Getting more Python Performance with Intel® optimized Distribution for Python

Agenda for 1st day
May 28, 2018

Description:

This course will introduce the background of the Intel distribution and why it is faster than a plain vanilla Python distribution running on Intel Architectures and guide students with examples on how to apply Python applications and libraries most efficiently resulting in fast applications.

Objective:

Initially, the student will learn the most basic steps to install and run an Intel optimized Python distribution. Then the students will be guided to apply techniques on how to get the best performance out of Python. The student use Intel libraries (e.g. pyDAAL) and examples from classical machine learning field in order to gain deeper insights in performance optimization.

Prerequisites:

- Basic knowledge of programming and ideally also Python language.

Agenda:

10:00	**11:30**	Intel® Distribution for Python • Introduction and Overview • Docker images • Introduction to Intel® VTune™ for Python
11:30	**11:45**	**Coffee Break**
11:45	**13:00**	Package Management with Conda - with Hands-on Labs Python Performance Techniques - Part 1 - with Hands-on Labs • Numpy, numexpr • Profiling
13:00	**14:00**	**Lunch Break**
14:00	**15:15**	Python Performance Techniques - Part 1 - with Hands-on Labs • Numba, cython • MPI • Parallelism (task and others)
15:15	**15:30**	**Coffee Break**
15:30	**17:30**	Classical Machine Learning with Intel® Data Analytics Acceleration Library (Intel® DAAL) • Overview Intel® Math Kernel Library • Overview Intel® DAAL • Hands on pyDAAL • Tech Preview: DAAL4py
17:30	**18:00**	**Wrap-up**

Optimizing your Application for Speed – Performance Analyzer Intel VTune Amplifier XE

Agenda for 2nd day
May 29, 2018

Description:

This full day workshop will focus on performance analysis, as well as tuning shared memory systems. As an introduction, the Intel architecture will be addressed. Then common programming issues and performance bottlenecks will be discussed and addressed during the hands-on trainings.

Objective:

The students will learn about user-mode sampling types for algorithmic and concurrency analysis as well as microarchitectural analysis types. Additional we are addressing general programming methodologies and common parallel bottlenecks and how to solve them.

Prerequisites:

- Basic understanding of parallel programming paradigms and C/C++ or Fortran programming.
- General understanding of microprocessor architectures.

Agenda:

10:00	**11:30**	Introduction to Intel® VTune™ Amplifier Performance Data Collection Modes and Technologies in Intel VTune Amplifier • Demo Lab 1: Basic Hotspot Analysis - Finding Performance Hotspots • Demo Lab 2: Concurrency Analysis - Analyzing Parallelism • Demo Lab 3: Locks and Waits Analysis - Identifying Parallelism issues • Instrumenting code – Collection control using APIs
11:30	**11:45**	**Coffee Break**
11:45	**13:00**	Intel Core and Intel Xeon and Intel Xeon Phi microarchitectures overview Front-End, Back-End, Execution Units, Scheduler • Introduction to Performance Monitoring Unit • Event Based Sampling (EBS) Technology
13:00	**14:00**	**Lunch Break**
14:00	**15:15**	Performance Analysis Methodology • Identifying hotspots, determining efficiency of hotspots, and fixing issue Advanced hands-on activities • Advanced Lab 1 – Find the performance hotspot using Advanced Hotspot Analysis • Advanced Lab 2 – Determining code efficiency; find stalls using General Exploration
15:15	**15:30**	**Coffee Break**
15:30	**16:45**	Advanced hands-on activities continued. • Advanced Lab 3 – Finding memory issues • Advanced Lab 4 – Finding computational issues Lab (new): Tuning Python Code with Intel® VTune™ Amplifier
16:45	**17:00**	**Q & A session**

Overcome Vectorization Bottlenecks in your Application – Using Intel Advisor XE and the new Roofline component

3rd day
May 30

Description:

All modern processors are enhanced by a SIMD (single instruction, multiple data) or vector facility to accelerate computation considerably by executing instructions on multiple operands in parallel. On x86 architecture these extensions are known by SSE, AVX, AVX-2 etc. Without making best use of these vector extensions, it is impossible to get close to the optimal performance the computer system provides. We will introduce the vector instructions relevant for x86-architecture based computer systems today and will briefly talk about how to exploit them. However, in general developers will rely on the compiler vectorization capabilities to get the best out of it. But this automatic vectorization frequently needs support from the developer to deal with dependencies, alignment, data type conversions, and many other topics we will cover here. This is supported by compiler vectorization reporting which helps to find why vectorization failed and what can be done to get it working. On top we will use the Intel Advisor XE component and the Intel Roofline to show how to improve the performance of inefficient vectorized applications.

Objective:

The student will get to know the following Intel components: C/C++ Compiler, Roofline, and Intel Advisor XE in the context of vectorizing applications efficiently.

Prerequisites:

- Intermediate experience with one of the programming languages C/C++ or Fortran.

Agenda:

10:00	**11:00**	Intel Advisor - General presentation • Introduction about SIMD and vector registers. SSE, AVX, AVX512 • Characterization of applications with roofline model • Vectorization approaches with Advisor: (Auto-vectorization, OpenMP, pragma) - Peel, vectorized body, remainder, alignment ; Cache simulation with Advisor
11:00	**11:15**	**Coffee Break**
11:15	**12:00**	Running the roofline model and survey in Advisor - with Hands-on Labs • Is my application performing as expected ? • Is my application vectorized ?
12:00	**13:00**	**Lunch Break**
13:00	**14:00**	Memory Access Pattern (MAP) Analysis in Advisor - with Hands-on Labs • How can I detect inefficient memory access pattern ? • How can I solve the problem ? • Using Intel SDLT to improve vectorization
14:15	**15:00**	Dependency Analysis in Advisor - with Hands-on Labs • Is vectorization safe ? (Compiler reports) • Pragma vector, ivdep , OpenMP 4.* and pragma simd
15:00	**15:15**	**Coffee Break**
15h15	**16:45**	Iso3DFD – Vectorization and optimizations of a finite difference kernel and discussions