WIKIPEDIA

Thuật toán Kruskal

Bách khoa toàn thư mở Wikipedia

Thuật toán Kruskal là một thuật toán trong lý thuyết đồ thị để tìm cây bao trùm nhỏ nhất của một đồ thị liên thông có trọng số. Nói cách khác, nó tìm một tập hợp các cạnh tạo thành một cây chứa tất cả các đỉnh của đồ thị và có tổng trọng số các cạnh là nhỏ nhất.Thuật toán Kruskal là một ví dụ của thuật toán tham lam.

Thuật toán này xuất bản lần đầu tiên năm 1956, bởi Joseph Kruskal[1].

Một vài thuật toán khác cho bài toán này bao gồm thuật toán Prim, thuật toán xóa ngược, và thuật toán Borůvka.

Mục lục

Bài toán dẫn nhập

Cho một đồ thị có trọng số với n đỉnh. Yêu cầu tìm ra cây khung nhỏ nhất.

Tư tưởng thuật toán

Thuật toán Kruskal dựa trên mô hình xây dựng cây khung nhỏ nhất bằng thuật toán hợp nhất.

- Thuật toán không xét các cạnh với thứ tự tuỳ ý.
- Thuật toán xét các cạnh theo thứ tự đã sắp xếp theo trọng số.

Để xây dựng tập n-1 cạnh của cây khung nhỏ nhất - tạm gọi là tập K, Kruskal đề nghị cách kết nạp lần lượt các cạnh vào tập đó theo nguyên tắc như sau:

- Ưu tiên các cạnh có trọng số nhỏ hơn.
- Kết nạp cạnh khi nó không tạo chu trình với tập cạnh đã kết nạp trước đó.

Đó là một nguyên tă´c chính xác và đúng đă´n, đảm bảo tập K nê´u thu đủ n - 1 cạnh sẽ là cây khung nhỏ nhâ´t.

Mô tả thuật toán

Giả sử ta câ`n tìm cây bao trùm nhỏ nhâ´t của đô` thị *G*. Thuật toán bao gô`m các bước sau.

- Khởi tạo rừng *F* (tập hợp các cây), trong đó mỗi đỉnh của *G* tạo thành một cây riêng biệt
- Khởi tạo tập *S* chứa tất cả các cạnh của *G*
- Chừng nào *S* còn khác rỗng và *F* gồm hơn một cây
  - Xóa cạnh nhỏ nhất trong *S*
  - Nếu cạnh đó nối hai cây khác nhau trong *F*, thì thêm nó vào *F* và hợp hai cây kề với nó làm một
  - Nếu không thì loại bỏ cạnh đó.

Khi thuật toán kê´t thúc, rừng chỉ gô`m đúng một cây và đó là một cây bao trùm nhỏ nhâ´t của đô` thị *G*.

Mã giả

Cho đô` thị **G=(X, E)**.

```
Bước 1: Sắp xếp các cạnh của đồ thị theo thứ tự trọng số tăng dần.
Bước 2: Khởi tạo T:= ∅
Bước 3: Lần lượt lấy từng cạnh thuộc danh sách đã sắp xếp. Nếu T+{e} không chứa chu trình thì gán T:=T+{e}.
Bước 4: Nếu T đủ n-1 phần tử thì dừng, ngược lại làm tiếp bước 2.
```

Kỹ thuật đánh nhãn đỉnh

Kỹ thuật đánh nhãn đỉnh Trong thuật toán Kruskal, để kiểm tra xem T + {e} có chứa chu trình hay không ta có thể dùng kỹ thuật gă´n nhãn đỉnh, kỹ thuật này khá đơn giản và hiệu quả.

- Ngay sau **bước 1** của thuật toán, ta gắn đỉnh i của đồ thị một nhãn là i
- **Trong bước 2:**
  - Nếu hai đầu cạnh e có cùng nhãn (tức là nhãn của e.v1 và nhãn của e.v2 bằng nhau) thì T+{e} tạo chu trình, ta không đưa e vào T.
  - Ngược lại [nếu Label(e.v1)!= Label(e.v2)] thì ta đưa e vào T và thực hiện công việc ghép nhãn bằng cách:
    - lab1 = Min(Label(e.v1), Label (e.v2))
    - lab2 = Max(Label(e.v1), Label (e.v2))
    - Sửa nhãn của tất cả các đỉnh nào có nhãn là lab2 thành nhãn lab1

Ghi chú

- Trong quá trình xây dựng T thì các cạnh có thể không liên thông nhau lúc đó T chỉ là rừng chứ chưa trở thành cây.
- Khi thuật toán dừng:
  - Nếu T chưa đủ n - 1 cạnh thì đồ thị G không liên thông(không có cây khung)
  - Ngược lại thì T là cây khung cần tìm.

Thời gian thực hiện

- Nếu *E* là số cạnh và *V* là số đỉnh của đồ thị thì thuật toán Kruskal chạy trong thời gian *O*(*E* log *V*).
- Có thể đạt được thời gian này bằng phương pháp sau: sắp xếp tất cả các cạnh theo trọng số trong thời gian *O*(*E* log *E*). Điều này cho phép thực hiện bước "xóa cạnh nhỏ nhất trong *S*" trong thời gian hằng số. Sau đó sử dụng cấu trúc dữ liệu cho các tập hợp không giao nhau để lưu trữ thông tin đỉnh nào nằm ở cây nào trong *F*. Ta cần thực hiện O(*E*) thao tác, hai thao tác 'tìm' và không quá một thao tác 'hợp' cho mỗi cạnh. Ngay cả những thuật toán đơn giản cho bài toán này, chẳng hạn hợp bằng trọng số cũng có thể thực hiện O(*E*) thao tác trong thời gian *O*(*E* log *V*). Vì vậy tổng thời gian là *O*(*E* log *E*) = *O*(*E* log *V*).

Chứng minh tính đúng đắn

Chứng minh gồm hai phần: chứng minh kết quả thuật toán là một cây bao trùm và cây bao trùm đó là nhỏ nhất.

Cây bao trùm

F luôn là một rừng do việc nối hai cây bằng một cạnh luôn tạo ra một cây mới. Giả thiết phản chứng *F* gồm ít nhất hai cây *A* và *B*. Khi cạnh đầu tiên nối các đỉnh trong *A* của *F* với phần còn lại của đồ thị được xem xét (cạnh này tồn tại do *G* liên thông) thì rõ ràng thuật toán sẽ chọn nó. Vì vậy *A* không thể là một cây trong *F* khi thuật toán kết thúc. Do đó, *F* liên thông và là một cây bao trùm.

Nhỏ nhất

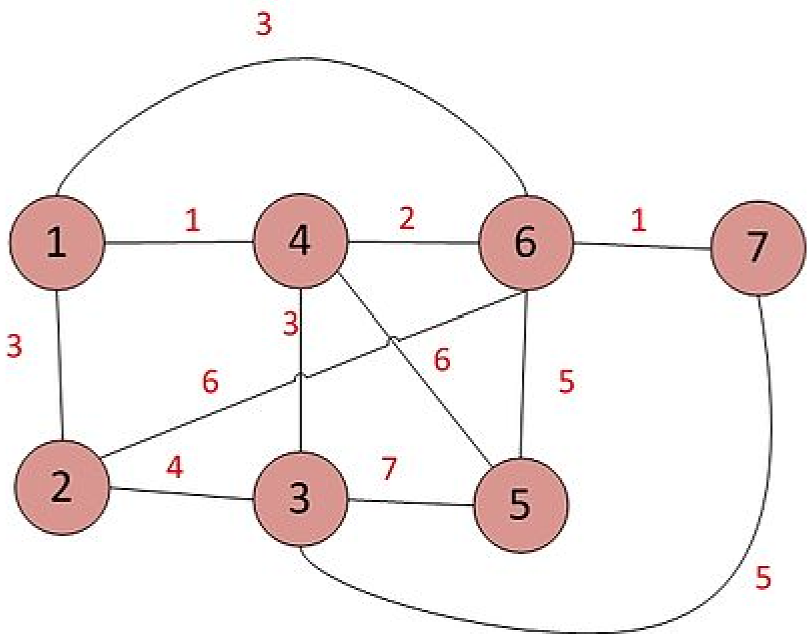
Ta chứng minh mệnh đề ***P*** sau đây bằng quy nạp: Nếu *F* là tập hợp các cạnh đã chọn tại bất kì thời điểm nào trong quá trình thực thi thuật toán thì tồn tại cây bao trùm nhỏ nhất chứa *F*.

- Rõ ràng ***P*** đúng khi thuật toán bắt đầu vì *F* là rỗng.
- Giả sử ***P*** là đúng cho một tập hợp *F* và giả sử *T* là một cây bao trùm nhỏ nhất chứa *F*. Nếu cạnh được thêm vào tiếp theo là *e* cũng nằm trong *T*, thì ***P*** đúng cho *F* + *e*. Nếu không, thì *T* + *e* chứa chu trình *C* và tồn tại cạnh *f* nằm trên *C* nhưng không trong *F*. (Nếu không có cạnh *f*, thì không thể thêm *e* vào *F*, do sẽ tạo ra chu trình *C* trong *F*.) Do đó *T* − *f* + *e* là một cây, và nó có cùng trọng số với *T*, do *T* có trọng số nhỏ nhất và *f* không thể nhỏ hơn *e*, vì nếu không thuật toán đã xem xét *f* trước *e* và chọn *f*. Vì vậy *T* − *f* + *e* là một cây bao trùm nhỏ nhất chứa *F* + *e* và ***P*** là đúng.
- Như vậy, ***P*** đúng khi thuật toán kết thúc và *F* là một cây bao trùm. Điều này chỉ có thể xảy ra nếu *F* là một cây bao trùm nhỏ nhất.

Ví dụ

- **Cho đồ thị G như hình vẽ:**. Yêu cầu tìm ra cây khung nhỏ nhất của đồ thị G.
- **G gồm có 7 đỉnh**
- Đồ thị G có n phần tử. Thuật toán Kruskal sẽ dừng khi có n-1 trong tập hợp T
  - n = 7
  - Vậy số cạnh trong tập hợp T: n - 1 = 7 - 1 = 6(*)

Bước 1: Liệt kê tất cả cạnh với trọng số của cạnh đó: Dựa vào đồ thị ta liệt kê ra các cạnh gồm đỉnh đầu, đỉnh cuối và trọng số:



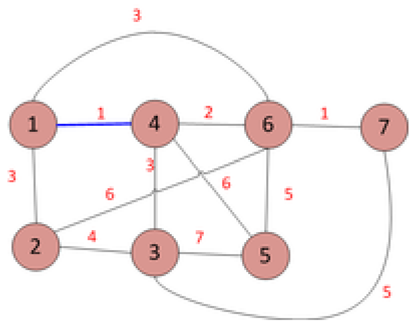
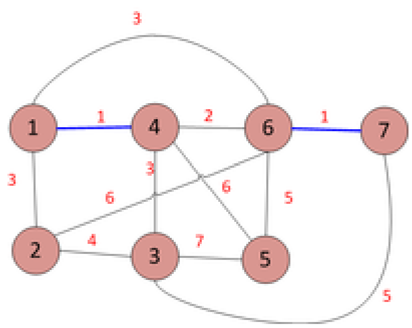
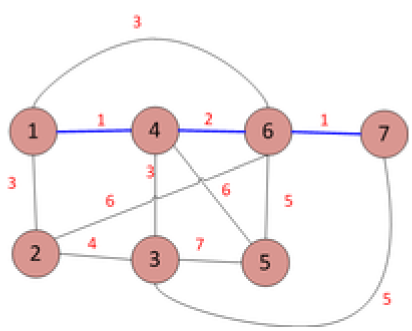
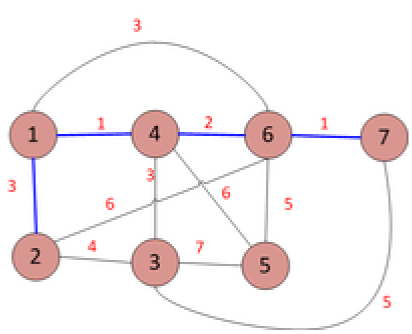
Đồ thị G

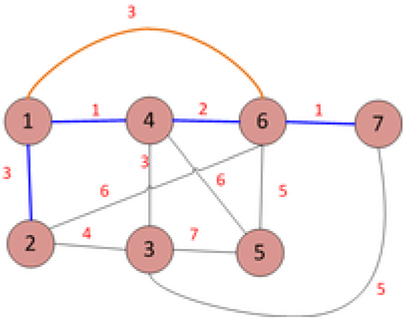
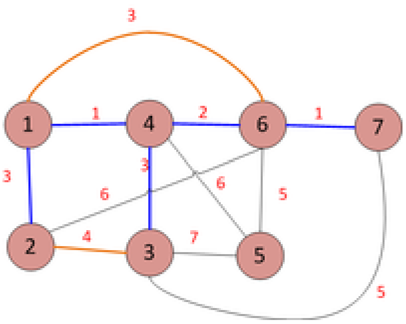
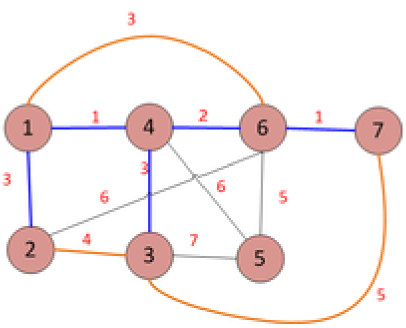
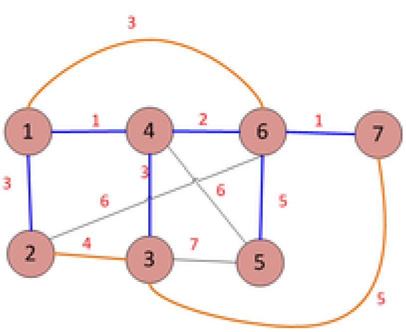
Điểm đầu	Điểm cuối	Trọng số
1	2	3
1	4	1
1	6	3
2	3	4
2	6	6
3	4	3
3	5	7
3	7	5
4	5	6
4	6	2
5	6	5
6	7	1

Bước 2: Sắp xếp các cạnh theo trọng số tăng dần:

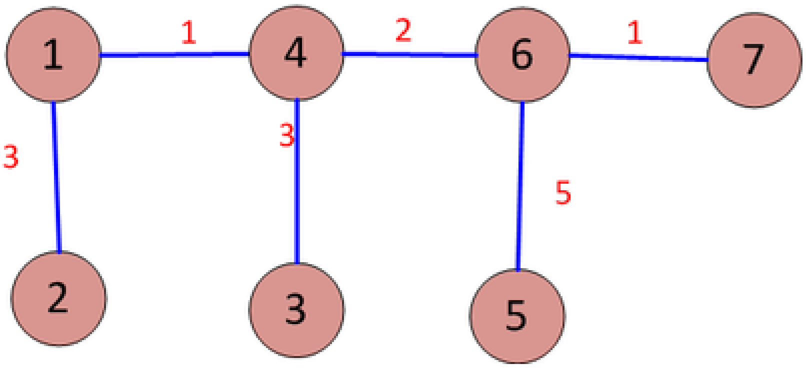
Điểm đầu	Điểm cuối	Trọng số
1	4	1
6	7	1
4	6	2
1	2	3
1	6	3
3	4	3
2	3	4
3	7	5
5	6	5
2	6	6
4	5	6
3	5	7

Bước 3: Dựa vào kết quả ở bước 2. Ta tiến hành tìm cây khung bằng thuật toán Kruskal

Kết quả	Cạnh đang xét
 Đồ thị G	1-4-1: Ta nhận thấy cạnh 1-4 không tạo ra một chu trình nào. Vì vậy, thêm 1-4 vào tập hợp
 Đồ thị G	6-7-1: Ta nhận thấy cạnh 6-7 không tạo ra một chu trình nào. Vì vậy, thêm 6-7 vào tập hợp
 Đồ thị G	4-6-2: Ta nhận thấy cạnh 4-6 không tạo ra một chu trình nào. Vì vậy, thêm 4-6 vào tập hợp
 Đồ thị G	1-2-3: Ta nhận thấy cạnh 1-2 không tạo ra một chu trình nào. Vì vậy, thêm 1-2 vào tập hợp
	1-6-3: Ta nhận thấy cạnh 1-6 tạo ra một chu trình. Không thêm vào tập hợp. 3-4-3: Ta nhận thấy cạnh 3-4 không tạo ra một chu trình. Vì vậy, thêm 3-4 vào tập hợp

 Đồ thị G	
 Đồ thị G	2-3-4: Ta nhận thấy cạnh 2-3 tạo ra một chu trình. Không thêm vào tập hợp.
 Đồ thị G	3-7-5: Ta nhận thấy cạnh 3-7 tạo ra một chu trình. Không thêm vào tập hợp.
 Đồ thị G	5-6-5: Ta nhận thấy cạnh 5-6 không tạo ra một chu trình nào. Vì vậy, thêm 5-6 vào tập hợp

- Đến đây, ta đã tìm được 6 cạnh. Vậy kết thúc thuật toán. (Thỏa (*))
- **Kết quả: Ta được đồ thị sau**



Đồ thị G

Với tổng chi phí là: Ta cộng tất cả các trọng số giữa các đỉnh lại với nhau

- Vậy tổng chi phí: 3 + 1 + 3 + 2 + 5 + 1 = 15

Xem thêm

- Cây bao trùm nhỏ nhất

Ghi chú

1. ^ Kruskal (1956)

Tham khảo

- Kruskal, Joseph. B. (tháng 2 1956), "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem" (http://links.jstor.org/sici?sici=0002-9939(195602)7%3A1%3C48%3AOTSSSO%3E2.0.CO%3B2-M), *Proceedings of the American Mathematical Society* **7** (1): 48–50 Kiểm tra giá trị ngày tháng trong: `|date=` (trợ giúp)
- Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). *Introduction to Algorithms* (ấn bản 2). MIT Press và McGraw-Hill. ISBN 0-262-03293-7. Phần 23.2: Các thuật toán Kruskal và Prim, tr. 567–574.
- Goodrich, Michael T.; Tamassia, Roberto (2006). *Data Structures and Algorithms in Java* (ấn bản 4). John Wiley & Sons, Inc. ISBN 0-471-73884-0. Phần 13.7.1: Thuật toán Kruskal, tr. 632.

Liên kết ngoài

- Minh họa thuật toán Kruskal bằng Java (http://students.ceid.upatras.gr/~papagel/project/kruskal.htm)
- Lập trình thuật toán Kruskal bằng ngôn ngữ C# (http://www.codeproject.com/KB/recipes/Kruskal_Algorithm.aspx)

Lấy từ "https://vi.wikipedia.org/w/index.php?title=Thuật_toán_Kruskal&oldid=25949562"