Nhóm 9:

Nguyễn Thái Sơn - 23521356
Bùi Ngọc Thiên Thanh - 23521436

Bài 1:

1) **Phân tích**

Bước 1 : Giai đoạn khởi tạo (init):

- Với mỗi phần tử a trong bảng chữ cái α, tạo ra một cây T_a chứa một nút duy nhất. Chi phí cho việc tạo ra T_a là O(1).
- Sau đó, thêm T_a vào danh sách F, có chi phí là O(1).
- Tổng chi phí cho giai đoạn khởi tạo là O(n). với n là số lượng phần tử trong bảng chữ cái.

Bước 2 : Giai đoạn vòng lặp chính

1. **Vòng lặp chính tiếp tục cho đến khi danh sách F chỉ còn một cây**
2. **Trong mỗi lần lặp, ta thực hiện:**
    - Chọn hai cây T_1 và T_2 có xác suất nhỏ nhất, mất chi phí O(logn)O cho mỗi lần chọn (nếu sử dụng hàng đợi ưu tiên).
    - Gộp T_1 và T_2 thành một cây T_3, mất chi phí O(1)
    - Cập nhật xác suất cho T_3 và thêm T_3 vào danh sách F, mất chi phí O(logn).
3. **Vì vòng lặp này thực hiện n−1 lần (tương ứng với n−1 lần gộp cây), nên tổng chi phí là O(nlogn).**

Độ phức tạp tổng quát của thuật toán: O(nlogn)

2) Cải tiến tối ưu

Team mình có đề xuất sử dụng hàng đợi ưu tiên (priority queue) hoặc một cấu trúc dữ liệu tương tự. giúp thực hiện các thao tác chọn và thêm một cách hiệu quả hơn so với các cấu trúc dữ liệu khác.

Bài 2:

1) PRIM

Code : https://www.ideone.com/oWcRxa

Phân tích các bước thực hiện

- Trong vòng lặp, thuật toán lấy đỉnh có khoảng cách nhỏ nhất từ hàng đợi
- Nếu đỉnh này đã được xử lý (khoảng cách của nó trong hàng đợi không khớp với khoảng cách hiện tại của nó trong mảng dis[]), thuật toán bỏ qua nó.

- Nếu không, đỉnh này sẽ được thêm vào cây khung nhỏ nhất, và giá trị ret sẽ tăng thêm trọng số của cạnh kết nối đến đỉnh này.
- Sau khi kết nạp đỉnh vào cây khung, thuật toán cập nhật khoảng cách của các đỉnh kề với đỉnh đó. Nếu khoảng cách từ đỉnh vừa kết nạp đến một đỉnh kề nhỏ hơn khoảng cách hiện tại trong mảng dis[], thuật toán sẽ cập nhật giá trị này và đẩy nó vào hàng đợi ưu tiên.

Độ phức tạp :

- Vòng lặp chính chạy cho đến khi hàng đợi ưu tiên trống. Mỗi đỉnh được đưa vào cây khung một lần và mỗi cạnh được xét đúng một lần.

- Đối với mỗi cạnh, nếu khoảng cách đến đỉnh đích nhỏ hơn khoảng cách hiện tại, ta sẽ
cập nhật và thêm đỉnh vào hàng đợi ưu tiên.

- Thao tác pop từ hàng đợi và thao tác push vào hàng đợi đều có độ phức tạp O(log n) vì hàng đợi ưu tiên có kích thước tối đa là n.

- Việc duyệt qua các đỉnh và các cạnh kề có độ phức tạp là O(m) (vì mỗi cạnh chỉ được duyệt qua một lần).
- Đối với mỗi đỉnh, ta thực hiện thao tác push và pop từ hàng đợi ưu tiên. Do đó, chi phí cho các thao tác này là O(log n).
- Với mỗi cạnh, ta thực hiện một thao tác push hoặc cập nhật trong hàng đợi ưu tiên, có độ phức tạp là O(log n).
- Tổng độ phức tạp của thuật toán Prim với hàng đợi ưu tiên là:
- O(mlogn)

2) Kruskal

Code : https://www.ideone.com/uUN5VJ

Phân tích các bước thực hiện

- Sắp xếp tất cả các cạnh theo thứ tự tăng dần của trọng số.
- Khởi tạo cây khung rỗng.
- Lần lượt duyệt qua các cạnh, kiểm tra xem cạnh đó có tạo thành chu trình với các cạnh đã chọn hay không (bằng cách sử dụng thuật toán Union-Find - Hợp-Nối).
- Nếu không tạo thành chu trình, thêm cạnh đó vào cây khung.

Độ phức tạp:

- Bước sắp xếp cạnh có độ phức tạp O(E log E).
- Union-Find có thể thực hiện với độ phức tạp gần O(E) nếu sử dụng Union by Rank và Path Compression.
- Tổng thể, độ phức tạp của thuật toán Kruskal là O(E log E). Vì E log E gần tương đương với E log V khi E >= V, nên độ phức tạp có thể viết thành O(E log V).
- Thuật toán Kruskal: O(E log V)