MỘT SỐ THUẬT TOÁN XỬ LÝ DỮ LIỆU LỚN

Giảng viên: Nguyễn Tu Trung, Trần Mạnh Tuấn
BM HTTT, Khoa CNTT, Trường ĐH Thủy Lợi

Hà Nội, 2019

Nội dung

- Thuật toán K-Means
- Xây dựng thuật toán MapReduce_K-Means
- Thuật toán Naïve Bayes
- Xây dựng thuật toán MapReduce_Bayes

Thuật toán K-Means

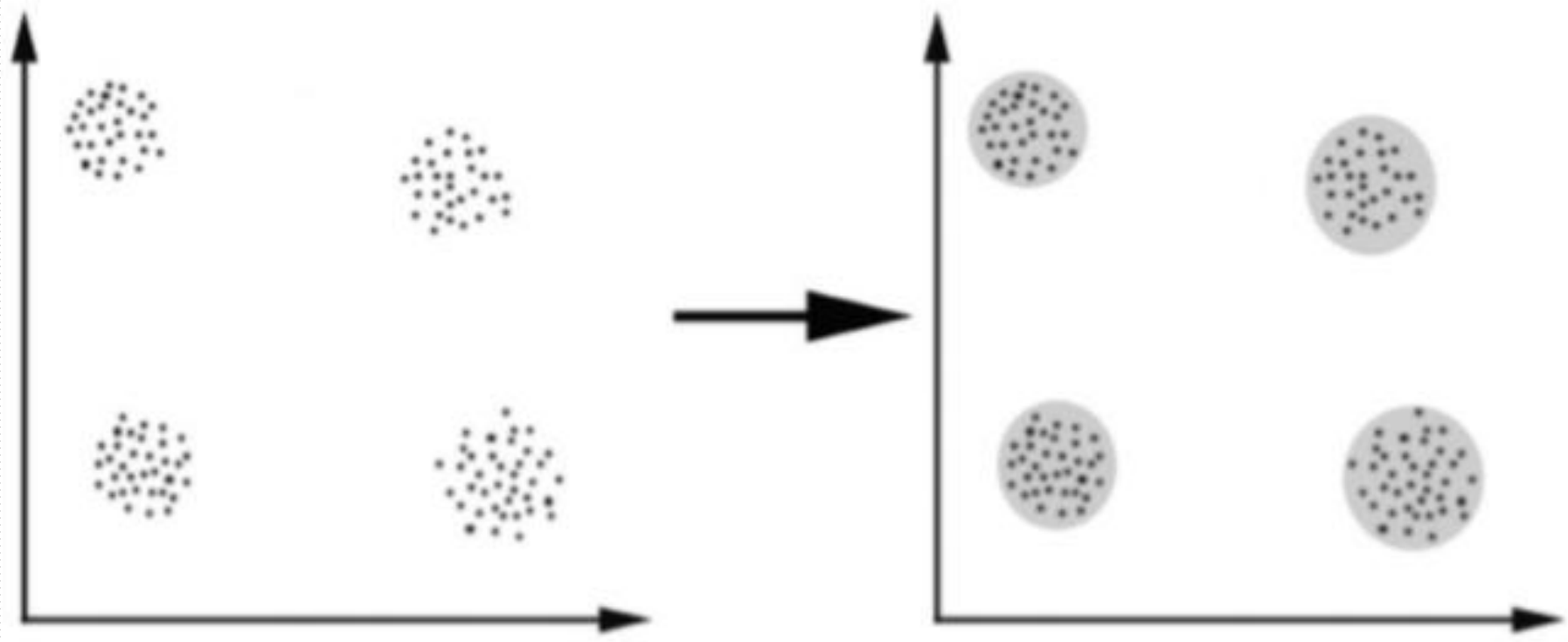
- Giới thiệu thuật toán K-Means
- Phát biểu bài toán phân cụm
- Các bước thuật toán K-Means
- Điều kiện dừng và chất lượng phân cụm
- Nhận xét thuật toán K-Means

Giới thiệu thuật toán K-Means

- Là một trong các thuật toán phân cụm đơn giản và điển hình nhất
- Do MacQueen đề xuất trong lĩnh vực thống kê năm 1967
- Mục đích:
  - Sinh ra k cụm dữ liệu từ một tập dữ liệu ban đầu gồm n đối tượng trong không gian p chiều $X_i = \{x_{i1}, x_{i2}, \ldots, x_{ip}\}$, i = 1..n, sao cho hàm tiêu chuẩn E đạt giá trị tối thiểu
    - $E = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, m_i)^2$
  - m_i là vector trọng tâm của cụm C_i, giá trị của mỗi phần tử là trung bình cộng các thành phần tương ứng của các đối tượng vector dữ liệu trong cụm đang xét
  - d là khoảng cách Euclide giữa hai đối tượng

Phát biểu bài toán phân cụm

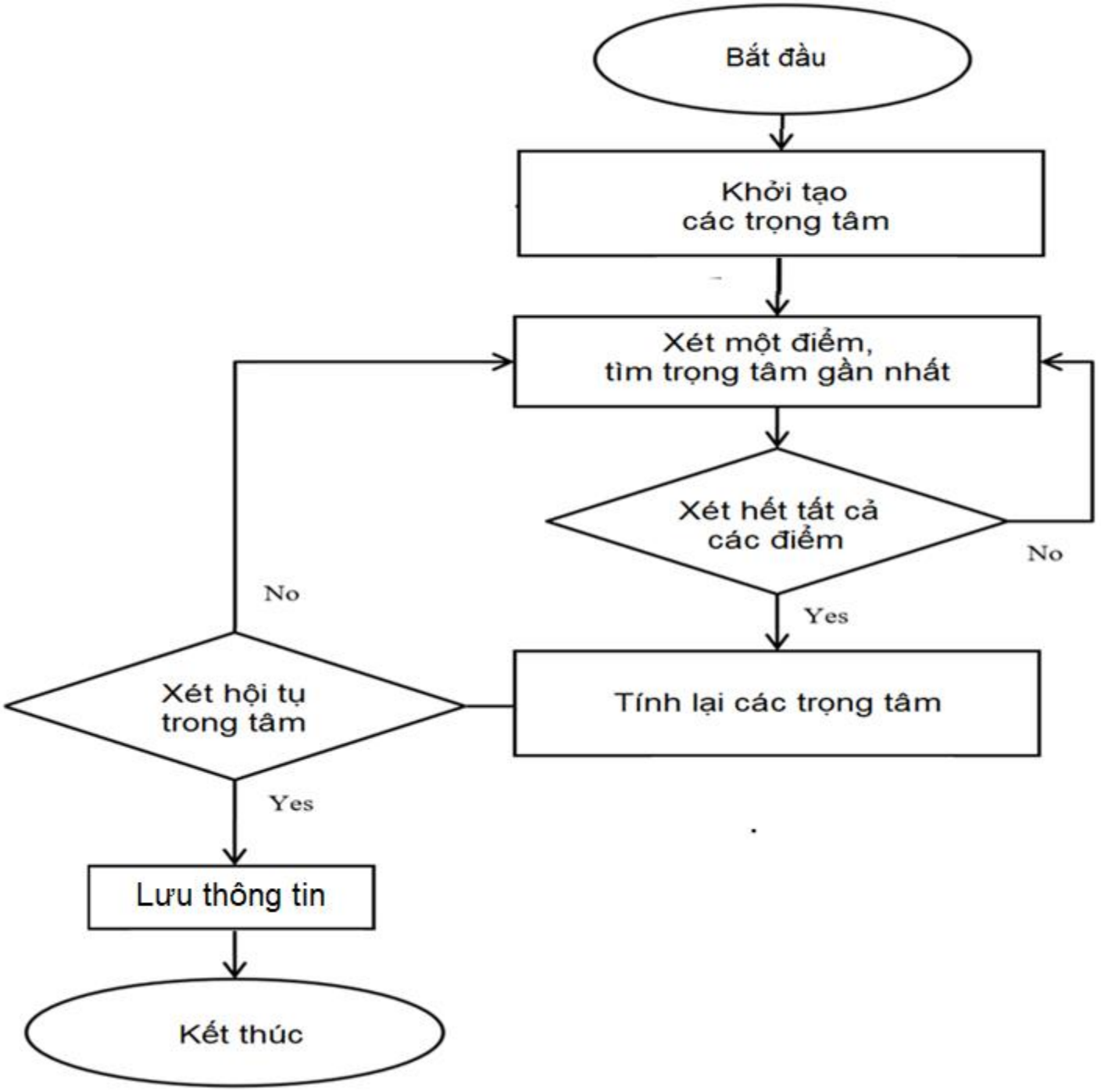
- Input: n đối tượng và số các cụm k
- Output: Các cụm C_i (i=1... k) sao cho hàm tiêu chuẩn E đạt giá trị tối thiểu



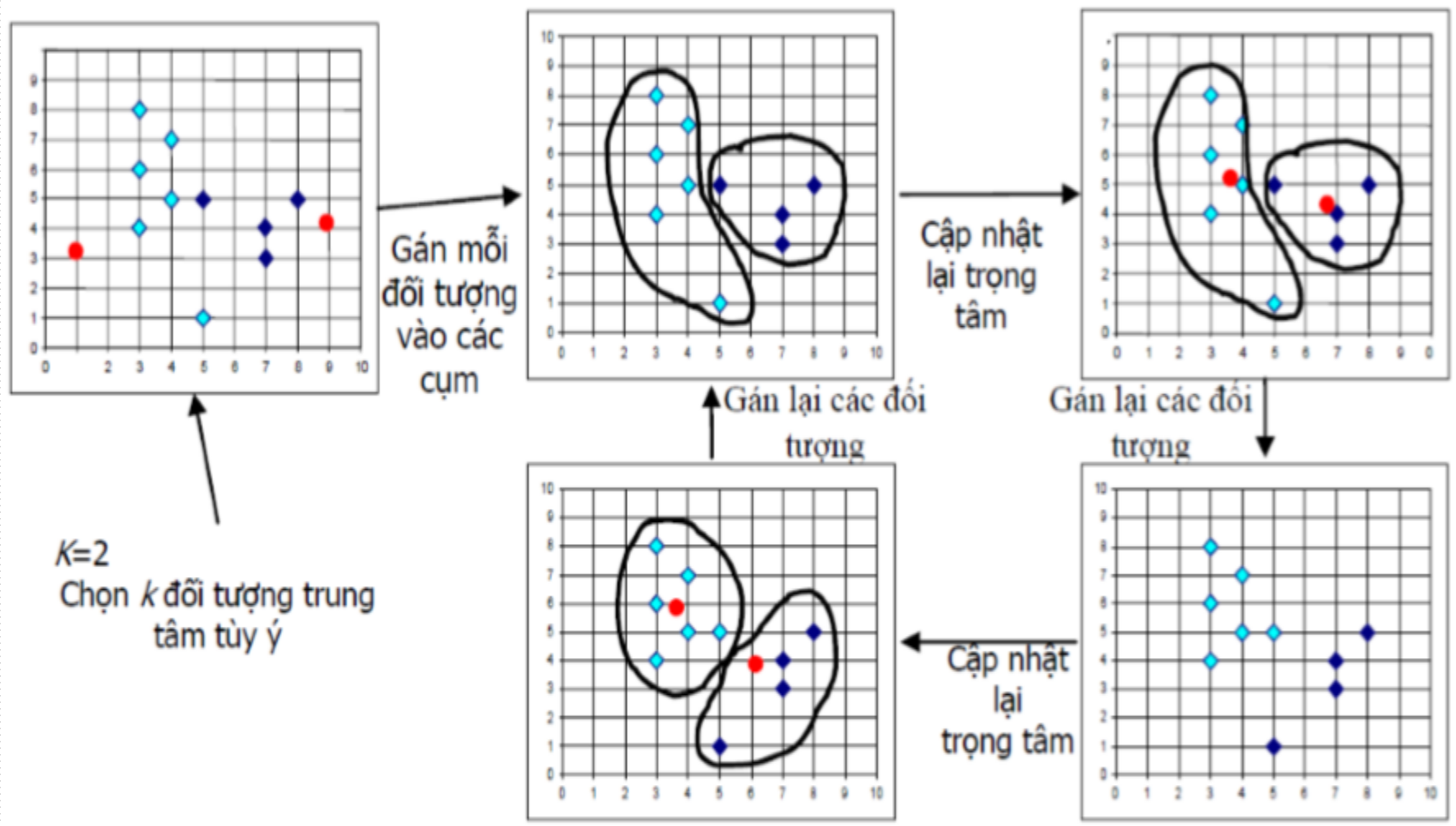
Các bước thuật toán K-Means

- Bước 1: Khởi tạo tâm cụm
  - Chọn k đối tượng m_j (j=1...k) là trọng tâm ban đầu của k cụm từ tập dữ liệu (Việc lựa chọn này có thể là ngẫu nhiên hoặc theo kinh nghiệm)
- Bước 2: Tính toán khoảng cách và gán cụm
  - Với mỗi đối tượng Xi (1 i n), tính toán khoảng cách từ nó tới mỗi trọng tâm mj với j=1,..,k, sau đó tìm trọng tâm gần nhất đối với mỗi đối tượng
- Bước 3: Cập nhật lại trọng tâm
  - Với mỗi j=1,..,k, cập nhật trọng tâm cụm mj bằng cách xác định trung bình cộng của các vector đối tượng dữ liệu
- Bước 4: Kiểm tra điều kiện dừng
  - Lặp các bước 2 và 3 cho đến khi các trọng tâm của cụm không thay đổi

Lưu đồ thuật toán K-Means



Minh họa quá trình phân cụm



Điều kiện dừng và chất lượng phân cụm

- Điều kiện dừng
  - Không có (hoặc có không đáng kể) việc gán lại các ví dụ vào các cụm khác
  - Không có (hoặc có không đáng kể) thay đổi về các điểm trung tâm (centroids) của các cụm
  - Không giảm hoặc giảm không đáng kể về tổng lỗi phân cụm E
- Chất lượng phân cụm
  - Phụ thuộc nhiều vào các tham số đầu vào như: số cụm k và k trọng tâm khởi tạo ban đầu
  - Nếu các trọng tâm khởi tạo ban đầu quá lệch so với các trọng tâm cụm tự nhiên thì kết quả phân cụm của k-means là rất thấp => các cụm dữ liệu được khám phá rất lệch so với các cụm trong thực tế

Nhận xét thuật toán K-Means

- Phần lớn khối lượng tính toán tập trung ở bước 2: tính khoảng cách từ mỗi điểm (đối tượng) tới các tâm cụm
- Số lượng đối tượng trong tập dữ liệu càng lớn, thời gian cần cho bước này càng nhiều
- Việc tính toán khoảng cách từ một điểm tới tâm cụm là độc lập, không phụ thuộc vào điểm khác
- => Việc tính khoảng cách từ các điểm đến các tâm cụm có thể thực hiện song song, đồng thời với nhau

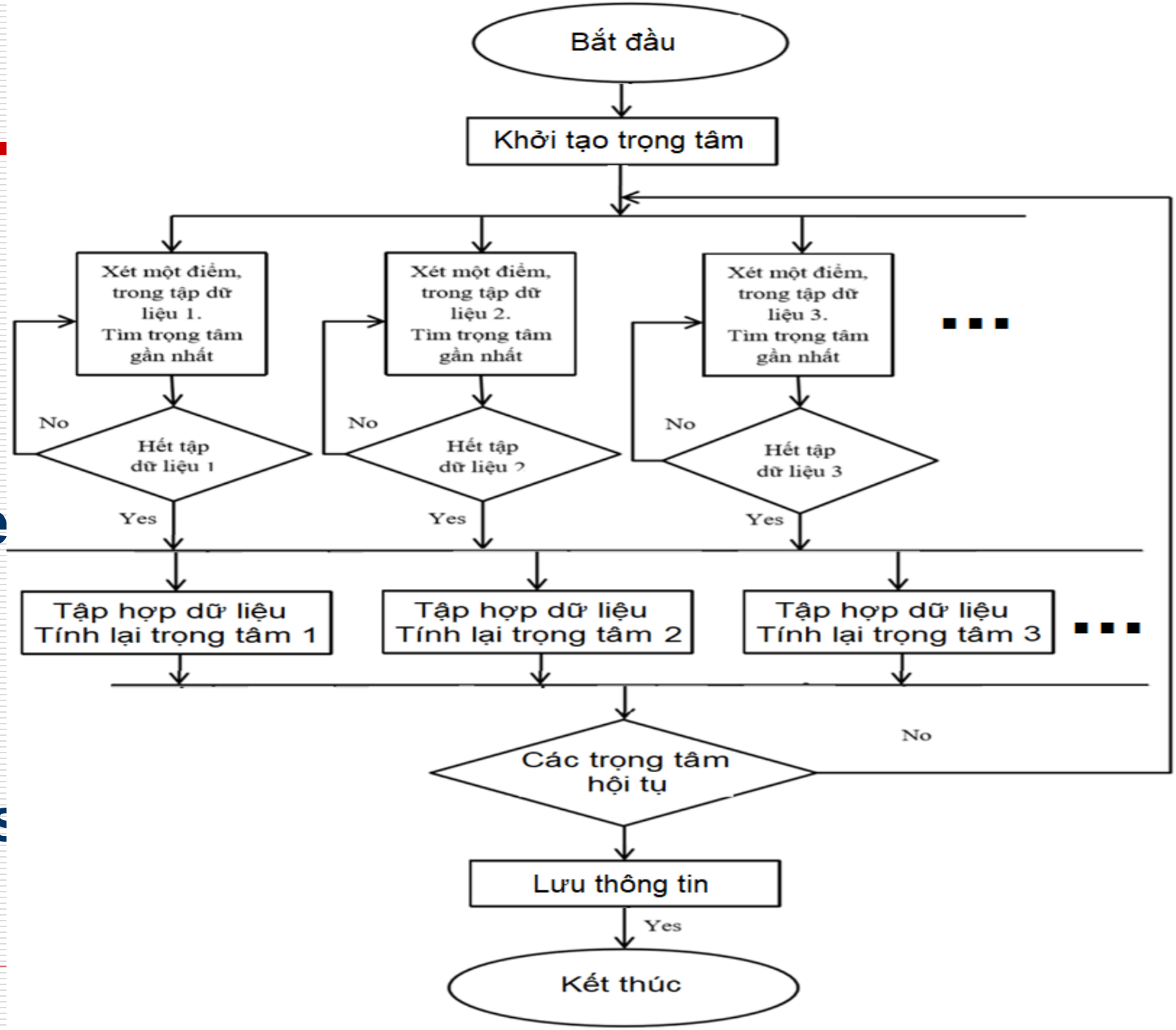
Xây dựng thuật toán MapReduce_K-Means

- Ý tưởng MapReduce hóa K-Means
- Lưu đồ MapReduce hóa thuật toán K-Means
- Giải pháp MapReduce hóa thuật toán K-Means
- Xây dựng hàm Map_K-Means
- Xây dựng hàm Reduce_K-Means
- Mã nguồn MapReduce_K-Means

Ý tưởng MapReduce hóa K-Means

- Tách dữ liệu thành các nhóm nhỏ
- Với mỗi vòng lặp
  - Map:
    - Phân cụm trên từng nhóm nhỏ dữ liệu
    - Với mỗi điểm dữ liệu, tìm trọng tâm gần nhất
  - Tất cả dữ liệu được gom theo từng tâm
  - Reduce:
    - Tính tâm mới của các dữ liệu được gom (theo từng tâm)

Lưu đồ MapReduce thuật toán K-Means



Giải pháp MapReduce hóa K-Means

- Đầu tiên: biểu diễn dữ liệu
  - Dữ liệu lưu trữ dưới dạng list các hàng
  - Mỗi hàng là list giá trị là các thành phần của vector biểu diễn cho một điểm
- Thứ hai: lưu trữ phân tán dữ liệu
  - Do các điểm được tính toán độc lập với nhau => có thể lưu trữ các phần của dữ liệu trên nhiều máy khác nhau để có thể xử lý song song và tăng tốc tính toán
- Thứ ba, trong mỗi vòng lặp
  - B1: Tính khoảng cách của mỗi điểm trong phần dữ liệu của nó với các trọng tâm
  - B2: Kiểm tra xem điểm đó gần trọng tâm nào nhất
  - B3: Gom các điểm thuộc cùng một cụm để tính lại trọng tâm sau mỗi vòng lặp

Giải pháp MapReduce hóa K-Means

- Dữ liệu cần phân cụm là danh sách các hàng (có thể lưu trên file txt) được chuyển sang kiểu key/value làm đầu vào cho thuật toán
- Mô hình cơ bản của MapReduce:
  - map (keyIn, valIn) -> list (keyInt, valInt)
  - reduce (keyInt, list (valInt)) -> list (keyOut, valOut)
- Áp dụng cho K-Means:
  - Xây dựng hàm Map_K-Means
  - Xây dựng hàm Reduce_K-Means

Xây dựng hàm Map_K-Means

- Đầu vào: cặp key/value biểu diễn toạ độ của một điểm
  - keyIn là giá trị bye offset của dòng
  - valIn là vector biểu diễn toạ độ của một điểm
- Xử lý:
  - Tính khoảng cách của điểm với các trọng tâm (chưa phải là trọng tâm cần tìm)
  - Chuyển về cụm có tâm gần nhất
- Đầu ra: cặp key/value trung gian
  - keyInt là tâm gần nhất (trọng tâm hoặc chỉ số tâm)
  - valInt là toạ độ điểm thuộc cụm có trọng tâm là keyInt

Xây dựng hàm Reduce_K-Means

- Trước khi hàm reduce thực hiện
  - Kết quả của hàm map được trộn lại
  - Các cặp cùng keyInt được gom thành một nhóm
- Đầu vào:
  - keyInt được chuyển từ hàm map
  - list(valInt) là list các điểm valInt thuộc về cụm thứ keyInt
- Xử lý:
  - Tính trung bình cộng từng thành phần của các điểm cùng cụm
  - Cập nhật lại trọng tâm của cụm đó
- Đầu ra:
  - keyOut là Tâm mới
  - valOut là Danh sách các điểm

Mã nguồn MapReduce_K-Means

- Thảo luận mã nguồn chương trình…