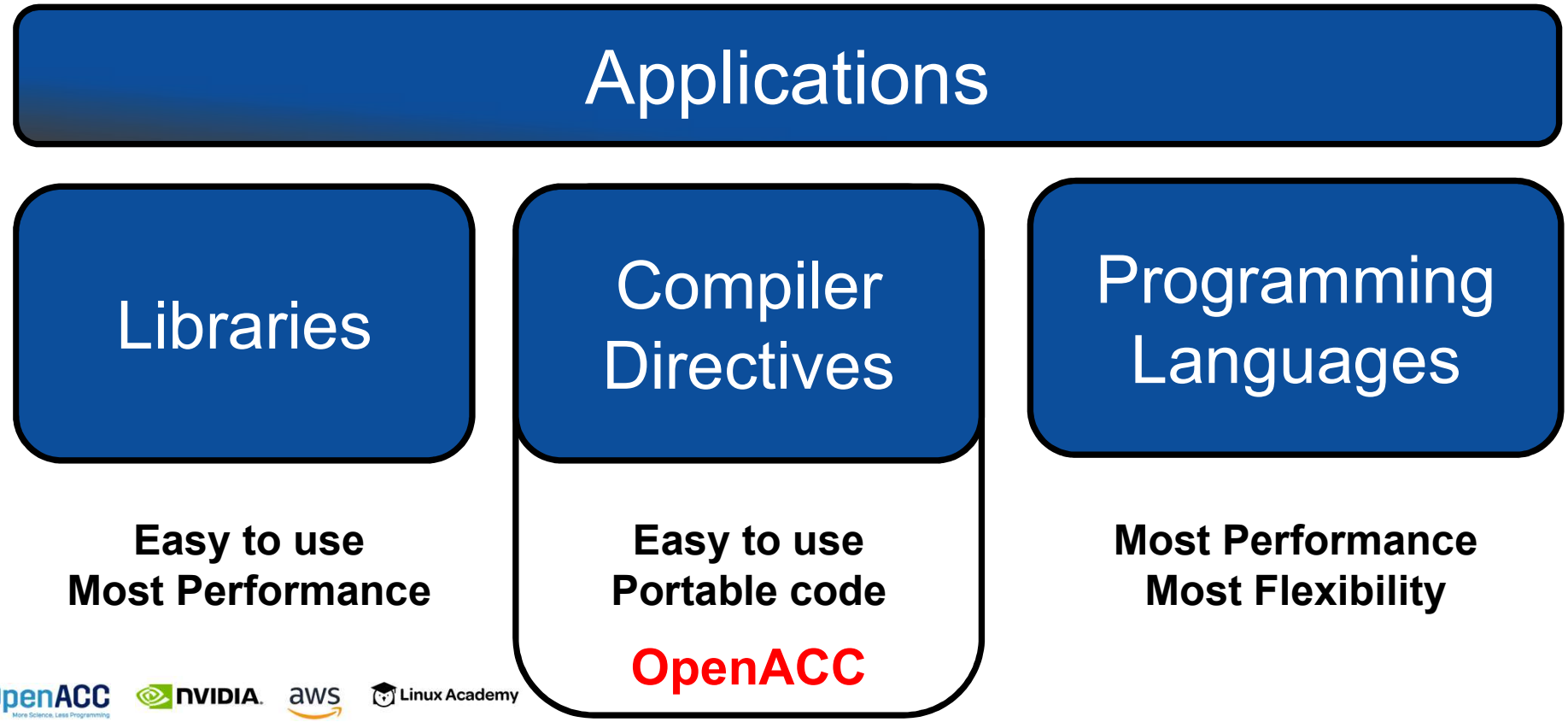
LECTURE 1 OUTLINE

Topics to be covered

- What is OpenACC and Why Should You Care?
- Profile-driven Development
- First Steps with OpenACC
- Week 1 Lab
- Where to Get Help

INTRODUCTION TO OPENACC

3 WAYS TO ACCELERATE APPLICATIONS



OpenACC NVIDIA aws Linux Academy

OPENACC IS…

a directives-based **parallel programming model** designed for **performance** and **portability**.

Add Simple Compiler Directive

```
main()
 {
  <serial code>
  #pragma acc kernels
  {
    <parallel code>
  }
}
```

OpenACC More Science, Less Programming · NVIDIA · aws · Linux Academy

OpenACC Directives

Manage Data Movement

Initiate Parallel Execution

Optimize Loop Mappings

```
#pragma acc data copyin(a,b) copyout(c)
{
  ...
  #pragma acc parallel
  {
  #pragma acc loop gang vector
      for (i = 0; i < n; ++i) {
          c[i] = a[i] + b[i];
          ...
      }
  }
  ...
}
```

OpenACC
Directives for Accelerators

- Incremental
- Single source
- Interoperable
- Performance portable
- CPU, GPU, Manycore

OpenACC More Science, Less Programming | NVIDIA | aws | Linux Academy

OPENACC

Incremental

- Maintain existing sequential code
- Add annotations to expose parallelism
- After verifying correctness, annotate more of the code

Single Source

- Rebuild the same code on multiple architectures
- Compiler determines how to parallelize for the desired machine
- Sequential code is maintained

Low Learning Curve

- OpenACC is meant to be easy to use, and easy to learn
- Programmer remains in familiar C, C++, or Fortran
- No reason to learn low-level details of the hardware.

OpenACC More Science, Less Programming NVIDIA aws Linux Academy

DIRECTIVE-BASED HPC PROGRAMMING

Who's Using OpenACC?

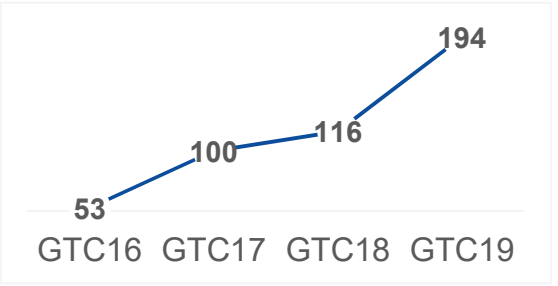
3 OF TOP 5 HPC APPS

Intersect360 RESEARCH

5 OF 13 CAAR CODES

summit

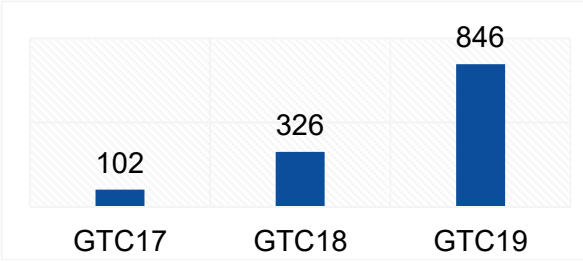
ACCELERATED APPS



725 TRAINED EXPERTS

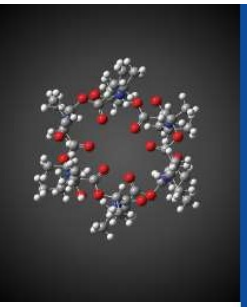
OpenACC HACKATHON
Advancing High Performance Computing

SLACK MEMBERS



160,000+ DOWNLOADS

PGI®
Community EDITION



GAUSSIAN 16

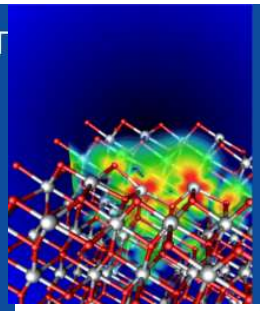
Mike Frisch, Ph.D
President and
CEO
Gaussian, Inc.

Using OpenACC allowed us to continue development of our fundamental algorithms and software capabilities simultaneously with the GPU-related work. In the end, we could use the same code base for SMP, cluster/network and GPU parallelism. PGI's compilers were essential to the success of our efforts.

ANSYS FLUENT

Sunil Saigal
Lead Software Developer
ANSYS Fluent

We've effectively used OpenACC for heterogeneous computing in ANSYS Fluent with impressive performance. We're now applying this work to more of our models and new platforms.



VASP

Prof. Georg Kresse
Computational Materials Physics
University of Vienna

For VASP, OpenACC is *the* way forward for GPU acceleration. Performance is similar and in some cases better than CUDA C, and OpenACC dramatically decreases GPU development and maintenance efforts. We're excited to collaborate with NVIDIA and PGI as an early adopter of CUDA Unified Memory.

COSMO

Dr. Oliver Fuhrer
Senior Scientist
Meteoswiss

OpenACC made it practical to develop for GPU-based hardware while retaining a single source for almost all the COSMO physics code.

E3SM

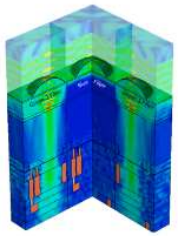
Mark A. Taylor
Multiphysics Applications
Sandia

The CAAR project provided us with early access to Summit hardware and access to PGI compiler experts. Both of these were critical to our success. PGI's OpenACC support remains the best available and is competitive with much more intrusive programming model approaches.

NUMECA FINE/Open

David Gutzwiller
Lead Software Developer
NUMECA

- Porting our unstructured C++ CFD solver FINE/Open to GPUs using OpenACC would have been impossible two or three years ago, but OpenACC has developed enough that we're now getting some really good results.



SYNOPSYS

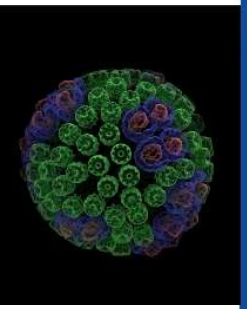
Dr. Lutz Schneider
Senior R&D Engineer
Synopsys Inc.

Using OpenACC, we've GPU-accelerated the Synopsys TCAD Sentaurus Device EMW simulator to speed up optical simulations of image sensors. GPUs are key to improving simulation throughput in the design of advanced image sensors.

MPAS-A

Richard Loft
Director, Technology Development
NCAR

Our team has been evaluating OpenACC as a pathway to performance portability for the Model for Prediction (MPAS) atmospheric model. Using this approach on the MPAS dynamical core, we have achieved performance on a single P100 GPU equivalent to 2.7 dual socketed Intel Xeon nodes on our new Cheyenne supercomputer.



VMD

John Stone
Senior Research Programmer
Beckham Institute
University of Illinois

Due to Amdahl's law, we need to port more parts of our code to the GPU if we're going to speed it up. But the sheer number of routines poses a challenge. OpenACC directives give us a low-cost approach to getting at least some speed-up out of these second-tier routines. In many cases it's completely sufficient because with the current algorithms, GPU performance is bandwidth-bound.

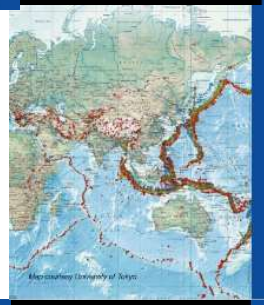
GTC

Zhihong Lin
Professor and Principal Investigator
UC Irvine

Using OpenACC our scientists were able to achieve the acceleration needed for integrated fusion simulation with a minimum investment of time and effort in learning to program GPUs.

OpenACC

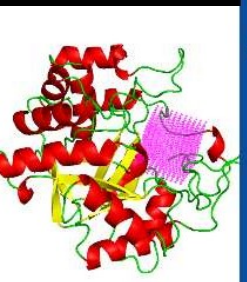
More Science, Less Programming



GAMERA

Takuma Yamaguchi, Kohei Fujita, Keiyoshi Ichimura, Miquel Hori, Lalith Wijerathne
The University of Tokyo

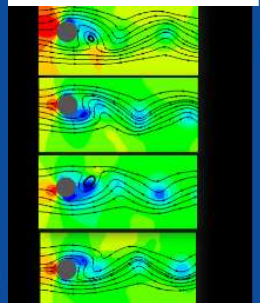
With OpenACC and a compute node based on NVIDIA's Tesla P100 GPU, we achieved more than a 14X speed up over a K Computer node running our earthquake disaster simulation code



SANJEEVINI

Abhilash Jayaraj
Project Scientist
Indian Institute of Technology
New Delhi

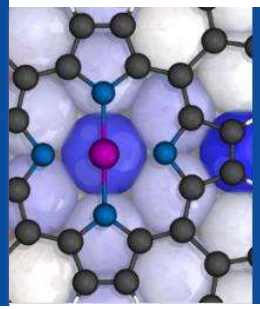
In an academic environment maintenance and speedup of existing codes is a tedious task. OpenACC provides a great platform for computational scientists to accomplish both tasks without involving a lot of efforts or manpower in speeding up the entire computational task.



IBM-CFD

Somnath Roy
Assistant Professor
Mechanical Engineering Department
Indian Institute of Technology Kharagpur

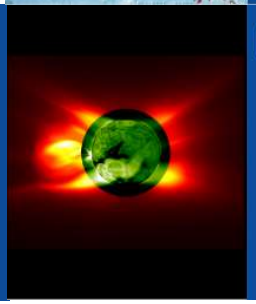
OpenACC can prove to be a handy tool for computational engineers and researchers to obtain fast solution of non linear dynamics problem. In immersed boundary incompressible CFD, we have obtained order of magnitude reduction in computing time by porting several components of our legacy codes to GPU. Especially the routines involving search algorithm and matrix solvers have been well-accelerated to improve the overall scalability of the code.



PWscf (Quantum ESPRESSO)

Filippo Spiga
Senior Contributor
Quantum ESPRESSO group

- CUDA Fortran gives us the full performance potential of the CUDA programming model and NVIDIA GPUs. While leveraging the potential of explicit data movement, ISO_C_BINDING directives give us productivity and source code maintainability. It's the best of both worlds.



MAS

Ronald M. Caplan
Computational Scientist
Predictive Science Inc.

Adding OpenACC into MAS has given us the ability to migrate medium-sized simulations from a multi node CPU cluster to a single multi-GPU server. The implementation yielded a portable single-source code for both CPU and GPU runs. Future work will add OpenACC to the remaining model features, enabling GPU accelerated realistic solar storm modeling.

OPENACC SYNTAX

OPENACC SYNTAX

Syntax for using OpenACC directives in code

C/C++

```
#pragma acc directive clauses
<code>
```

Fortran

```
!$acc directive clauses
<code>
```

- A ***pragma*** in C/C++ gives instructions to the compiler on how to compile the code. Compilers that do not understand a particular pragma can freely ignore it.
- A ***directive*** in Fortran is a specially formatted comment that likewise instructs the compiler in the compilation of the code and can be freely ignored.
- "***acc"*** informs the compiler that what will come is an OpenACC directive
- ***Directives*** are commands in OpenACC for altering our code.
- ***Clauses*** are specifiers or additions to directives.

EXAMPLE CODE

LAPLACE HEAT TRANSFER

Introduction to lab code - visual

We will observe a simple simulation of heat distributing across a metal plate.

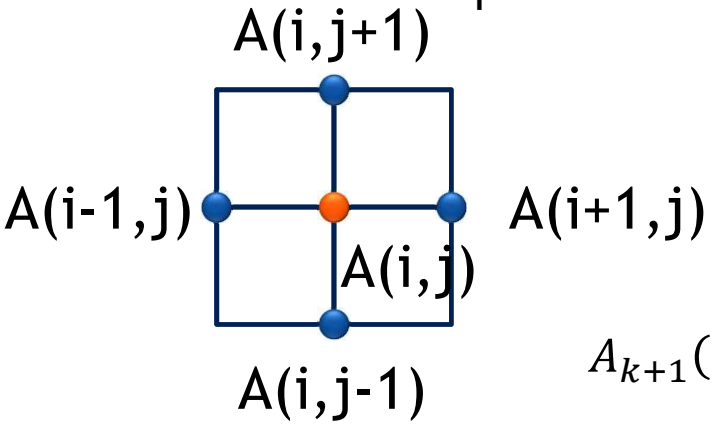
We will apply a consistent heat to the top of the plate.

Then, we will simulate the heat distributing across the plate.

OpenACC NVIDIA aws Linux Academy

EXAMPLE: JACOBI ITERATION

- Iteratively converges to correct value (e.g. Temperature), by computing new values at each point from the average of neighboring points.
- Common, useful algorithm
- Example: Solve Laplace equation in 2D: $\nabla^2 f(x, y) = 0$



$$A_{k+1}(i,j) = \frac{A_k(i-1,j) + A_k(i+1,j) + A_k(i,j-1) + A_k(i,j+1)}{4}$$

JACOBI ITERATION: C CODE

```
while ( err > tol && iter < iter_max ) {
  err=0.0;


  for( int j = 1; j < n-1; j++) {
    for(int i = 1; i < m-1; i++) {

      Anew[j][i] = 0.25 * (A[j][i+1] + A[j][i-1] +
                           A[j-1][i] + A[j+1][i]);

      err = max(err, abs(Anew[j][i] - A[j][i]));
    }
  }


  for( int j = 1; j < n-1; j++) {
    for( int i = 1; i < m-1; i++ ) {
      A[j][i] = Anew[j][i];
    }
  }

  iter++;
```

Iterate until converged

Iterate across matrix elements

Calculate new value from neighbors

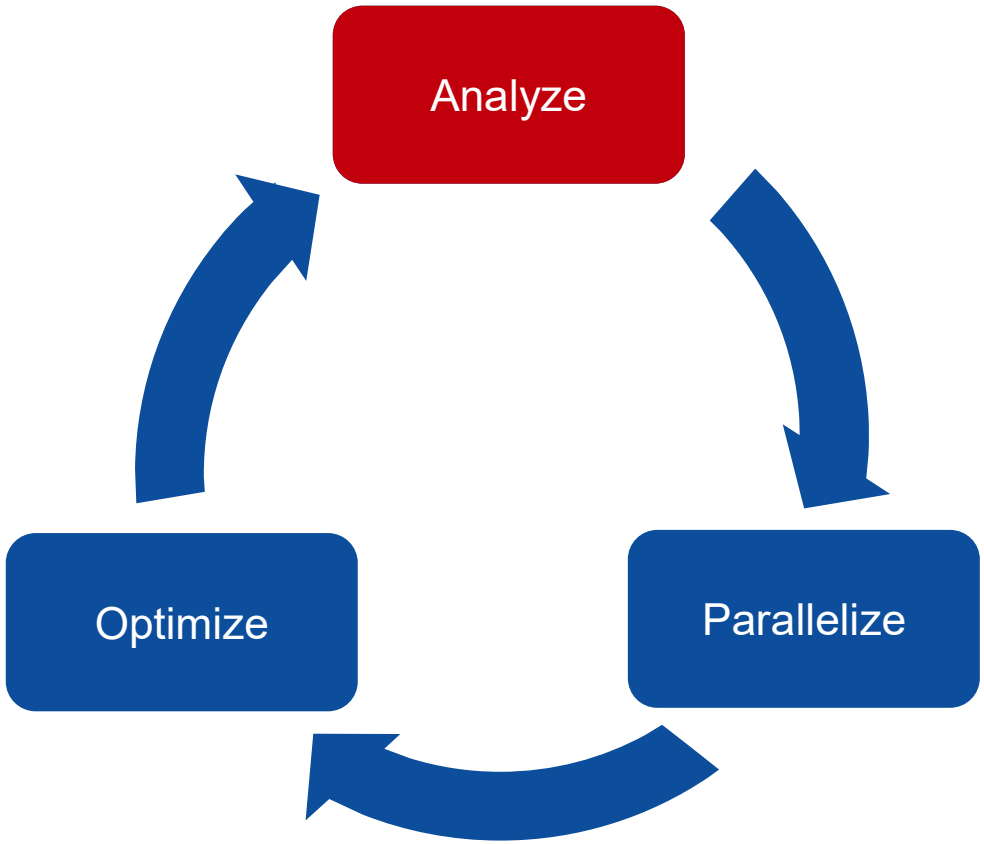
Compute max error for convergence

Swap input/output arrays

PROFILE-DRIVEN DEVELOPMENT

OPENACC DEVELOPMENT CYCLE

- **Analyze** your code to determine most likely places needing parallelization or optimization.
- **Parallelize** your code by starting with the most time consuming parts and check for correctness.
- **Optimize** your code to improve observed speed-up from parallelization.



PROFILING SEQUENTIAL CODE

Profile Your Code

Obtain detailed information about how the code ran.

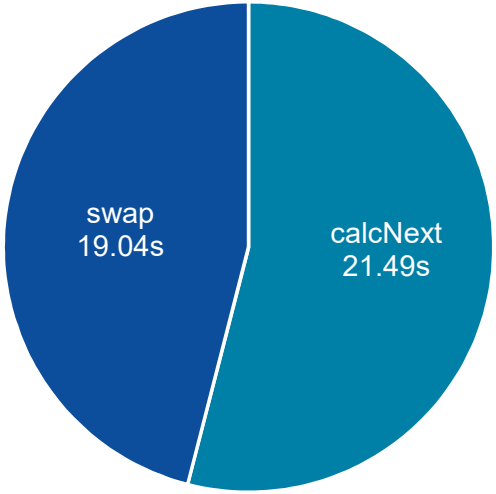
This can include information such as:

- Total runtime
- Runtime of individual routines
- Hardware counters

Identify the portions of code that took the longest to run. We want to focus on these “hotspots” when parallelizing.

Lab Code: Laplace Heat Transfer

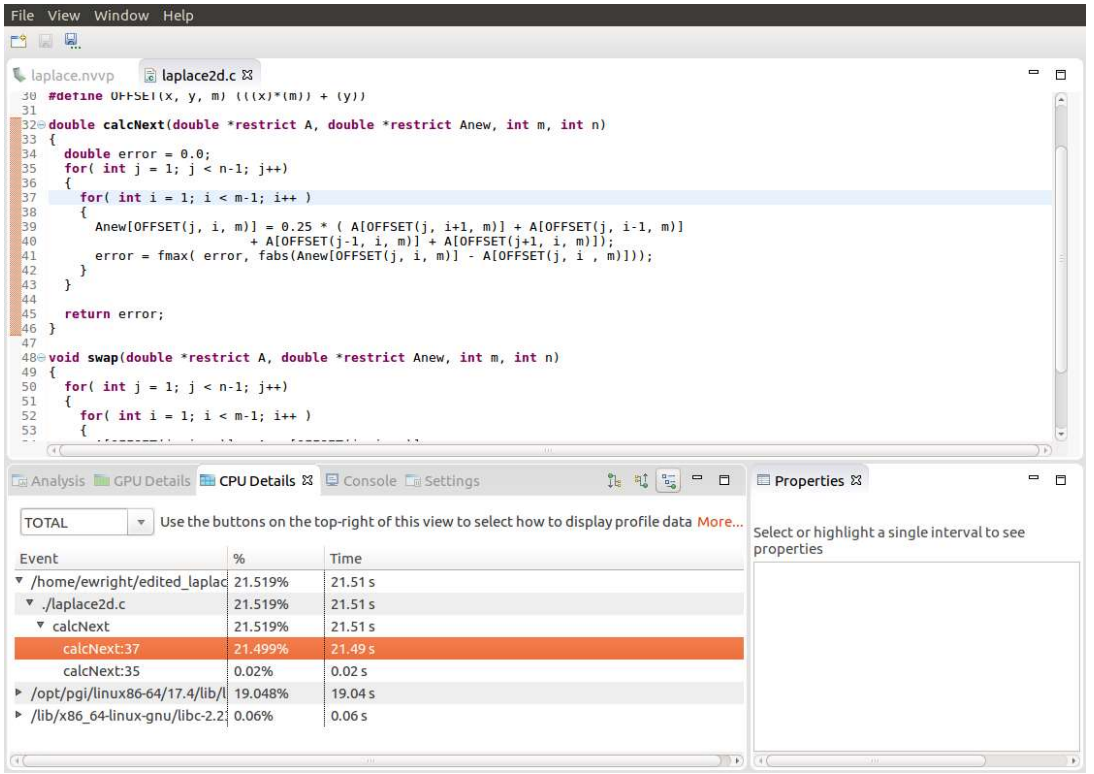
Total Runtime: 39.43 seconds



PROFILING SEQUENTIAL CODE

PGPROF

- We are also able to select the different elements in the CPU Details by double-clicking to open the associated source code
- Here we have selected the “calcNext:37” element, which opened up our code to show the exact line (line 37) that is associated with that element



OpenACC NVIDIA aws Linux Academy

OPENACC PARALLEL LOOP DIRECTIVE

OPENACC PARALLEL DIRECTIVE

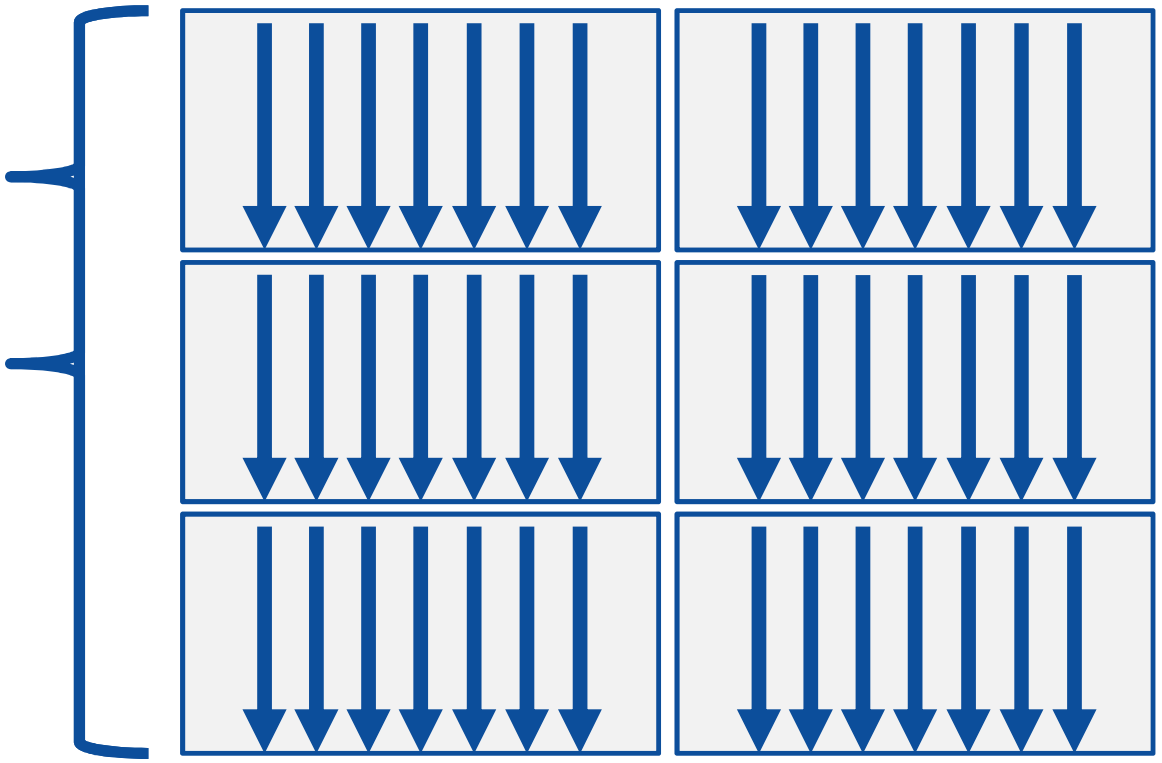
Expressing parallelism

```
#pragma acc parallel
{

    #pragma acc loop
    for(int i = 0; i < N; i++)
    {
        // Do Something
    }

}
```

The ***loop*** directive informs the compiler which loops to parallelize.



OpenACC NVIDIA aws Linux Academy

OPENACC PARALLEL DIRECTIVE

Parallelizing a single loop

C/C++

```
#pragma acc parallel
{
  #pragma acc loop
  for(int i = 0; j < N; i++)
    a[i] = 0;
}
```

Fortran

```
!$acc parallel
  !$acc loop
  do i = 1, N
    a(i) = 0
  end do
!$acc end parallel
```

OpenACC NVIDIA aws Linux Academy

- Use a **parallel** directive to mark a region of code where you want parallel execution to occur
- This parallel region is marked by curly braces in C/C++ or a start and end directive in Fortran
- The **loop** directive is used to instruct the compiler to parallelize the iterations of the next loop to run across the parallel gangs

PARALLELIZE WITH OPENACC PARALLEL LOOP

```
while ( err > tol && iter < iter_max ) {
  err=0.0;

#pragma acc parallel loop reduction(max:err)
  for( int j = 1; j < n-1; j++) {
    for(int i = 1; i < m-1; i++) {

      Anew[j][i] = 0.25 * (A[j][i+1] + A[j][i-1] +
                           A[j-1][i] + A[j+1][i]);

      err = max(err, abs(Anew[j][i] - A[j][i]));
    }
  }

#pragma acc parallel loop
  for( int j = 1; j < n-1; j++) {
    for( int i = 1; i < m-1; i++ ) {
      A[j][i] = Anew[j][i];
    }
  }
  iter++;
}
```

Parallelize first loop nest, max *reduction* required.

Parallelize second loop.

We didn't detail *how* to parallelize the loops, just *which* loops to parallelize.

OpenACC NVIDIA aws Linux Academy

BUILD AND RUN THE CODE

PGI COMPILER BASICS

pgcc, pgc++ and pgfortran

- Use pgcc, pgc++, and pgfortran to compile for C, C++, Fortran
- The -ta flag enables building OpenACC code for a “Target Accelerator” (TA)
- -ta=multicore – Build the code to run across threads on a multicore CPU
- -ta=tesla:managed – Build the code for an NVIDIA (Tesla) GPU and manage the data movement for me (more next week)

```
$ pgcc –fast –Minfo=accel –ta=tesla:managed main.c
$ pgc++ -fast -Minfo=accel –ta=tesla:managed main.cpp
$ pgfortran –fast –Minfo=accel –ta=tesla:managed main.f90
```

CLOSING REMARKS

KEY CONCEPTS

This week we discussed…

- What is OpenACC
- How profile-driven programming helps you write better code
- How to parallelize loops using OpenACC's **parallel loop** directive to improve time to solution

Next:

- Managing your data with OpenACC

OPENACC RESOURCES

Guides • Talks • Tutorials • Videos • Books • Spec • Code Samples • Teaching Materials • Events • Success Stories • Courses • Slack • Stack Overflow

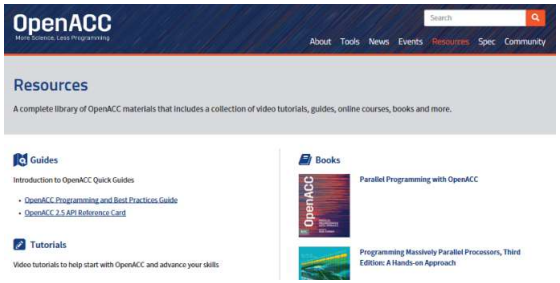
FREE Compilers

PGI Community Edition

OpenACC More Science, Less Programming · NVIDIA · aws · Linux Academy

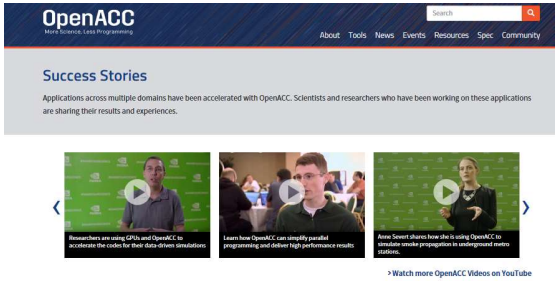
Resources

https://www.openacc.org/resources



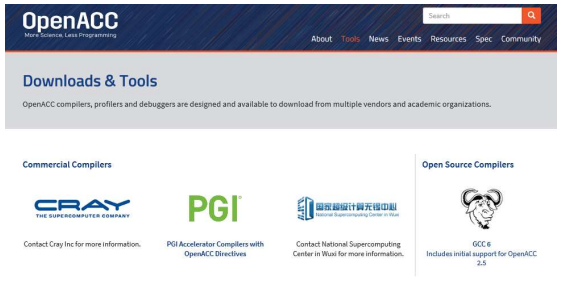
Success Stories

https://www.openacc.org/success-stories



Compilers and Tools

https://www.openacc.org/tools



Events

https://www.openacc.org/events