## Câu 1:

## a) Nêu 3 đặc trưng chính của Big Data (3V) và giải thích. (0.5 điểm)

Ba đặc trưng chính của Big Data là 3V:

Volume (Khối lượng): Dữ liệu có khối lượng rất lớn, thường tính bằng terabyte (TB), petabyte (PB) hoặc hơn.Ví dụ: Dữ liệu người dùng trên Facebook, video YouTube, hay log hệ thống.

Velocity (Tốc độ): Dữ liệu được sinh ra và cập nhật liên tục với tốc độ rất cao. Ví dụ: Dòng tweet, giao dịch ngân hàng, cảm biến IoT.

Variety (Đa dạng): Dữ liệu có nhiều định dạng khác nhau: có cấu trúc (SQL), bán cấu trúc (JSON, XML) và phi cấu trúc (hình ảnh, âm thanh, video).

## b) Ngoài 3V, các đặc trưng mở rộng khác của Big Data là gì (5V)? (0.5 điểm)

Các đặc trưng mở rộng thành 5V gồm:

Veracity (Độ tin cậy): Dữ liệu có thể không chính xác, không nhất quán hoặc thiếu độ tin cậy, cần được làm sạch và xác thực.

Value (Giá trị): Mục tiêu cuối cùng của Big Data là khai thác giá trị hữu ích từ dữ liệu để hỗ trợ ra quyết định, dự báo, hoặc tối ưu hoạt động.

## c) Giải thích khái niệm ETL (Extract – Transform – Load) trong Big Data. (1.0 điểm)

ETL là quy trình xử lý dữ liệu trong hệ thống Big Data hoặc Data Warehouse, gồm 3 bước chính:

Extract (Trích xuất): Lấy dữ liệu từ nhiều nguồn khác nhau như CSDL, file log, API, cảm biến,…

Transform (Chuyển đổi): Làm sạch, chuẩn hóa, thay đổi định dạng, tổng hợp hoặc tính toán dữ liệu để phù hợp với mô hình lưu trữ hoặc phân tích.

Load (Nạp dữ liệu): Nạp dữ liệu đã xử lý vào hệ thống lưu trữ đích, như Data Warehouse, Data Lake, hoặc hệ thống phân tích như Hadoop/Spark.

## d) Hệ sinh thái Hadoop gồm những thành phần chính nào? (1.0 điểm)

Hệ sinh thái Hadoop bao gồm nhiều thành phần, trong đó các thành phần chính là:

HDFS (Hadoop Distributed File System): Hệ thống lưu trữ dữ liệu phân tán.

YARN (Yet Another Resource Negotiator): Quản lý tài nguyên và lập lịch xử lý.

MapReduce: Mô hình xử lý dữ liệu song song, chia công việc thành Map và Reduce.

## e) Vì sao Spark được xem là phát triển thế hệ mới của Hadoop? (1.0 điểm)

Apache Spark được xem là thế hệ mới của Hadoop vì:

Xử lý nhanh hơn nhiều lần: Spark lưu trữ dữ liệu trong bộ nhớ (in-memory) thay vì đọc/ghi liên tục xuống đĩa như MapReduce → tốc độ nhanh gấp 10–100 lần.

Hỗ trợ đa dạng mô hình xử lý: Không chỉ xử lý batch mà còn streaming, machine learning, graph processing, SQL, tất cả trong một framework duy nhất.

Dễ lập trình hơn: Hỗ trợ nhiều ngôn ngữ: Scala, Java, Python, R, và cung cấp API thân thiện hơn MapReduce.

Tích hợp linh hoạt: Có thể chạy độc lập hoặc kết hợp với Hadoop (trên HDFS/YARN).

## Câu 2:

A trỏ B, C

B trỏ D

C trỏ A,B,D

D trỏ C

| | Vòng lặp 1 | Vòng lặp 2 | | |
|---|---|---|---|---|
| A | 1/1 + 1/3 = 4/3 | 1/1 + 2.5/3 | | |
| B | 1/1 = 1 | 0.33/1 | | |
| C | 1/2 + 1/1 + 1/1= 5/2 | 1.33/2 + 1/1+ 0.33/1 | | |
| D | 1/3 | 2.5/3 | | |

## Câu 3:

## a) Mục tiêu chính của K-Means (0.25 điểm)

Mục tiêu: chia n điểm dữ liệu thành K cụm sao cho tổng bình phương khoảng cách (intra-cluster variance) giữa các điểm và tâm (centroid) của cụm tương ứng là nhỏ nhất. Nói ngắn: gom những điểm gần nhau thành cùng một cụm.

## b) Các bước chính của thuật toán K-Means (0.75 điểm)

- Chọn K (số cụm) và khởi tạo K centroid (ngẫu nhiên hoặc theo quy tắc).
- Gán mỗi điểm cho cụm có centroid gần nhất (thường theo khoảng cách Euclid).
- Cập nhật centroid của mỗi cụm bằng trung bình tọa độ các điểm thuộc cụm đó.
- Lặp lại bước 2 và 3 cho đến khi hội tụ (không thay đổi phân cụm hoặc centroid ổn định) hoặc đạt ngưỡng vòng lặp.
  (Ghi chú: chuẩn hoá/chuẩn bị dữ liệu và chạy nhiều lần với khởi tạo khác nhau giúp tránh cực trị địa phương.)

## c) Bài toán cụ thể — K = 2, các điểm:

A(1,1), B(1.5,2), C(3,4), D(5,7), E(3.5,5), F(4.5,5), G(3.5,4.5), H(4,6)

Thực hiện K-Means với **khởi tạo centroid ban đầu** chọn là:

- $\text{Centroid}_1 = A = (1,1)$
- $\text{Centroid}_2 = D = (5,7)$

Mô tả **các vòng lặp** chính

*Vòng 1 — gán điểm theo centroid ban đầu*

Khoảng cách so sánh cho từng điểm → phân cụm:

- Cụm 1 ($\text{centroid}_1 = A$): A, B, C
- Cụm 2 ($\text{centroid}_2 = D$): D, E, F, G, H

(Tại vòng 1, C có khoảng cách bằng nhau với 2 centroid; ta gán theo quy tắc ≤ → về cụm 1.)

**Cập nhật centroid sau vòng 1** (lấy trung bình tọa độ các điểm trong cụm):

- $\text{Centroid}_1$ = trung bình(A,B,C) = (1+1.5+3)/3, (1+2+4)/3(1 + 1.5 + 3)/3,\ (1 + 2 + 4)/3(1+1.5+3)/3, (1+2+4)/3 = (1.8333, 2.3333)(1.8333,\ 2.3333)(1.8333, 2.3333)
- $\text{Centroid}_2$ = trung bình(D,E,F,G,H) = (5+3.5+4.5+3.5+4)/5, (7+5+5+4.5+6)/5(5 + 3.5 + 4.5 + 3.5 + 4)/5,\ (7 + 5 + 5 + 4.5 + 6)/5(5+3.5+4.5+3.5+4)/5, (7+5+5+4.5+6)/5 = (4.1, 5.5)(4.1,\ 5.5)(4.1, 5.5)

*Vòng 2 — gán lại theo centroid mới*

Tính khoảng cách tới hai centroid mới rồi gán:

- Cụm 1: A(1,1), B(1.5,2)
- Cụm 2: C, D, E, F, G, H

(Ở vòng 2, điểm C chuyển sang cụm 2 vì nó gần centroid₂ hơn.)

**Cập nhật centroid sau vòng 2**:

- Centroid₁ = trung bình(A,B) = (1+1.5)/2, (1+2)/2(1 + 1.5)/2,\ (1 + 2)/2(1+1.5)/2, (1+2)/2 = (1.25, 1.5)(1.25,\ 1.5)(1.25, 1.5)
- Centroid₂ = trung bình(C,D,E,F,G,H) = trung bình các tọa độ → (3.9167, 5.25)(3.9167,\ 5.25)(3.9167, 5.25) (6 điểm)

Vòng 3 — kiểm tra gán lại

Gán lại theo centroid mới:

- Kết quả vẫn giữ nguyên: Cụm 1 = {A, B}, Cụm 2 = {C, D, E, F, G, H}

Centroid không đổi so với vòng 2 → **hội tụ**.

Kết quả cuối cùng (K = 2)

- **Cụm 1 (Cluster 1)**: A(1,1), B(1.5,2)
  - Centroid ≈ **(1.25, 1.50)**
- **Cụm 2 (Cluster 2)**: C(3,4), D(5,7), E(3.5,5), F(4.5,5), G(3.5,4.5), H(4,6)
  - Centroid ≈ **(3.9167, 5.25)**

(Thuật toán hội tụ sau 2 lần cập nhật centroid — tức sau ~2 vòng gán/cập nhật.)