Scaling Out: Effective Cluster Computing with Distributed Dask

Course Summary

Description

This class addresses the transition from working successfully on a single server or experimenting with a minimal cluster to achieving successful, reliable, repeatable use of larger Dask compute clusters. We focus on a deep dive into all of the critical components in a distributed Dask cluster, how they work together, and how you can configure them to maximize throughput and minimize costs.

Objectives

At the end of this course, students will understand:

- What components make up a distributed Dask cluster and what purposes they serve
- How to configure cluster resources to meet your workload needs
- How to identify problems, debug, and troubleshoot successfully

Topics

- Introduction
- Distributed Dask: Cast of Characters
- Basic Operation of Dask Clusters
- Tasks
- Distributed Data
- Resource usage and Resilience
- Best Practices, Debugging
- Use Case Example: Orchestrating Batch ML Scoring
- Q & A Discussion

Audience

This course is intended for engineers or data scientists who typically work with large data clusters.

Prerequisites

Students should have experience in Python and Pandas and/or SQL programming, both at a basic level.

Duration

One day

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Scaling Out: Effective Cluster Computing with Distributed Dask

Course Outline

I. Introduction

A. About Dask and Coiled Computing: Making scale-out computing easier
B. Simplest distributed cluster: manual setup
C. Changes in transitioning to distributed environment
Storage, fast universal memory access, single shared executable
D. Implications for users (devs) and admin (IT)

II. Distributed Dask: Cast of Characters

A. Client, Scheduler, Nanny, Worker
B. Where these services are located, their relationships and roles
C. Supporting Players: cluster resource manager (e.g., k8s, Coiled Cloud, YARN, etc.)

III. Basic Operation of Dask Clusters

A. User perspective
B. Creating clusters with helper tools: Cloud Provider, Coiled Cloud, etc.
C. Cluster API
D. Sizing your cluster
E. Scaling your scaling – manual/automatic
F. Admin perspective
  a. CLI: dask-scheduler and dask-worker
  b. Managing the worker environment
  c. Additional admin concerns (security, tagging, and costs)

IV. Tasks

A. Submitting tasks and directing output
B. Scheduling policy
C. Finding your tasks and data (programmatically)
D. Seeing your tasks and data: the Dask Dashboard

V. Distributed Data

A. Source data via tasks
B. Source data scatter
C. Storing data worker-local
D. Handling output (result) data, direct parallel write vs. gather/result

VI. Resource usage and Resilience

A. Output spill location and resource management
B. Work stealing
C. Loss of processes
D. Loss of storage on workers

VII. Best Practices, Debugging

A. Dashboard information pages
B. Additional GUIs (e.g., profiler)
C. Review of best practices
D. Remote debugging
E. client.run

VIII. Use Case Example: Orchestrating Batch ML Scoring

A. Source data on disk
B. ML model
C. Options for inference, pros/cons
D. Supplying dependencies via code or container image
E. Basic workflow
F. Improvements and optimizations (e.g., batch size)

IX. Q & A, Discussion