COMP0235 Coursework challenge

Introduction

You have been employed by UCL as a Research Software Engineer in your role you help researchers from many fields scale up various computational analyses they have created. A group of biochemistry researchers at UCL's Medical School would like to make use of all the data at the AlphaFold Database hosted by the European Bioinformatics Institute.

They have written a small data analysis script that takes as input a list of 3D protein models and outputs a series of results files. The analysis is comprised of 2 executation steps; the first takes in a protein model and uses a machine learning tool called Merizo Search to analyse the model, the second step parses the output files and outputs a summary of the data that the researchers wish to capture.

A prediction for each protein model takes their script 40 seconds and there are 215 million proteins in the whole database. They estimate it would take the computer they have access to nearly 10,000 days to analyse the whole database!

They have approached you to help turn their data analysis program in to a distributed analysis system which can run their predictions in a more timely fashion. They provide you with their python scripts, some example input and output data and the links to the datasets they would like you to analyse. Their pipeline executes Merizo Search and a small results parsing python script they wrote. To test the system the researchers provide links to two initial datasets you should analyse for them, one for Human data and one for Escherichia Coli (*E.coli*) data.

The researcher's pipeline and data

The researchers provide you with web links to two files. One file containing all the 3D models for all the proteins in the human proteome, **UP000005640_9606_HUMAN_v4.tar**, and one containing 3D models for all proteins in the *E.coli* proteome, **UP000000625_83333_ECOLI_v4.tar**.

Their code comes as two python scripts. The first, **pipeline.script.py,** runs the data analysis pipeline they have written. The second, **results_parser.py** is a short piece of code that their pipeline script requires which parses and summarises the outputs from the first step.

If you open **pipeline.script.py** you can follow the logic of their analysis. The script makes use of multiprocessing.pool and can be trivially parallelised on a single computer. It follows these steps

1. The script first reads in the locations of all the **.pdb** files in a provided input directory
2. Next the script forks one or more threads to run two data analysis steps
3. Merizo search is run over a single pdb file to produce two output files
4. **results_parser.py** is run against the resulting **_search.tsv** file to produces the final **.parsed** file

The researchers would like you to return to them all of the outputs of the Merzio Search and **results_parser.py** tool plus threes summary files that you must calculate for them (see below)

Coursework task

You you are required to build a distributed data analysis across 4 or 5 cloud machines. This restricted resourcing is there to make you think about how to complete such tasks where you do not have access to limitless funds and hardware. You are free to accomplish the coursework as you see fit. However, your machines must have the following specification:

Machine ID	**Number**	**Cores**	**RAM**	**HDD1**	**HDD2**
Host	1	2	4GB	10GB	NA
Worker	3	4	32GB	25GB	NA
Storage (optional)	1	4	8GB	10GB	200GB
Total	5	18	108GB	95GB	200GB

Your solution **MUST** include **one** host machine and **three** worker machines. You may optionally include a storage machine if you believe your solution to the problem requires it.

Your data analysis solution should include the following features:

1. Use an appropriate configuration system. We have covered Ansible and Salt but others are available.
2. Use an appropriate method of data storage to store the 3D models (contained in the files **UP000005640_9606_HUMAN_v4.tar** and **UP000000625_83333_ECOLI_v4.tar**).
3. Use an appropriate method to store the results files and an appropriate method to make these accessible to the researchers you are collaborating with.
4. Use some appropriate monitoring and logging of your mini-cluster and your data analysis pipeline
5. You must collate the **.parsed** files in to a pair of summary files, one for Human inputs and one for *E.Coli* inputs, these must be called **human_cath_summary.csv** and **ecoli_cath_summary.csv**, these must also be available to the researchers to access.

6. You should collate a further csv which includes the mean plDDT scores across all the human results and the mean plDDT score across all the *E.Coli* results called **plDDT_means.csv**

Examples of all input and output files are provided

Challenges/hints

1. All parts of the process should be automated, from commisionsing your VMs, configuring and installing the pipeline software and running the pipeline. For instance, you should not require a user/installer of your system to use ssh to access the worker/storage nodes
2. On your 4 or 5 machines we estimate it should take about one to two days to run all the calculations.
3. The host instances are too small to run the calculations
4. Worker instances have only 4 CPUs be aware of the Load Average of the machine at runtime
5. You need to be able to understand how to install and run the merizo search and **results_parser.py** programs
6. You need to be able to understand how to fetch the required datasets from AlpahFoldDB and install the appropriate search database for merizo search
7. You should ensure you can successfully run **pipeline.script.py**. This could be on either your own machine or on one of the cloud machines you have access to. In the directory `example_files/`. You have an example input file, **test.pdb**. If you run the script successfully you should produce a number of intermediary files, examples of these can be found in the directory; **test_segement.tsv, test_search.tsv and test.parsed**. Examples of final output summary files you must produce are also included, **human_cath_summary.csv, ecoli_cath_summary.csv, plDDT_means.csv**
8. Your system must have some means through which the researchers can collect the results and summary files

Deliverables

1. A short report (no more than 4,000 words) that explains *why* you have designed your data analysis system/pipeline the way you have. You should explain the logic of the system and importantly why you have made the decisions you have. Where there are choices in technology you could use or alternative design approaches, you should explain and justify your choices. For example, if you have chosen to use Ansible for machine configuration you should, explain why you have chosen this instead of the many other possibilities (ie. chef, salt, puppet, etc…), and why it is most appropriate for your use case. As another example, you should explain and justify your choice for data and results storage. You should explain and justify all choices salient to how you have chosen to implement your pipeline **in reference to alternative decisions you could have made**. You should discuss how scalable your system is for different scales of inputs. What is the maximum feasible size of

analysis? What can be done to improve your system? What might you change in the future? The report must contain a link to a github repository that contains all the code you used to accomplish the coursework task. You must include the IP addresses and identities of the machines running your analysis system.

2. The second deliverable is a set of (4 or 5) cloud machines running on Condenser which are fully configured with your data analysis system installed. It should be possible for the person marking your coursework to log in to these machines (using the provided **lecturer_key.pub**) and run your data analysis pipeline as per the instructions you've provided.
3. The last deliverable is a github repository that contains all the code required to create and install your distributed data analysis system and the code required to run the analysis to produce all the results files. Your code repository must include instructions on how to use your code to install AND then run your data analysis system. The person marking your coursework should be able to check out your github to a fresh set of machines and follow the instructions such that it will install everything and run the pipeline on the 2 example (human and ecoli) datasets.

Pipeline Dependencies

Executables

1. Merizo Search - https://github.com/psipred/merizo_search
2. pipeline_script.py
3. results_parser.py

Datasets

1. Human AlphaFoldDB models (23,391 models): **https://ftp.ebi.ac.uk/pub/databases/alphafold/latest/UP000005640_9606_HUMAN_v4.tar**
2. *E.coli* AlphaFoldDB models (4,363 models): **https://ftp.ebi.ac.uk/pub/databases/alphafold/latest/UP000000625_83333_ECOLI_v4.tar**
3. CATH Foldclass DB for merizo search: **http://bioinfadmin.cs.ucl.ac.uk/downloads/merizo_search/cath_foldclassdb.tar.gz**

Critical Python dependencies

1. biopython
2. torch
3. numpy
4. scipy
5. faiss-cpu

Background Reading

1. https://en.wikipedia.org/wiki/Human_genome
2. https://en.wikipedia.org/wiki/Escherichia_coli
3. https://en.wikipedia.org/wiki/Protein_structure
4. https://en.wikipedia.org/wiki/Proteome
5. https://en.wikipedia.org/wiki/AlphaFold
6. https://alphafold.ebi.ac.uk/
7. https://www.biorxiv.org/content/10.1101/2024.03.25.586696v1
8. https://en.wikipedia.org/wiki/CATH_database and https://www.cathdb.info/
9. https://en.wikipedia.org/wiki/Protein_Data_Bank_(file_format)