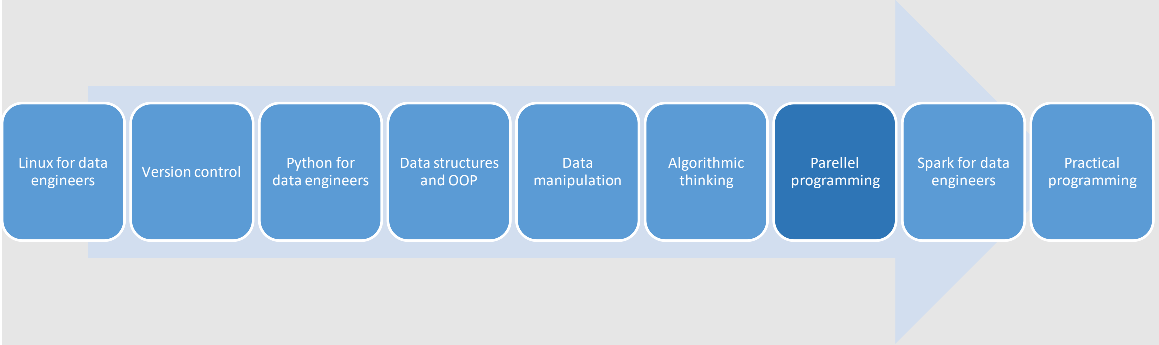
Topic 7 – Parallel programming

This document is the handbook for Topic 8 – **Parallel Programming** – within Module 3 – **Programming and Scripting Essentials**.

The purpose of this document is to guide your learning throughout this topic and help you to maximise the value you get from the materials provided by the BPP School of Technology.

Context

This handbook is for one of 9 topics for this Module.



Every topic contributes towards the ultimate learning objectives for the Module, which you will be assessed on at the end of the term.

Module Learning Outcomes

On successfully completing this module, you will be able to:

- **Employ** software development tools and techniques for designing, deploying and maintaining secure data products and pipelines, including debugging, version control and testing.
- **Construct** algorithms that correctly and efficiently handle data at scale whilst mitigating risks.
- **Demonstrate** the knowledge of the steps needed to prepare the code for production.

Module Assessment

The Level 5 Data Engineer EPA has two assessment methods, each with its own mapping of KSBs. The Assessment plan and assessment guidance documents above list the criteria and KSBs that are assessed. The criteria group the KSBs and describe what the apprentice needs to do to achieve a pass or distinction for that assessment method.

Both assessment methods need to be passed by the candidate:

(1) Project with report

The learner will complete a project and write a report of 3500 words. Project brief submitted at gateway:

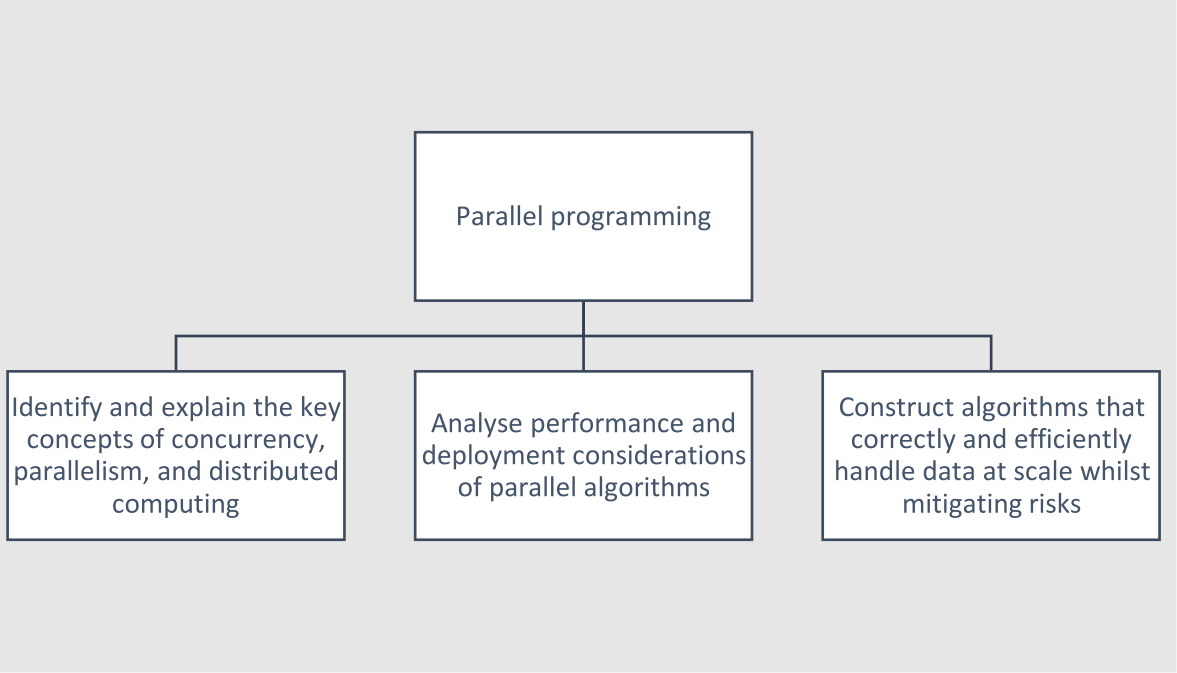
- Learners will have 10 weeks to complete the project and submit the report to the EPAO
- Learners also need to prepare and give a presentation to an independent assessor on their project
- The presentation with questions will last at least 50 minutes. The independent assessor will ask at least 6 questions about the project and presentation
- The project has to have real business application and benefit. Candidates are expected to showcase the use of appropriate standards for sustainability, privacy and security, thoroughly document their data pipeline designs, explain the choice of relevant tooling and demonstrate operational awareness of deployment, access control, risks, and how other stakeholders may be impacted positively and negatively

(2) Professional discussion underpinned by a portfolio of evidence

- Learners will have a professional discussion with an independent assessor. It will last 80 minutes
- They will be asked at least 10 questions about Data Engineering
- The portfolio of evidence will be used to help answer the questions
- We expect the candidates to demonstrate examples of working with data teams on data projects and data products, showcase ideas for future-proofing data, be clear on applying problem-solving skills, show regulatory awareness, and sensitivity towards data quality, data governance and areas for continuous improvement, both personal and organisational

Topic Learning Outcomes

As a step towards build your skills towards the final module assessment, the learning objectives for this topic are:



Parallel programming

Identify and explain the key concepts of concurrency, parallelism, and distributed computing

Analyse performance and deployment considerations of parallel algorithms

Construct algorithms that correctly and efficiently handle data at scale whilst mitigating risks

Introduction

Parallel programming is a crucial skill in the era of big data and high-performance computing. This topic will equip you with the ability to identify and explain key concepts of concurrency, parallelism, and distributed computing, and their application within Python programming, a language known for its simplicity and power. We'll delve deep into these foundational concepts, exploring how they differ and complement each other. You'll learn about threads, processes, synchronization mechanisms, and distributed systems architecture. Understanding these concepts is essential for leveraging the full power of modern multi-core processors and distributed computing environments.

We will delve into the performance and deployment considerations of parallel algorithms, empowering you to make informed decisions when implementing these in real-world scenarios. This focus on practical aspects of implementing parallel solutions will include examining various metrics for measuring performance gains, such as speedup and efficiency. You'll learn to identify bottlenecks, optimize communication overhead, and balance workloads across multiple processors or nodes. We'll also discuss deployment strategies for different parallel computing platforms, including clusters, grids, and cloud environments.

Additionally, you will explore the advantages of Spark for parallelism and assess other similar platforms, expanding your toolkit for managing large-scale data processing tasks. We'll dive into Spark's distributed computing model, its integration with Python through PySpark, and compare it with other big data processing frameworks. This knowledge will be crucial as we address the challenges of designing parallel algorithms that not only work correctly but also scale efficiently with increasing data sizes and computing resources.

Throughout this topic, we'll explore parallel algorithm design patterns, data partitioning strategies, and techniques for ensuring correctness in the face of concurrency. We'll also discuss common pitfalls and risks in parallel programming, such as race conditions, deadlocks, and data inconsistencies, and strategies to mitigate them. By mastering these concepts and skills, you'll be well-equipped to construct algorithms that correctly and efficiently handle data at scale while mitigating risks.

Acquiring this knowledge and these skills will not only make you a more versatile and effective data engineer but also pave the way for new opportunities in your future career. You'll be prepared to tackle complex computational problems efficiently, handle large-scale data processing tasks, and adapt to the ever-evolving landscape of parallel and distributed computing. This expertise will enhance your capabilities in fields such as scientific computing, machine learning, and cloud-based applications, positioning you at the forefront of modern computing practices.

Structure

Topics for this programme follow a Prepare-Collaborate-Apply structure:

Prepare

This is the stage where you build the knowledge to underpin your learning. This might involve completing interactive e-learning packages, watching videos, or working through reading materials.

It is essential that you make the most of the learning materials provided before attending webinars, as this will allow you to test your knowledge and stretch you understanding further.

Collaborate

This is where you will receive guidance from our expert tutors and coaches to shape and refine your understanding through in-depth explanation, discussion, testing and carrying out more advanced practical and realistic tasks. This also helps to develop valuable team-working skills.

Apply

You now apply the knowledge you have developed to real-world tasks.

Off-the-job learning tasks

This stage is all about ensuring you truly grasp and retain what you've learned. Through completion of off-the-job (OTJ) revision tasks and tests, you'll get plenty of practice applying your knowledge. Plan to dedicate 6-8 hours each week to guided study and portfolio work, with sessions typically on the same day each week.

Task 1 brief:

This task involves setting up and executing a series of PySpark operations in Microsoft Azure Databricks. The steps include:

Setup: Log into Microsoft Azure using the Github Student Pack, select Azure Databricks, create a new deployment using the free trial tier, create a new cluster, and create a new notebook.

Cell 1 - Initialise PySpark: Set up the PySpark environment and initialize it.

Cell 2 - Load and Collect Data: Upload a file named “Example.csv” to Databricks, load the text file using the SparkContext, split the lines into a list, and collect the dataset into local RAM. You’re also asked to research and write about the pros and cons of running collect() in Spark.

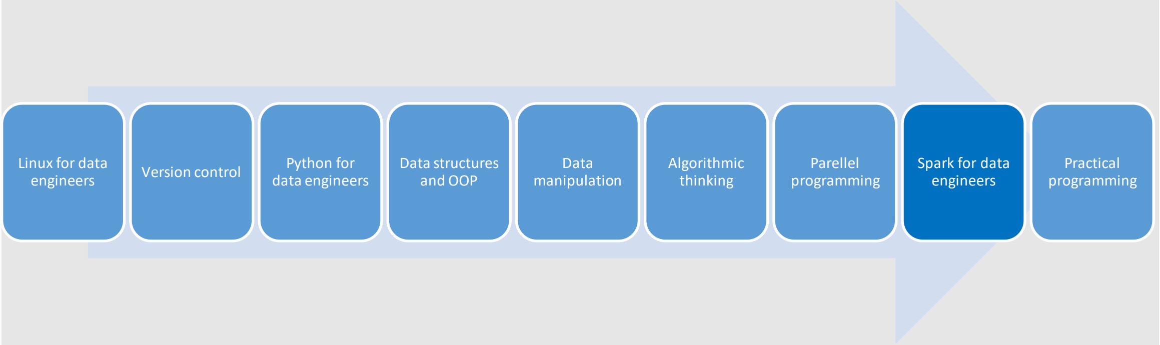
Cell 3 - GroupBy Operator: Group the records by the name of the person, show the first group, count the groups, and collect the job counts. You’re asked to explain the output in plain English.

Cell 4 - Map vs FlatMap: Compute a relation of words by line using map and flatMap functions, and explain the difference between these two functions with examples and a plain English explanation.

Throughout the task, you’re expected to fill in the blanks in the provided code snippets and provide explanations for the outputs and operations.

Link

This handbook is for one of 9 topics for this Module.



The sequence of topics in this module is carefully designed so that your knowledge and skills will develop as you progress.

The next topic is **Spark for data engineers**.