**Bai 2**

## Thuật toán Kruskal

### *Ý tưởng:*

Tương tự như thuật toán prim gọi T là cây bao trùm tối thiểu cần tìm. Như vậy T là tập hợp một số cung của đồ thị G sao cho T cũng là đồ thị liên thông và tổng giá trị trong các cung là nhỏ nhất**.**

* Sắp xếp danh sách cung đầu vào theo trọng số của cung thứ tự tăng dần.
* Ban đầu khởi tạo cho T là một cung nhỏ nhất trong tập cung đã sắp xếp(là cung đầu tiên trong danh sách)
* Trong quá trình xây dựng cây bao trùm nhỏ nhất ta luôn có một rừng cây.
* Thêm dần vào T các cạnh có chi phí nhỏ nhất trong số các cạnh chưa được thêm.
* Như vậy khi thêm một cạnh có thể:

- Tạo thêm cây mới.

- Mở rộng cây đã có trong T.

- Nối hai cây thành một cây mới.

- Thuật toán dừng lại khi tất cả các đỉnh được đánh dấu.

### *Cài đặt:*

*Cấu trúc dữ liệu:*

* Ta khai báo các thành phần của cung:

int dau, // đỉnh đầu của cung

cuoi, // đỉnh cuối của cung

trs; // trọng số của cung

Sau đó xây dựng một mảng cấu trúc tapcanh[] để chứa toàn bộ thông tin của đồ thị để phục vụ cho tính toán.

* Dùng các mảng dd[], ddc[] lần lượt đánh dấu một đỉnh, cung đã được chọn chưa. dd[i], ddc[i] bằng 1 nếu đã chọn và bằng 0 nếu chưa chọn.
* Dùng biến d để đặt số hiệu cho từng cây trong rừng.

Sử dụng thuật toán Heap Sort để sắp xếp mảng cấu trúc theo giá trị tăng dần của trọng số cung. Sau khi sắp xếp ta cài đặt thuật toán kruskal

### *Đánh giá thuật toán:*

Với thuật giải nay bào toán sẽ tốn thời gian lớn nhất ở hai hàm sắp xếp và hàm kruskal

-Với cách sử dụng heap để sắp xếp cho một mảng kich thước la m là O(m log m)

-Với sử dụng giải thuật Kruskal thời gian tối đa dành cho thuật giải là O(m log n).

Như vậy tùy theo số đỉnh n và số cạnh m vào mà có thời gian có thể là O(m logm) hoặc O(m log n).

* 1. **Bài toán “Tìm đường đi ngắn nhất trong đồ thị có trọng số” – Thuật toán Dijkstra.**

**3.1.1 Bài toán**

Cho G = (V,E) là đồ thị liên thông ( vô hướng ) có trọng số

V = {1…n} là tập các đỉnh , E là tập các cạnh (cung).

Trọng số (i,j) là w(i,j) >0 và đỉnh x sẽ mang nhãn L(x). Khi kết thúc thuật giải thì L(x) chính là độ dài ngắn nhất từ a đến z.

Input: Đồ thị liên thông G=(V,E,w) có trọng số w(i,j)>0 với mọi cạnh (i,j), đỉnh a và z.

Output: L(z) chiều dài đường đi ngắn nhất từ a đến z, và đường đi ngắn nhất.

**3.1.2 Ý tưởng**

- Thuật toán Dijkstra cho phép tìm đường đi ngắn nhất từ một đỉnh a đến các đỉnh còn lại của đồ thị và chiều dài (trọng số ) tương ứng.

- Phương pháp của thuật toán là xác định tuần tự đỉnh có chiều dài từ a theo tứ tự tăng dần.

- Thuật toán được xây dựng trên cơ sở gán cho mỗi đỉnh các nhãn tạm thời. Nhãn tạm thời của các đỉnh cho biết cận trên của chiều dài đường đi ngắn nhất từ a đến các đỉnh đó. Nhãn của đỉnh sẽ được biến đổi trong các bước lặp, mà ở mỗi bước lặp sẽ có một nhãn tạm thời trở thành chính thức. Nếu nhãn của một đỉnh nào đó trở thành chính thức thì đó cũng chính là chiều dài ngắn nhất của đường đi từ a đến đỉnh đó.

**3.1.3 Mô tả thuật toán**

1- Khởi tạo: L(a):=0. Với mọi đỉnh x ≠ a thì L(x) = ∞.

Ký hiệu T := V

2- Chọn v ∈ T sao cho L(v) có giá trị nhỏ nhất, tức là:

L(v):=min {L(u) | u ∈ T}

Đặt: T := T - {v}

3- Nếu z = v, kết thúc, L(z) là chiều dài đường đi ngắn nhất từ a đến z

Từ z lần ngược theo đỉnh được ghi nhớ ta có đường đi ngắn nhất.

Ngược lại nếu z ≠ v, sang bước (4).

Nếu giá trị nhỏ nhất ở bước (2) là ∞ , kết thúc, kết luận không tồn tại đường đi từ a đến z.

4- Với mỗi x∈T kề v nếu L(x)>L(v) + w(v,x), thì

L(x) := L(v)+w(v,x)

Và ghi nhớ đỉnh v cạnh đỉnh x để sau này xây dựng đường đi ngắn nhất.

Quay về bước (2)

**3.1.4 Độ phức tạp của thuật toán:**

Thuật toán được mô tả bởi 2 vòng lặp lồng nhau, T(n) = O(n2)

## 3.2 Balo tham lam

### 1) Ý tưởng

Martello và Toth (1990) đã đưa ra một thuật toán gần đúng kiểu tham lam (*greedy approximation algorithm*) để giải bài toán xếp ba lô. Giải thuật này sắp xếp các đồ vật theo thứ tự giảm dần về giá trị, sau đó theo thứ tự đó xếp các đồ vật vào ba lô cho đến khi không cho thêm được đồ vật nào vào nữa.

### 2)Giải quyết

Xây dựng 1 mảng "đơn giá" dg có độ dài n với dg[i]=v[i]/m[i] rồi lần lượt chọn những đồ vật có đơn giá cao nhất cho đến khi ko thể chọn đồ vật này nữa thì sẽ chọn tới đồ vật có đơn giá thấp hơn cho tơi khi túi đầy

Giải thuật tham lam giải bài toán xếp balô dựa trên chiến lược "chọn cái tốt

nhất trước". Việc chọn các vật đưa vào balô có thể theo 3 chiến lược nh ư sau:

- Ưu tiên vật nhẹ trước (với hi vọng chọn được nhi ều đồ vật).

- Ưu tiên vật có giá trị cao trước.

- Ưu tiên chọn những vật có tỉ số giá trị/trọng lượng lớn trước.

Ta thấy chiến lược thứ 3 tổng quát hơn nhất. Do vậy việc tìm nghiệm của

chúng ta sẽ tiến hành theo 2 bước:

1. Sắp xếp các đồ vật giả m dần theo tỉ lệ: giá trị/trọng lượng.

2. Lần lượt đưa vào balô những đồ v ật nào có thể đưa đượ c theo trình tự đã sắp xếp đó.

Thuật giải chi tiết như sau:

Procedure Gready;

Begin

for i := 1 to n do

Begin

a[i]:=v[i]/w[i]; { sắp xếp w,v,a,id theo a; }

id[i] := i;

End;

for i:=1 to n do

Begin

if m >= w[i] then

Begin

m := m - w[i];

t := t + v[i];

s := s + [id[i]];

End;

End;

End;

Kết quả: S là tập các đồ vật được chọn, T là tổng giá trị của chúng. Thuật

giải này có độ phức tạp O(nlogn) do thao tác chủ yếu ở phần sắp xếp.

**Ví dụ** : Có ba lô có trọng lượng 37 và 4 loại đồ vật như bảng:

Loại đồvật Trọng lượng Giá trị

A 15 30

B 10 25

C 2 2

D 4 6

Loại đồ vật Trọng lượng Giá trị Đơn giá (Giá trị/Trọng lượng)

B 10 25 2.5

A 15 30 2.0

D 4 6 1.5

C 2 2 1.0

- Theo bảng thứ tự ưu tiên là B,A,D và C:

– Vật B,chọn tối đa là 3 cái,vì mỗi cái trọng lượng là 10 ,trọng lượng ba lô còn lại la 7.

– Vật A không chọn được vì trọng lượng vật A là 15 Vật A , không chọn được vì trọng lượng vật A là 15 trong khi ba lô chỉ còn 7.

– Vật D chọn được 1 cái, trọng lượng ba lô còn lại = 3.

– Vật C chọn được 1 cái.

Như vậy chúng ta đã chọn 3 cái loại B, một cái loại D và 1 cái loại C. Tổng trọng lượng là 3\*10 + 1\*4 + 1\*2 = 36 và tổng giá trị là 3\*25+1\*6+1\*2 = 83.

### 3)Đánh giá thuật toán

giải thuật này có *độ phức tạp O(nlogn) thời gian*