Cornell Virtual Workshop

Topics

Programming Languages

An Introduction to Linux
This tutorial is for the beginning Linux user, intended to get the user acquainted with some of the basic principles of the Linux operating system.

An Introduction to C Programming
This module is for the beginning programmer who wants to learn the effective use of the C language. If you have never programmed before you can also use this document to learn the basic concepts of programming; however, you may want to have other references to guide you.

An Introduction to Fortran Programming
This module is for the beginning programmer who is interested in learning the effective use of the Fortran language. If you have never programmed before you can also use this document to learn the basic concepts of programming; however, you may want to have other references to guide you.

An Introduction to Python
Python is a programming language designed with ease of programming and readable code as its foremost goals. Python has risen to prominence in scientific computing as the ideal tool for doing data conversions, scripting parameter studies, and in facilitating the scientific workflow. In this online course, a quick overview of the language is presented, along with a few tricks to maximize the utility of Python for engineering and science modeling.

Python for High Performance
While Python is a scripting language, it has plenty of facilities for high performance computing. This article covers some of its features and libraries that are particularly helpful when moving scientific code to a large cluster resource. It also includes specific recipes for compilation and execution on the TACC clusters.

An Introduction to R on XSEDE Resources
This module will serve as an introduction to R, briefly covering the basic syntax of the language and illustrating some of its data handling and statistical capabilities. The focus will be on how to run R in various environments and especially how to run R in parallel.

Balancing Scripts and Compiled Code in Scientific Applications

This module works through examples of scientific application code written in a mix of scripting languages and C, C++ or Fortran code in order to evaluate where, within an application, scripting is a good choice.

MATLAB Programming

MATLAB provides matrix manipulation, plotting, and general purpose scientific programming capability, as well as functionality through specialized "toolboxes" such as the Optimization toolbox, the Statistics toolbox, the Signal Processing toolbox, the Image Processing Toolbox, etc.

Introduction to GPGPU and CUDA Programming

This module briefly covers general GPU topics such as hardware architecture and application speedup, followed by an introductory section of CUDA programming and performance optimization topics.

Advanced SLURM

SLURM (Simple Linux Utility for Resource Management) is a group of utilities used for managing workloads on compute clusters. On Stampede, all jobs executed on the compute nodes are managed by SLURM. This module is for users who are already familiar with the process of submitting jobs via SLURM, but whose needs go beyond submitting simple batch files or interactive jobs.

Parallel Computing

Applications of Parallel Computing

This set of lectures is an online rendition of Applications of Parallel Computers taught by Jim Demmel at U.C. Berkeley in Spring 2012. This online course is sponsored by the Extreme Science and Engineering Discovery Environment (XSEDE), and is only available through the XSEDE User Portal.

Parallel Programming Concepts and High-Performance Computing

Concepts concerning parallel processing and its efficient realization within different hardware and software environments.

Stampede Environment

This module provides the information you need to start working on Stampede and includes sections on logging on to Stampede, the hardware/software environment, moving files to Stampede, compiling, and using the SLURM batch system.

Message Passing Interface (MPI)

MPI is a de facto standard specifying the interface and functionality of a message-passing library, a collection of routines for facilitating communication (exchange of data and synchronization) among the tasks in a distributed memory parallel program. MPI is the first standard and portable message passing library that offers good performance.

MPI Point-to-Point Communication

This module details and differentiates the various types of point-to-point communication available in MPI. Point-to-point communication involves transmission of a message between a pair of processes, as opposed to collective communication, which involves a group of processes.

MPI Collective Communications

The purpose of collective communication is to manipulate a shared piece or set of information. In this module, we introduce these routines in three categories: synchronization, data movement, and global computation.

MPI One-Sided Communication

One-sided communication provides natural access to Remote Memory Access (RMA) functionality that is provided by low-latency interconnect fabrics such as InfiniBand. In this module, we will introduce the various components of MPI RMA and how to use them.

MPI Advanced Topics

This module will introduce you to some of the advanced capabilities of MPI beyond ordinary message passing, including how to customize your environment in the following areas: derived datatypes; groups of processes and their associated communicators; virtual topologies among processes; and parallel I/O using MPI-IO. Application to specific architectures such as Stampede will be discussed.

Parallel I/O

This module presents basic concepts and techniques that will allow your application to take advantage of parallel I/O to increase throughput and improve scalability. Emphasis is placed on the Lustre parallel file system, and on MPI-IO as a fundamental API.

OpenMP

In the shared-memory, heterogeneous environment that Stampede has on each node, it is much easier to introduce parallelism into your code with OpenMP than to do pthread programming from scratch or to use MPI. This module introduces OpenMP and describes how to use it.

Hybrid Programming with OpenMP and MPI

In hybrid programming, the goal is to combine techniques from OpenMP and MPI to create a high-performance parallel code that is better tailored for the non-uniform and heterogeneous memory access characteristics of Stampede. To meet this goal, it is necessary to understand the effects of processor affinity and memory allocation policy, and to exert some control over them.

MIC

The Xeon Phi coprocessor is a system on a PCIe card designed to provide high levels of floating point performance for highly parallel HPC code. Its architecture is known as Many Integrated Core (MIC). This module describes the MIC architecture behind the Xeon Phi, its performance characteristics, how and when to run code on the coprocessors available within Stampede in order to best take advantage of the resources available.

How to Make the Most of MIC

This module focuses on a question of high interest to many HPC users: what changes might I need to make to my program, or even to my algorithm, so my application can make good use of many-core processors such as the Intel Xeon Phi on Stampede? To answer this, we consider just the main characteristics of Intel's Many Integrated Core (MIC) architecture, along with their implications for how a MIC-enabled code should be put together.

Code Improvement

Profiling and Debugging
This module describes how to obtain detailed performance data for jobs on Stampede. It also discusses tools and techniques for online parallel application debugging.

Scalability
When you request an allocation on a large HPC resource like Stampede, you need to demonstrate that your code is scalable. This module will help you understand what algorithmic and design choices are most likely to help your application perform well in parallel. It will also guide you in figuring out which choices actually do make it scale better on Stampede and other machines of its class.

Code Optimization
There are a few simple things one can do in one's code to make the most of typical computing resources, up to and including high performance computing (HPC) resources. This module covers basic aspects of code optimization that can have a big impact, as well as common performance pitfalls to avoid. The module also explains the main features of microprocessor architecture and how they relate to the performance of compiled code.

PerfExpert
PerfExpert is an easy to use profiling tool developed at TACC. This module shows how to use PerfExpert to obtain a concise assessment of a program's utilization of CPU resources by providing statistics on cache hits and misses, mis-predicted branches, FPU instructions, and more. In addition, PerfExpert is able to provide specific recommendations to improve an application's performance based upon its assessment.

Computational Steering
This module provides an introduction to what computational steering is, the potential benefits from using it, and examples on how you can integrate steering into your existing application.

Use Cases
This module captures Felix Bachmann's presentation at XSEDE12 on Use Cases and Quality Attribute Scenarios.

Vectorization
Vectorization is a process by which mathematical operations found in tight loops in scientific code are executed in parallel on special vector hardware found in CPUs and coprocessors. This module describes the vectorization process as it relates to computing hardware, compilers, and coding practices.

Checkpointing
In this module we'll explore several categories of C/R solutions and go into examples and exercises of some implementations that are particularly well-suited for high-performance computing.

HDF5 for Checkpoint/Restart
HDF5 (the Hierarchical Data Format, version 5) is a library, data model, and file format designed for managing data, especially in HPC where efficient I/O is a high priority. While there are many ways in which HDF5 can be used for modeling data and creating file formats, in this module we focus on how to use HDF5 within a program to save the relevant parts of state needed in order to resume the application with minimal loss of CPU time.

Data Analysis

Globus File Transfer
Globus is a research data management system that provides fast, reliable and secure file transfers and sharing. This module describes the procedures for setting up Globus accounts, creating a Globus endpoint on a personal computer and transferring files using both the Globus web interface and the command line interface, which can be used in experiment scripts.

Large Data Visualization
This module gives an introduction to some concepts of visualization with a focus on the parallel computing techniques used to handle large datasets.

ParaView
ParaView is a visualization application highly capable for computational fluid dynamics and other subjects. It is open source and can run in parallel on Stampede. This module includes a lab which covers visualization of a sample dataset both on a local computer and on TACC resources.

VisIt
VisIt is a visualization application highly capable for computational fluid dynamics and other subjects. It is open source and can run in parallel on Stampede. This module includes a lab which covers visualization of a sample dataset both on a local computer and on TACC resources. VisIt is a free visualization application that runs on Windows and Linux, as well as TACC's Stampede. This module will walk through logging into Stampede and running a sample visualization.

Data Transfer
There are a number of utilities available to accomplish the essential task of transferring data and/or code between the file system on your workstation and a larger computing resource. The one you choose depends on the size and number of files to be transferred as well as the ease of invoking the utility, and the ability to use a script. This module presents the various options and the pros and cons of each of them as well as ways to make these transfers faster.

Relational Databases
This module provides an introduction to relational databases, the most common type of database and what you are most likely to find available at an XSEDE site.

MapReduce
This module describes the basic MapReduce paradigm, the Hadoop MapReduce framework, and techniques for running MapReduce frameworks on HPC resources.

Back