**Thuật toán bài Net:**

Nhận xét rằng các đường truyền quan trọng là các cạnh cầu của đồ thị vô hướng. Xét (𝑢,) là một cạnh cầu như vậy với 𝑙𝑜𝑤[𝑣] > 𝑛𝑢𝑚[𝑢],𝑝𝑟𝑒𝑣[𝑣] = 𝑢. Khi đó việc bỏ đi cạnh (𝑢,) sẽ tách đồ thị thành hai phần chính: Phần thứ nhất là các đỉnh thuộc cây cobn gốc 𝑣 trên cây DFS và phần thứ hai là tập các đỉnh còn lại. Dễ thấy điều kiện để (𝑢,v) là một đường truyền quan trọng là trong mỗi phần đều phải có các đỉnh thuộc cả loại A và loại B.

Đặt 𝑓[𝑢],𝑓𝐵[𝑢] là số lượng đỉnh thuộc loại A (𝑎[𝑣] = 1) và số lượng đỉnh thuộc loại B (𝑏[𝑣] = 1) ta có công thức qui hoạch động như sau:

𝑓𝐴[𝑢] = 𝑎[𝑢] + ∑ 𝑓𝐴[𝑣] 𝑣∈𝑘𝑒(𝑢),𝑝𝑟𝑒𝑣[𝑣]=𝑢

𝑓𝐵[𝑢] = 𝑏[𝑢] + ∑ 𝑓𝐵[𝑣] 𝑣∈𝑘𝑒(𝑢),𝑝𝑟𝑒𝑣[𝑣]=𝑢

Thuật toán có thể mô tả qua các bước:

B1: DFS(1) để tính các mảng **num[u],** **low[u],** **tp[u]**

B2: Qui hoạch động ngược trên mảng tp để tính các mảng **fA[...],** **fB[...]**

B3: Với (𝑢,) là cạnh cầu: 𝑝𝑟𝑒𝑣[𝑣] = 𝑢. Nếu 𝑓[𝑣] = 𝐾 hoặc 𝑓𝐴[𝑣] = 0 hoặc 𝑓𝐵[𝑣] = 𝐿 hoặc 𝑓𝐵[𝑣] = 0 thì (𝑢,𝑣) là một đường truyền quan trọng.

**Thuật toán bài BLO**

Trước tiên giả thiết rằng một đồ thị có 𝑚 thành phần liên thông với số lượng đỉnh mỗi thành phần là 𝑎1,2,…,𝑎𝑚 thì số cuộc thăm viếng không thực hiện được sẽ là:

𝑎1(𝑛 − 𝑎1) + 𝑎2(𝑛 − 𝑎2) + ⋯𝑎𝑚(𝑛 − 𝑎𝑚) (\*)

Đặt [𝑢] là số đỉnh thuộc cây con gốc 𝑢 trong cây DFS ta có công thức qui hoạch động:

[𝑢] = 1 + ∑ 𝑓[𝑣]

𝑝𝑟𝑒𝑣[𝑣]=𝑢

+) Nếu 𝑢 là không phải là đỉnh khớp thì khi tách 𝑢 và các cạnh liên thuộc với nó ra khỏi đồ thị. Đồ thị mới có 2 thành phần liên thông với 𝑎1 = 1,2 = 𝑛 − 1. Do vậy số cuộc thăm viếng không thực hiện được (tính theo (\*)) là:

1 ∙ (𝑛 − 1) + (𝑛 − 1) ∙ 1 = 2 ∙ (𝑛 − 1)

+) Nếu 𝑢 là đỉnh khớp và 𝑝𝑟𝑒[𝑢] = 0 (𝑢 là đỉnh xuất phát). Gọi 𝑣1,2,…,𝑣𝑘 là các con của 𝑢 (𝑝𝑟𝑒𝑣[𝑣𝑖] = 𝑢) ta thấy được khi bỏ 𝑢 và các cạnh liên thuộc đồ thị sẽ có 𝑘 + 1 thành phần liên thông với số đỉnh lần lượt là:

𝑎1 = [𝑣1],𝑎2 = 𝑓[𝑣2],…,𝑎𝑘 = 𝑓[𝑣𝑘],𝑎𝑘+1 = 1 và áp dụng trực tiếp (\*) ta có được kết quả.

+) Nếu 𝑢 là đỉnh khớp và 𝑝𝑟𝑒[𝑢] = 𝑤 ≠ 0. Gọi 𝑣1,2,…,𝑣𝑘 là các con tạo nhánh của 𝑢 (𝑝𝑟𝑒𝑣[𝑣𝑖] = 𝑢,𝑙𝑜𝑤[𝑣𝑖] ≥ 𝑢). Dễ thấy đồ thị được tách thành 𝑘 + 2 thành phần liên thông với:

𝑎1 = [𝑣1],𝑎2 = 𝑓[𝑣2],…𝑎𝑘 = 𝑓[𝑣𝑘],𝑎𝑘+1 = 1,𝑎𝑘+2 = 𝑛 − ∑ 𝑎𝑖 𝑖≤𝑘+1

Áp dụng (\*) ta có được kết quả cần tìm. Thuật toán chia thành các bước:

B1: DFS để tính mảng **num[...],** **low[...],** **tp[...]** B2: Qui hoạch động ngược để tính mảng **f[...]**

B3: Lặp qua tất cả các đỉnh. Với mỗi đỉnh tùy theo nó thuộc loại nào trong số ba loại đỉnh nói trên ta áp dụng công thức tương ứng để tính.

**Thuật toán bài BLUEHOUSE**

+) Trước tiên chúng ta xét trường hợp khi đồ thị xây dựng là một cây vô hướng. Trên cây vô hướng này ta phân biệt hai loại đỉnh:

 Đỉnh trong: Là các đỉnh có bậc lớn hơn 1  Đỉnh lá: Là các đỉnh có bậc bằng 1.

Ta sẽ chỉ ra rằng nếu như số đỉnh lá là 𝑚 thì số cạnh nối sẽ là [𝑚+1]:

2

Gọi 𝑡[1],𝑡𝑝[2],…,𝑡𝑝[𝑛] là thứ tự topo DFS của cây bắt đầu từ một đỉnh nào đó. Trên topo này ta gọi các đỉnh lá theo thứ tự là 𝑥1,2,…,𝑥𝑚. Thực hiện thêm các cạnh giữa các lá theo qui tắc sau: Nối (𝑥1,3),(𝑥2,𝑥5),(𝑥4,𝑥7),(𝑥6,𝑥9),… (Nếu có dãy ba đỉnh lá liên tiếp 𝑢,,𝑤 thì nối (𝑢,𝑤) còn lại 1 đỉnh lá 𝑣). Sau quá trình trên nếu số lá còn lại chưa nối là 2 thì nối nốt chúng, nếu số lá còn lại là 1 thì nối nó với đỉnh gốc. Thủ tục sau mô tả cách nối:

**sn:=0;**

**for** **i:=1** **to** **n** **do** **begin**

**u:=tp[i];**

**if** **deg[u]=1** **then** **begin**

**inc(sn);** **s[sn]=u;** **if** **sn>=3** **then** **begin**

**u:=s[sn-2],** **v=:=s[sn-1],** **w:=s[sn];** **Thêm** **cạnh** **(u,w)** **cần** **nối**

**dec(sn,2);** **s[sn]:=v;** **end;**

**end;**

**if** **sn=2** **then** ***Thêm*** ***cạnh*** ***(s[1],*** ***s[2])*** **else** **if** **sn=1** **then** ***Thêm*** ***cạnh*** ***(s[1],*** ***root)*;**

Có thể chứng minh cách nối như trên đảm bảo rằng đồ thị mới sẽ không có cầu và số cạnh cần nối luôn là [𝑚+1] và con số này là nhỏ nhất có thể.

2

+) Nếu đồ thị không phải là cây. Xây dựng đồ thị mới với mỗi đỉnh là một thành phần song liên thông ta có được một cây. Bài toán qui về việc nối các thành phần song liên thông để bỏ đi các cầu trên đồ thị mới.

Nhận xét rằng số hiệu các thành phần song liên thông tìm được trong thuật toán Tarjan sẽ là một thứ tự topo DFS ngược trên đồ thị mới nên không cần thực hiện DFS trên đồ thị mới này. Thuật toán có thể mô tả như sau:

B1: Sử dụng thuật toán Tarjan tìm các thành phần song liên thông, mỗi thành phần song liên thông có một đỉnh đại diện cho nó.

B2: Tính bậc của mỗi thành phần song liên thông trên đồ thị mới (bằng cách duyệt qua các cạnh của đồ thị ban đầu)

B3: Lặp các thành phần song liên thông từ số hiệu cao đến thấp nếu bậc bằng 1 thì lấy ra theo thứ tự. Sử dụng đoạn code giả lệnh được trình bày ở trên.