



# **A HYBRID HIGH-PERFORMANCE COMPUTING AND BIG DATA FRAMEWORK FOR PARALLELIZED GENE-PROTEIN EXPRESSION ANALYSIS AND DISEASE CLASSIFICATION**

**CBIO312: HIGH PERFORMANCE COMPUTING**

**Name and ID:**

**Farah Ibrahim 221001140**

**Malak Atef 221000906**

**Zeina Ahmed 221000417**

**Under Supervision of: Dr. Mohamed EL-Sayeh**

# INTRODUCTION

- Project Goal: Build and compare traditional HPC and Big Data frameworks for analyzing gene/protein expression data.

## Tools Used:

- VirtualBox + Ubuntu (for VM cluster)
- MPI + mpi4py (Task 1)
- Docker + Spark + PySpark (Task 2)
- scikit-learn, pandas, logistic regression



# METHODOLOGY

## Task 1: Mini-HPC Cluster using MPI

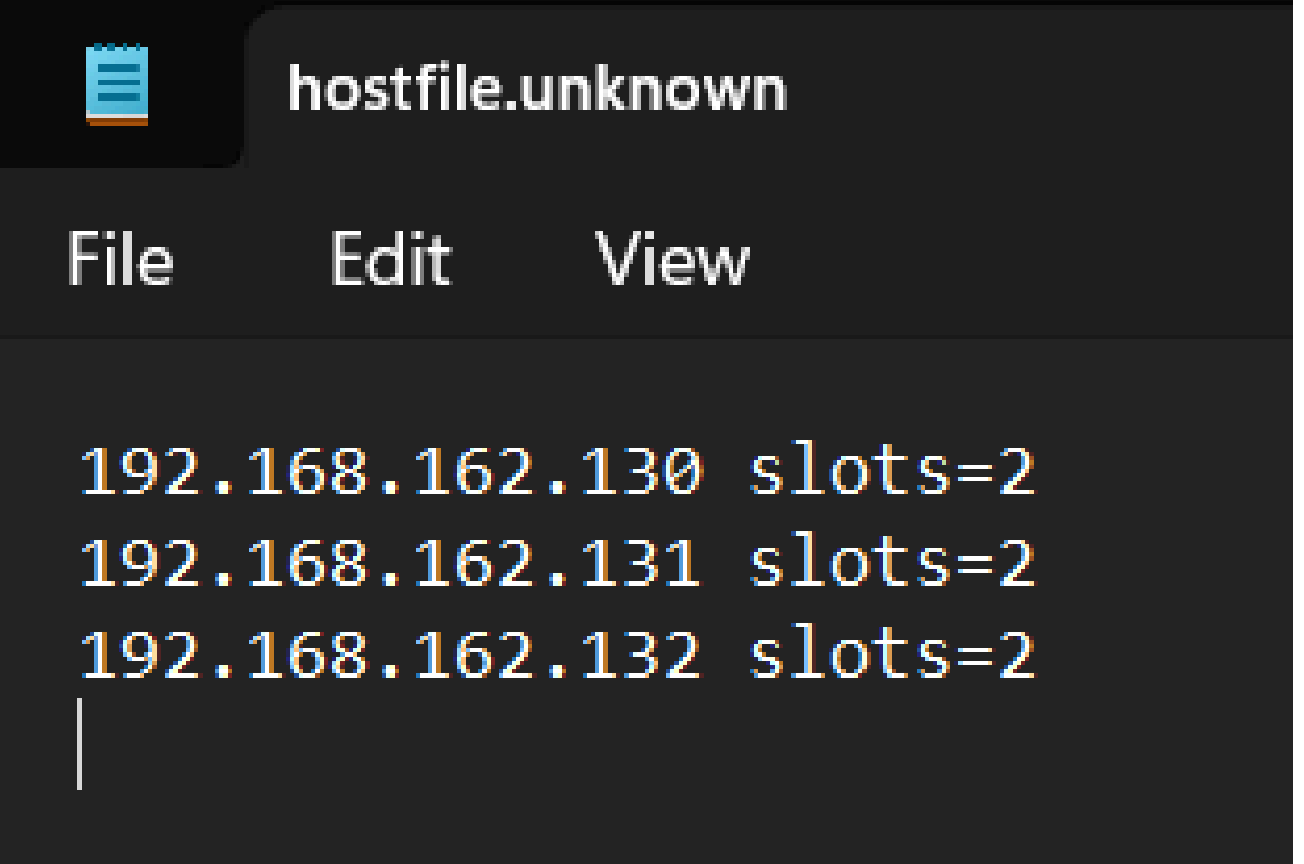
- VM Setup:
  - 3 Ubuntu VMs (1 master, 2 workers)
  - NAT networking and hostnames mapped
- Passwordless SSH:
  - ssh-keygen and ssh-copy-id across nodes
  - Verified with test commands



# METHODOLOGY & RESULTS

## Task 1: Mini-HPC Cluster using MPI

- MPI Configuration:
  - Created hostfile
  - Installed openmpi, python3, mpi4py, scikit-learn
- Script: process\_dataset\_mpi.py
  - Part 1: Gene expression comparison (healthy vs diseased)
  - Part 2: Logistic regression on gene/protein features
  - Used mpirun -np 6 to run across nodes



```
hostfile.unknown
File Edit View
192.168.162.130 slots=2
192.168.162.131 slots=2
192.168.162.132 slots=2
```



# METHODOLOGY & RESULTS

## Task 2: Dockerized Spark Cluster

### Docker Installation on All Nodes

- Master: docker swarm init
- Workers joined using docker swarm join

### Swarm Initialization

### Spark Stack Deployment

- Wrote spark-swarm.yml
- Deployed with docker stack deploy



# METHODOLOGY & RESULTS

## Task 2: Dockerized Spark Cluster

May 31 20:53

Spark Master at spark://c x

Not Secure http://192.168.162.130:8081

Import bookmarks...

Spark 3.5.6

Spark Master at spark://c543a6527f12:7077

URL: spark://c543a6527f12:7077

Alive Workers: 2

Cores in use: 4 Total, 0 Used

Memory in use: 5.6 GiB Total, 0.0 B Used

Resources in use:

Applications: 0 Running, 0 Completed

Drivers: 0 Running, 0 Completed

Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20250531174717-10.0.1.7-41135	10.0.1.7:41135	ALIVE	2 (0 Used)	2.8 GiB (0.0 B Used)	
worker-20250531174718-10.0.1.6-43973	10.0.1.6:43973	ALIVE	2 (0 Used)	2.8 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------



# METHODOLOGY & RESULTS

## Task 2: Dockerized Spark Cluster

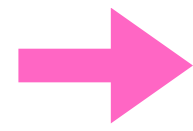
### ML Pipeline in PySpark

- Script: bio\_classifier.py
- Read CSV → Feature engineering → Logistic regression
- Accuracy saved to result.txt **“Model Accuracy: 100.00%”**



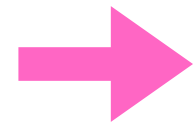
# ERRORS & CHALLENGES FACED

## Networking Issues



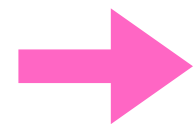
Static IPs not persisting between reboots → fixed via netplan config

## SSH Errors



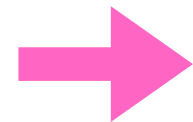
Wrong permissions on .ssh folder → fixed with chmod 700 and 600

## MPI Crashes



Hostfile misconfigured → corrected IPs and slot counts

## Docker Permissions



Required newgrp docker to apply usermod





# CONCLUSION

- We successfully deployed a hybrid computing architecture combining HPC and Spark.
- MPI was lightweight and faster to set up for smaller tasks.
- Spark offered better scalability and fault-tolerance.
- Both methods were effective at handling bioinformatics ML tasks.
- We gained real-world experience in systems setup, distributed ML, and debugging parallel computing environments.

**THANK YOU**

