1. ***Spanning tree***

- Định nghĩa: cho 1 đồ thị G là 1 đồ thị liên thông gồm n đỉnh, cây khung là 1 tập con của đồ thị mà đảm bảo các tiêu chí như sau:

* Gồm n đỉnh
* N đỉnh vẫn liên thông với nhau
* Không có bất cứ chu trình nào (điều này là hiển nhiên vì cây thì không thể có chu trình được)
* Có số số cạnh là n-1, nếu ít hơn n-1 cạnh thì không liên thông mà nếu nhiều hơn n-1 thì sẽ tạo thành chu trình

1. ***Minimum spanning tree(cây khung nhỏ nhất)***

* Định nghĩa: là 1 cây khung của 1 đồ thị liên thông có trọng số với tổng trọng số của các cạnh là nhỏ nhất so với các cây khung khác được tạo nên từ cùng 1 đồ thị.

1. ***Prim's algorithm***

* Về thuật toán Prim, tư tưởng của thuật toán này là khi mà khởi tạo thì cây chỉ chứa đúng 1 đỉnh được coi là gốc, đỉnh này có thể là bất kì đỉnh nào ở trong đồ thị cha của nó. Cây sẽ được mở rộng từng bước một, mỗi bước sẽ thêm 1 cạnh mới vào cây cho tới khi bao trùm được tất cả các đỉnh của đồ thị ban đầu thì dừng lại. Cụ thể hơn về thuật toán prim như sau:
* Input của thuật toán là 1 đồ thị liên thông có trọng số G với số số đỉnh là n và tập hợp các cạnh là E (lưu ý ở đây là trọng số của cạnh có thể âm). Bắt đầu thuật toán thì ta khởi tạo 1 cây mới chứa 1 đỉnh bất kì từ đồ thị G. Sau đó thì thực hiện vòng lặp với mỗi vòng lặp thực hiện những công việc như sau: tìm cạnh e là cạnh gần với cây nhất (cạnh có trọng số nhỏ nhất) nối giữa 1 đỉnh chưa ở trong cây và 1 đỉnh đã ở trong cây rồi với điều kiện là cạnh này không được phép tạo thành chu trình ở trong cây, sau đó là việc thêm cạnh thỏa mãn các điều kiện mình vừa mới nói vào cây. Sau khi lặp công việc trên n-1 lần thì ta sẽ được 1 cây khung nhỏ nhất bằng cách sử dụng thuật toán prim. Ví dụ minh họa cho thuật toán thì mình sử dụng đồ thị như trên slide, khi bắt đầu thì mình chọn 1 đỉnh bất kì và ở đây là mình chọn đỉnh b, với lần lặp đầu tiên sẽ tìm cạnh gần đỉnh b nhất trong số những cạnh kề với đỉnh b là a,c và f. Hiển nhiên ở đây đỉnh gần với đỉnh b nhất là đỉnh f vì nó có trọng số là 1 nhỏ hơn 2 và 3. Sau lần lặp đầu tiên thì cây của mình sẽ có cạnh bf. Tiếp tục với vòng lặp thứ 2 thì mình sẽ lại tiếp tục tìm kiếm những đỉnh gần cây nhất và cứ tiếp tục như vậy đến khi số đỉnh của cây khung bằng với số đỉnh của đồ thị G ban đầu thì dừng vòng lặp lại.
* Về độ phức tạp của thuật toán này: ta duyệt tổng cộng n lần với n là số đỉnh của đồ thị, mỗi lần lấy 1 đỉnh ra khỏi bộ nhớ heap mất O(logn) và cập nhật trọng số của tất cả các đỉnh kề với đỉnh đó là mất m lần với m là số cạnh của đồ thị mỗi lần cập nhật như vậy có độ phức tạp là O(logn). Như vậy độ phức tạp của thuật toán Prim là O((m+n)logn)

1. ***Kruskal's algorithm***

Về tư tưởng của thuật toán: Kruskal không xét các cạnh 1 cách tùy ý mà xét theo thứ tự các cạnh đã sắp xếp theo trọng số. Cụ thể hơn về thuật toán như sau:

Giả sử ta cần tìm cây khung nhỏ nhất của đồ thị liên thông có trọng số G. Thuật toán bao gồm các bước như sau:

Bước 1 khởi tạo cây khung T hiện đang rỗng.

Bước 2 là thực hiện vòng lặp với mỗi vòng lặp thêm 1 cạnh có trọng số nhỏ nhất trong đồ thị G với điều kiện là cạnh đó không được tạo thành chu trình ở trong cây khung T, trong trường hợp có 2 hoặc lớn hơn 2 đều có cùng trọng số nhỏ nhất thì chọn bất kì 1 trong số các cạnh đó

Khi kết thúc thuật toán ta sẽ thu được cây khung nhỏ nhất của đồ thị G

Đây là ví dụ minh họa sử dụng kruskal. Như mình vừa trình bày thì nó sẽ chọn các cạnh có trọng số nhỏ nhất và thêm cạnh đó vào đồ thị qua từng vòng lặp. Ở bảng bên trái này là kết quả qua từng vòng lặp. và kết quả của thuật toán này cũng tương tự như thuật toán prim vừa rồi, tổng trọng số của cây khung cũng là 24

Độ phức tạp của thuật toán kruskal này là mlogm cộng với mlogn

Tiếp theo là phần so sánh 2 thuật toán prim và kruskal

Về sự giống nhau thì 2 giải thuật này đều là giải thuật tham trong việc tìm cây khung nhỏ nhất của đồ thị đơn liên thông có trọng số

Về sự khác biệt thì kruskal sẽ khởi tạo với 1 cạnh còn prim khởi tạo với 1 đỉnh bất kì và Thuật toán Prim chỉ thực hiện được trên đồ thị liên thông còn kruskal thì đồ thị không cần liên thông cũng có thể tìm ra cây khung nhỏ nhất được