***Bài 2:*** *Trình bày Kruskal để giải bài toán tìm cây bao trùm tối thiểu của đồ thị: Ý tưởng phương pháp, lược đồ thuật toán, chứng minh tính đúng; đánh giá độ phức tạp thuật toán.*

Cho đồ thị vô hướng liên thông trong đó là tập các đỉnh , là tập các cạnh. Mỗi cạnh e của đồ thị được gắn với một số thực ,gọi là độ dài của nó.Giả sử là cây bao trùm nhỏ nhất của đồ thị tức là cây chứa tất cả các đỉnh của đồ thị G và có tổng độ dài các cạnh là nhỏ nhất.Sau đây, sẽ là ý tưởng phương pháp, lược đồ thuật toán, chứng minh tính đúng , độ phức tạp của thuật toán Krusal để xây dựng cây bao trùm tối thiểu H của đồ thị G.

* Ý tưởng phương pháp:
* Thuật toán sẽ xây dựng tập cạnh T của cây bao trùm nhỏ nhất theo từng bước:
* Sắp xếp các cạnh của đồ thị G theo thứ tự tăng dần của độ dài
* Bắt đầu từ , ở mỗi bước ta sẽ duyệt các cạnh đã được sắp xếp theo thứ tự tăng dần của độ dài, từ cạnh có độ dài nhỏ cho đến cạnh có độ dài lớn hơn, để tìm ra cạnh mà việc bổ sung nó vào T không tạo thành chu trình trong tập này
* Thuật toán sẽ kết thúc khi ta thu được tập T gồm n-1 cạnh của đồ thị G.
* Lược đồ thuật toán:
* Mã giả:

**ProcedureKrusal**

**Begin**

;

While() do

**Begin**

Chọn e là cạnh có độ dài nhỏ nhất trong E;

E:=E\{e};

If( không chứa chu trình) then {e};

**End**

if then Đồ thị không liên thông

**End.**

* Chứng minh tính đúng:
* Chứng minh thuật toán sinh cây bao trùm:
* Tập cạnh T không có chu trình, nhưng nếu thêm một cạnh bất kỳ giữa hai đỉnh (cạnh này luôn có) sẽ tạo nên chu trình vì nếu không có thì quá trình lặp chưa kết thúc. Do đó theo tính chất T không có chu trình, nhưng nếu thêm một cạnh nối hai đỉnh bất kỳ không kề nhau thì xuất hiện một chu trình, T là một cây.
* Mặt khác, mỗi đỉnh của đồ thị G đều kề với tập cạnh T vì nếu không thì còn có thể thêm cạnh nữa vào T. Vậy T liên thông và có n-1 cạnhSuy ra cây T là cây bao trùm của đồ thị G.
* Chứng minh cây bao trùm sinh được là nhỏ nhất:
* Ta chứng minh bằng phản chứng.Giả sử tồn tại cây S của đồ thị mà .Ký hiệu là cạnh đầu tiên trong dãy các cạnh của T xây dựng theo thuật toán vừa mô tả không thuộc S.Khi đó đồ thị con của G sinh bởi cây S được bổ sung cạnh sẽ chứa một chu trình duy nhất C đi qua .Do chu trình C phải chứa cạnh e thuộc S nhưng không thuộc T nên đồ thị con thu được từ S bằng cách thay cạnh e của nó bằng cạnh là S’ sẽ là cây bao trùm.Theo cách xây dựng do đó , đồng thời số cạnh chung của và T đã tăng thêm một so với số cạnh chung của S và T.Lặp lại quá trình trên từng bước một ta có thể biến đổi S thành T và trong mỗi bước tổng độ dài không tăng, tức là (Mâu thuẫn).Vậy chứng tỏ T là cây bao trùm bé nhất.
* Độ phức tạp của thuật toán:

Đầu tiên phải sắp xếp các cung theo thứ tự tăng dần của trọng số. Nếu dùng Quick sort, thì chi phí cho thao tác này là , với là số cạnh.

Với mỗi cây T đang xây dựng, ta phân hoạch các đỉnh đã lựa chọn thành các nhóm sao cho các đỉnh trong cùng một nhóm thì liên thông. Nếu xây dựng phân hoạch này ngay từ đầu thì chi phí để tạo phân hoạch mới khi thêm một cạnh vào chỉ là hằng số. Và đồng thời phân hoạch này giúp ta có thể kiểm tra được điều kiện cạnh e đang xét có tạo thành chu trình với các cạnh đã chọn hay không cũng với chi phí là hằng số. Với thuật toán Kruskal thì ta quét lần lượt các cạnh từ 1 đến . Do đó chi phí để xây dựng cây T là

Tóm lại, chi phí chung của cả thuật toán là

***Bài 3:****Tự đặt một số đề bài toán, phân tích bài toán, xây dựng giải thuật, đánh giá độ phức tạp và viết chương trình để minh họa phương pháp tham lam (ít nhất 01 bài toán).*

Thuật toán chia các cụm điểm trong không gian (Clustering alogrithm)

* Phát biểu bài toán: Cho tập gồm đối tượng và các khoảng cách giữa hai đối tượng bất kì. Chia tập thành nhóm sao cho các đối tượng trong cùng một nhóm thì gần nhau và thuộc hai nhóm khác nhau thì xa nhau.
* Bài toán này gặp rất nhiều trong thực tế. Ví dụ chia hàng tỷ thiên thể thành các hệ, các thiên hà.Ta nhận thấy rằng các thiên thể trong cùng một thiên hà rất gần nhau, còn các thiên thể trong thiên hà khác nhau thì sẽ cách xa nhau.
* Khái niệm “khoảng cách” không chỉ là độ dài thông thưởng mà có thể mở rộng.Ví dụ so sánh độ giống nhau giữa 2 đối tượng. Hai đối tượng càng khác nhau ta nói “khoảng cách” của chúng càng lớn.
* Với khái niệm khoảng cách mở rộng thì bài toán chia cụm điểm có thêm nhiều ứng dụng mới, ví dụ:phân loại tài liệu trong tìm kiếm web, tìm kiếm ảnh trong kho dữ liệu ảnh, nhận dạng các đoạn mã gien,…
* Phát biểu tường minh bài toán
* Cho tập U gồm n đối tượng . Khoảng cách giữa các đối tượng được cho bởi ma trận , .Khoảng cách này có các tính chất sau:
* Thuật giải tham lam của bài toán:
* Thuật giải bao gồm 3 bước:
* *Bước 1:Mô hình hóa n đối tượng thành n đỉnh của đồ thị G có trọng số, trọng số cung nối 2 đỉnh là khoảng cách của 2 đối tượng biểu diễn bởi 2 đỉnh đó.*
* *Bước 2:Tìm cây bao trùm nhỏ nhất T của đồ thị G.*
* *Bước 3:Xóa đi cạnh có trọng số lớn nhất của T, ta được k thành phần liên thông này là nhóm cần tìm.*
* **Ví dụ:** có 5 đối tượng với khoảng cách d như sau:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| p | 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 1 | 2 | 2 | 2 |
| 2 | 1 | 0 | 2 | 2 | 2 |
| 3 | 2 | 2 | 0 | 1 | 1 |
| 4 | 2 | 2 | 1 | 0 | 1 |
| 5 | 2 | 2 | 1 | 1 | 0 |

Ta cần chia 5 đối tượng này thành 2 nhóm cụm điểm .

Ta giải quyết ví dụ trên thông qua ba bước

* *Bước 1:Mô hình hóa thành đồ thị có 5 đỉnh có trọng số*

1

1

1

2

2

2

1

2

2

2

* *Bước 2:Dùng thuật toán prim ta tìm được cây bao trùm nhỏ nhất, ta được cây T như sau: (Cây T có các cạnh được in đậm như hình vẽ )*

1

1

1

2

2

2

1

2

2

2

* *Bước 3:Xóa 1 cạnh có độ dài lớn nhất của T là ta được 2 thành phần liên thông*

1

1

1

Như vậy, từ 5 đối tương ban đầu có thể chia ra thành 2 nhóm và .

Tính tham lam của thuật toán thể hiện ở hai điểm:

+) thuật toán có sử dụng Prim để xây dựng cây khung nhỏ nhất, đây là thuật toán có tính tham lam

+) xóa đi cạnh có độ dài lớn nhất để thu được nhóm cách xa nhau.

Nếu định nghĩa tính tối ưu của thuật toán theo khoảng cách nhóm nhỏ nhất; khoảng cách nhóm của 2 nhóm là khoảng cách giữa 2 phần tử gần nhau nhất thuộc 2 nhóm đó.Thuật toán trên tuy là tham lam nhưng giống như thuật toán Prim, nó đảm bảo tính tối ưu này.

Dữ liệu vào được ghi vào file MatranKC.txt, trong đó:

+) hàng đầu tiên ghi số đối tượng và số nhóm cần chia ,

+) hàng tiếp theo mỗi hàng có số ghi lại ma trận khoảng cách.

Các dữ liệu chính có:

+) : số đối tượng,

+) số cạnh của đồ thị

+) là ma trận khoảng cách.

**Giả Mã**:

Int main()

InputData*(); //Đọc dữ liệu từ file MatranKC.txt*

StructureGraph*();//Dựng mô hình đồ thị từ n đối tượng và các khoảng cách giữa chúng*

QuickSort(weitEd,1,m*);//Sắp xếp các cạnh theo thứ tự tăng dần trọng số bằng thuật toán Quick sort*

PrintEd();*//In danh sách các cạnh đã sắp xếp để kiểm tra*

SpanTree();*//Tìm cây bao trùm nhỏ nhất bằng thuật toán Prim*

PrintTree();*//In cây bao trùm vừa tìm được*

QuickSortMST(newWeitEd,1,n-1*);//Sắp xếp các cạnh của cây bao trùm bằng thuật toán Quick sort*

DelEdgMST();*//Xóa đi k-1 cạnh lớn nhất của cây bao trùm vừa tìm được để nhận được thành phần liên thông tương ứng với nhóm cần chia.*

PrintClusters();*//In thành phần liên thông ra, tương ứng với các cạnh chưa bị xóa của cây bao trùm nhỏ nhất*.

* Độ phức tạp của thuật toán (ta chỉ đánh giá những thủ tục bắt buộc phải dung, các thủ tục như in cây để kiểm tra lại có thể bỏ qua):

InputData();Tốn thời gian nhất là đọc ma trận x , dó đó độ phức tạp của bước này bằng

StructureGraph(); thủ tục này cần xây dựng đồ thị có đỉnh và cạnh do đó độ phức tạp của nó bằng .

QuickSort(weitEd,1,m); thủ tục này cần sắp xếp một mảng có phần tử bằng thuật toán Quick sort, độ phức tạp của nó bằng

PrintEd();

SpanTree();thuật toán Prim để dựng cây khung có cạnh, trong trường hợp xấu nhất, mỗi lần thêm cạnh nó phải duyệt toàn bộ cạnh, do đó độ phức tạp của thủ tục này là

PrintTree();

QuickSortMST(newWeitEd,1,n-1); sắp xếp lại cạnh của cây bao trùm theo thứ tự tăng dần của trọng số, độ phức tạp bằng

DelEdgMST(); xóa cạnh có độ dài lớn nhất của cây bao trùm chi phí thời gian bằng .

PrintClusters();

Độ phức tạp của cả thuật toán