



Các phương pháp duyệt đồ thị

Các phương pháp duyệt đồ thị

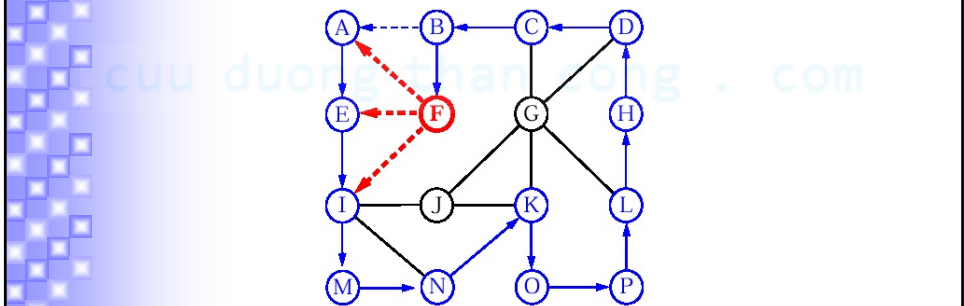
- Duyệt theo chiều sâu (Depth-First Search)
- Duyệt theo chiều rộng (Breadth-First Search)

The diagram shows a directed graph with 16 nodes labeled A through P. The nodes are arranged in a grid-like structure. Node F is highlighted in red. Red dashed arrows indicate a path from B to A, C to B, D to C, E to F, and I to J. Blue solid arrows indicate other connections between nodes.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

2

- Duyệt theo chiều sâu (Depth-First Search)
- Duyệt theo chiều rộng (Breadth-First Search)



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật 2

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật 2



Depth-First Search (DFS)

cuu duong than cong . com



Khái niệm

- **DFS** trên một đồ thị vô hướng cũng giống như khám phá một mê cung với một cuộn chỉ và một thùng sơn đỏ để đánh dấu, tránh bị lạc.
- Ta bắt đầu từ đỉnh **s**, buộc đầu cuộn chỉ vào **s** và đánh dấu đỉnh này là “**đã thăm**”. Sau đó ta đánh dấu s là đỉnh hiện hành **u**.

Khái niệm

- Bây giờ, ta đi theo cạnh **(u, v)** bất kỳ.
- Nếu cạnh **(u, v)** dẫn chúng ta đến đỉnh “**đã thăm**” **v**, ta quay trở về **u**.
- Nếu đỉnh **v** là đỉnh mới, ta di chuyển đến **v** và không quên lăm cuộn chỉ của mình theo. Đánh dấu đỉnh **v** là “**đã thăm**”. Đặt **v** thành đỉnh hiện hành và lặp lại các bước trước.

Khái niệm

- Cuối cùng, ta có thể đi đến một điểm mà tại đó tất cả các cạnh kề với nó đều dẫn chúng ta đến các đỉnh “**đã thăm**”. Khi đó, ta sẽ quay lui bằng cách cuộn ngược cuộn chỉ và quay lại cho đến khi trở lại một đỉnh kề với một cạnh còn chưa được khám phá. Lại tiếp tục qui trình khám phá như trên.
- Khi chúng ta trở về **s** và không còn cạnh nào kề với nó chưa bị khám phá là lúc **DFS** dừng.

Thuật toán Depth-First Search

Algorithm DFS(v);

Input : Một đỉnh v của đồ thị

Output : Một cách gài nhãn cho các cạnh để
“**nhớ lại**” hoặc “**backedge**”

for (mọi cạnh e kề với v) **do**

if cạnh e chưa **nhớ lại** **then**

 Gọi w là đỉnh kề của e

if đỉnh w là đỉnh mới **then**

 Gài nhãn e là “**nhớ lại**”

 Gọi lại DFS(w)

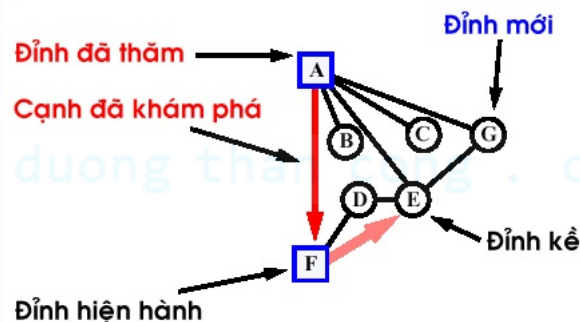
else

 Gài nhãn e là “**backedge**”

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

7

Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

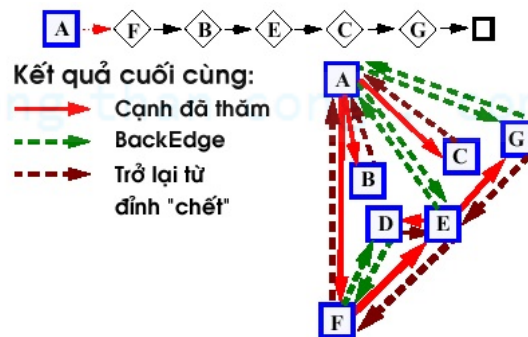
8

Xác định đỉnh kề trong DFS

- Kết quả của DFS phụ thuộc vào cách ta chọn đỉnh kế tiếp.

Xác định đỉnh kề trong DFS

- Nếu ta bắt đầu tại A và thử cạnh nối đến F, sau đó đến B, rồi đến E, C, cuối cùng là G ta được:



Xác định đỉnh kề trong DFS

- Nếu cũng bắt đầu từ A nhưng đi theo trình tự:



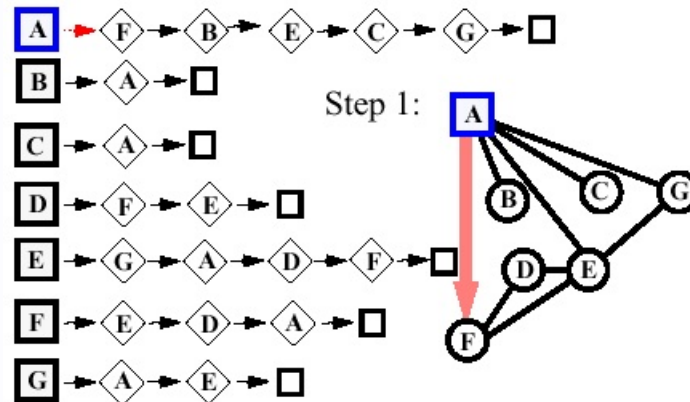
tập các cạnh đã thăm, backedge và các điểm đệ qui sẽ khác trước. (Hãy tự làm và kiểm chứng).

Thuật toán Depth-First Search

- Bây giờ ta sẽ xét từng bước của DFS qua ví dụ trên:



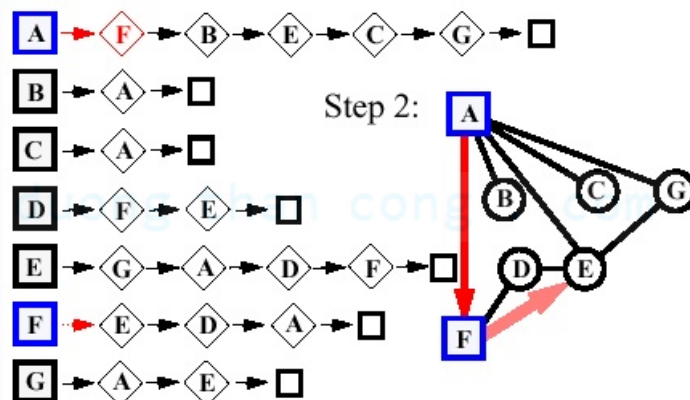
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

13

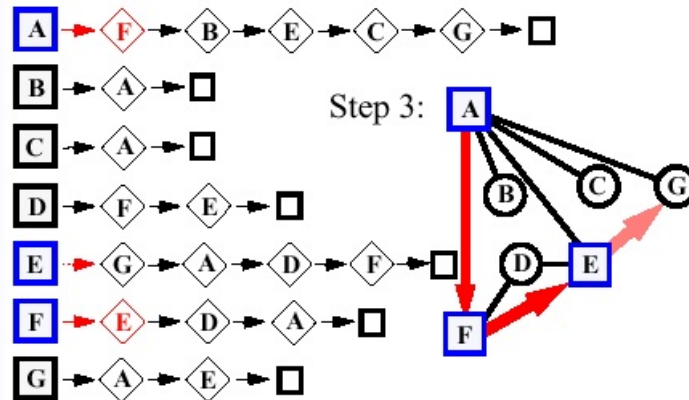
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

14

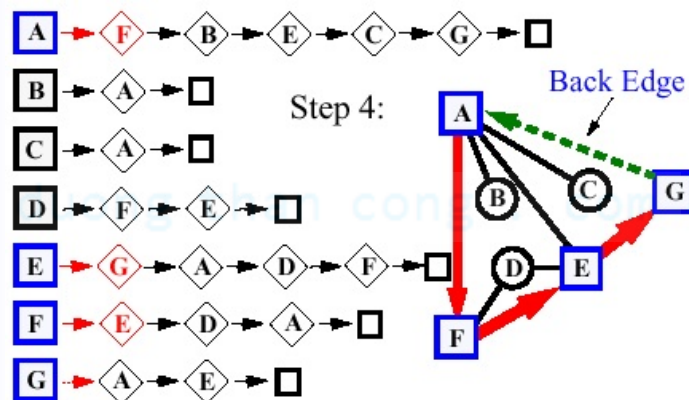
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

15

Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

16

Thuật toán Depth-First Search

The diagram illustrates the Depth-First Search (DFS) algorithm. It shows a search tree and a sequence of steps.

Search Tree:

- A (Root) has children B, C, D, E, F, G.
- B has child A.
- C has child A.
- D has children F, E.
- E has children G, A, D, F.

Sequence of Steps:

- A → F → B → E → C → G → □
- B → A → □
- C → A → □
- D → F → E → □
- E → G → A → D → F → □
- F → E → D → A → □
- G → A → E → □

Step 5: The diagram shows the search tree with nodes A, B, C, D, E, F, G. The path taken by the algorithm is highlighted in red: A → B → C → D → E → F → G. The current node is A, and the next node to be visited is G. The path from A to G is shown with a dashed green line.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

17

Thuật toán Depth-First Search

Step 6:

The diagram illustrates the Depth-First Search algorithm. It shows a sequence of steps from Step 1 to Step 6. Each step displays a search tree with nodes A, B, C, D, E, F, and G. Nodes are represented by squares (explored) and diamonds (in the current path). Red arrows indicate the path taken. Step 6 shows the search reaching node F, with a red arrow pointing to it from node A. A green dashed line indicates the path from A to F.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

18

Thuật toán Depth-First Search

Step 7:

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

19

Thuật toán Depth-First Search

Step 8:

The diagram illustrates the Depth-First Search (DFS) algorithm. It shows a sequence of nodes and edges, with some nodes highlighted in blue and others in black. The search path is indicated by red arrows, and the current node being explored is highlighted in blue. The diagram is labeled "Step 8:".

The nodes and edges are as follows:

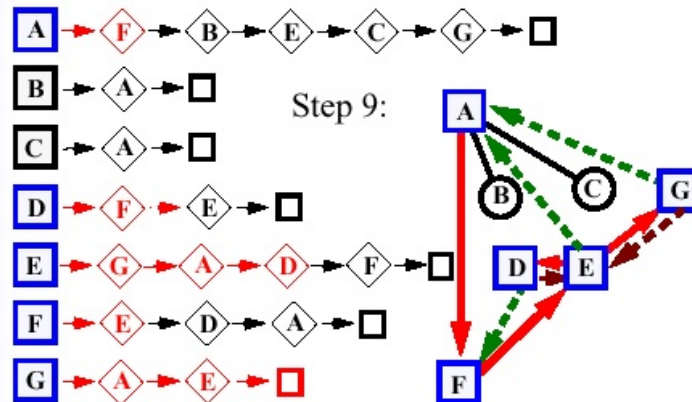
- Node A (blue) has edges to F (red), B (black), and C (black).
- Node B (black) has an edge to A (black).
- Node C (black) has an edge to A (black).
- Node D (blue) has edges to F (red) and E (black).
- Node E (black) has edges to G (red), A (red), and D (red).
- Node F (red) has edges to E (black) and D (red).
- Node G (red) has an edge to A (red).

The search path is highlighted in red, and the current node being explored is highlighted in blue. The search path starts at node A, goes to F, then B, E, C, G, and finally to a black square. The current node being explored is A, which has edges to B, C, and F. The search path is highlighted in red, and the current node is highlighted in blue.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

20

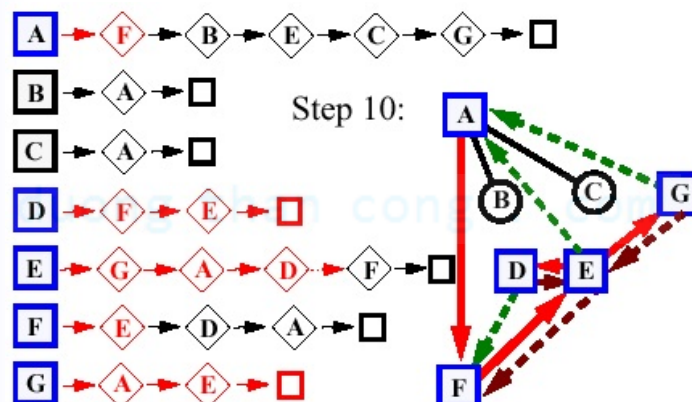
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

21

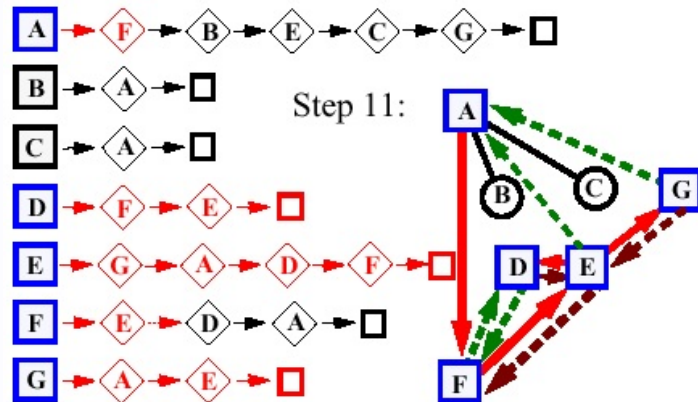
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

22

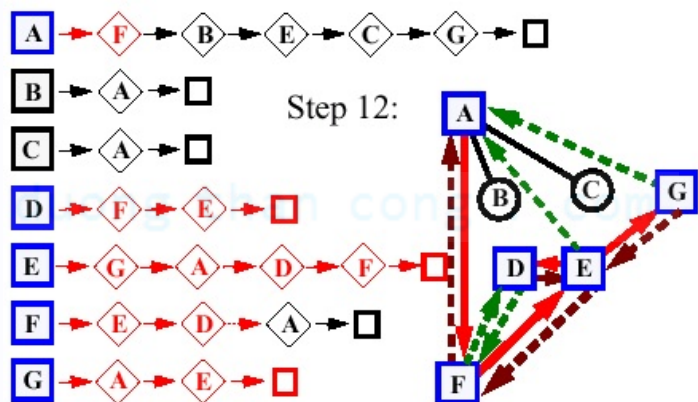
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

23

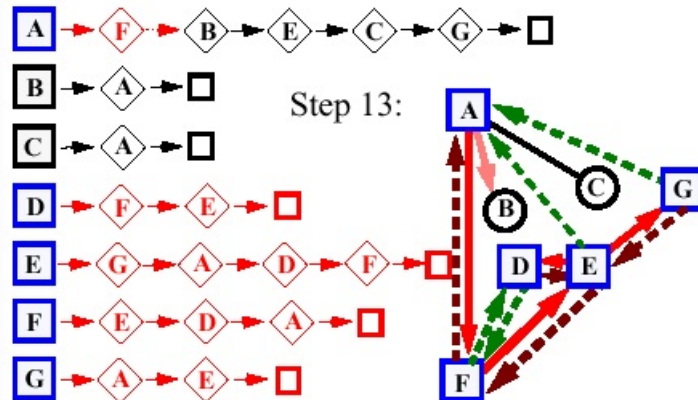
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

24

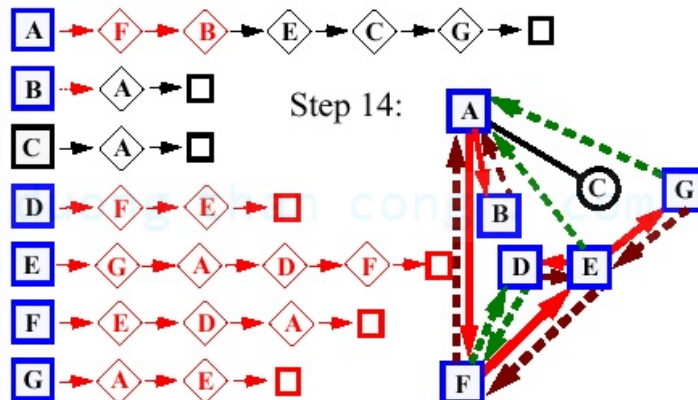
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

25

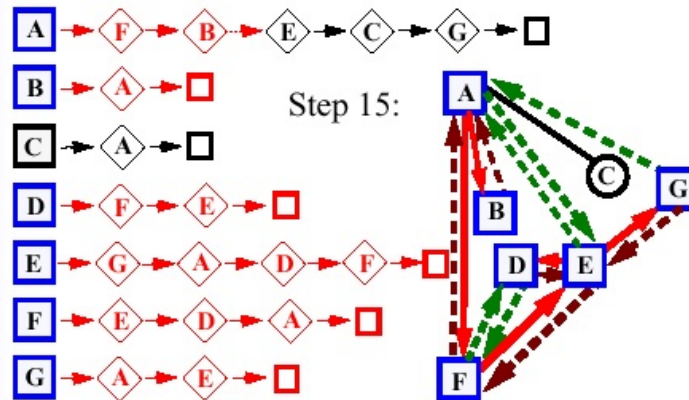
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

26

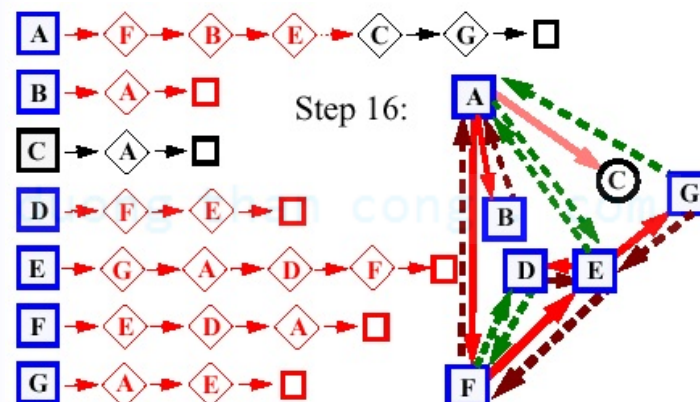
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

27

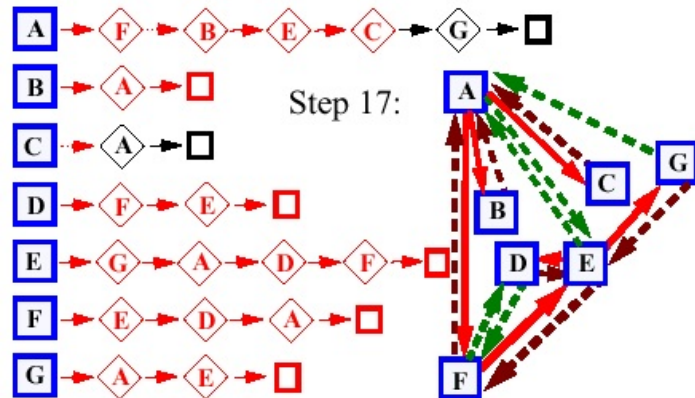
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

28

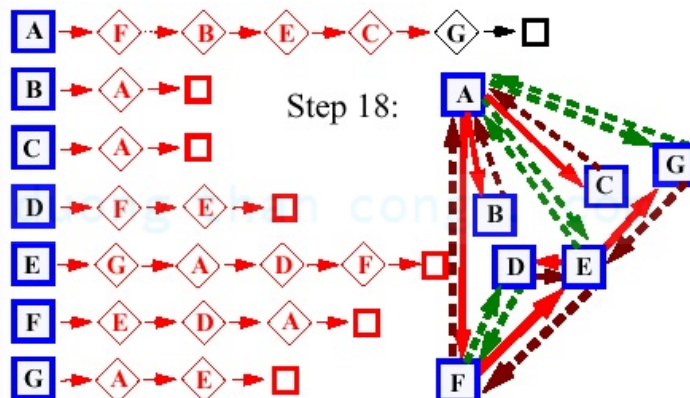
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

29

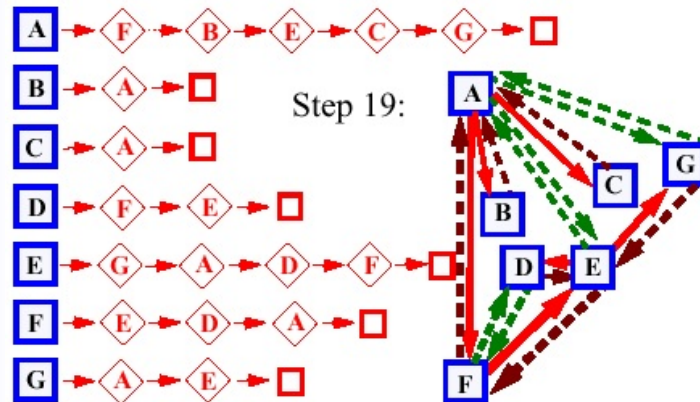
Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

30

Thuật toán Depth-First Search



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

31

Thuật toán Depth-First Search

- **Mệnh đề:** Gọi **G** là một đồ thị vô hướng, trên đó ta sẽ thực hiện thao tác DFS với đỉnh bắt đầu là **s** thì:
 1. Phép duyệt sẽ thăm tất cả các đỉnh cùng thành phần liên thông với **s**.
 2. Các cạnh có nhãn “**đã thăm**” sẽ tạo ra một cây tối đại của thành phần liên thông chứa **s**.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

32

Thuật toán Depth-First Search

■ Chứng minh:

- Khẳng định 1. là hiển nhiên vì DSF duyệt qua tất cả các đỉnh kề với đỉnh hiện hành. (Có thể chứng minh hoàn chỉnh hơn bằng phản chứng).
- Khẳng định 2. đúng do ta chỉ đánh dấu các cạnh đi đến một đỉnh mới nên không thể tạo ra chu trình. Như vậy DFS tạo ra một cây. Hơn nữa, DFS thăm tất cả các đỉnh thuộc thành phần liên thông nên cây này là cây tối đại.

Độ phức tạp thuật toán

■ Hãy nhớ rằng:

- **DFS** được gọi đúng 1 lần ứng với mỗi đỉnh.
- Mỗi cạnh được xem xét đúng 2 lần, mỗi lần từ một đỉnh kề với nó

Độ phức tạp thuật toán

- Với n_s đỉnh và m_s cạnh thuộc thành phần liên thông chứa s , một phép **DFS** bắt đầu tại s sẽ chạy với thời gian $O(n_s + m_s)$ nếu:
 - Đồ thị được biểu diễn bằng CTDL dạng danh sách kề.
 - Đặt nhãn cho một đỉnh là “**đã thăm**” và kiểm tra xem một đỉnh “**đã thăm**” chưa tốn chi phí $O(\text{degree})$.
 - Bằng cách đặt nhãn cho các đỉnh là “**đã thăm**”, ta có thể xem xét một cách hệ thống các cạnh kề với đỉnh hiện hành nên ta sẽ không xem xét một cạnh quá 1 lần.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

35

cuu duong than cong . com

Breadth-First Search (BFS)

duong than cong . com

Khái niệm

- Cũng giống như DFS, **Breadth-First Search (BFS)** duyệt qua toàn bộ các đỉnh thuộc một thành phần liên thông của đồ thị và xác định được một cây tối đại của nó với một số thuộc tính hữu ích:

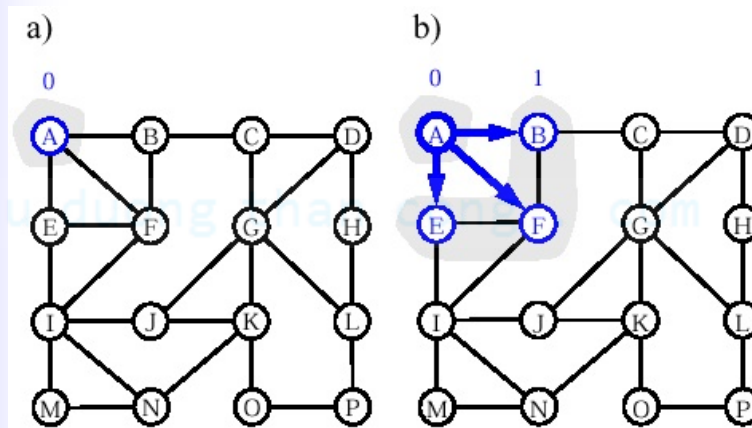
Khái niệm

- Đỉnh xuất phát **s** ở mức 0, và cũng như trong **DFS**, được xem như một điểm mốc trong quá trình tìm kiếm.
- Ở lượt thứ nhất, cuộn chỉ được mở ra dọc theo chiều dài một cạnh, và tất cả các đỉnh kề với điểm mốc (cách điểm mốc đúng một cạnh) đều được thăm.
- Các đỉnh này được đặt ở mức 1 (các cạnh tương ứng cũng vậy)

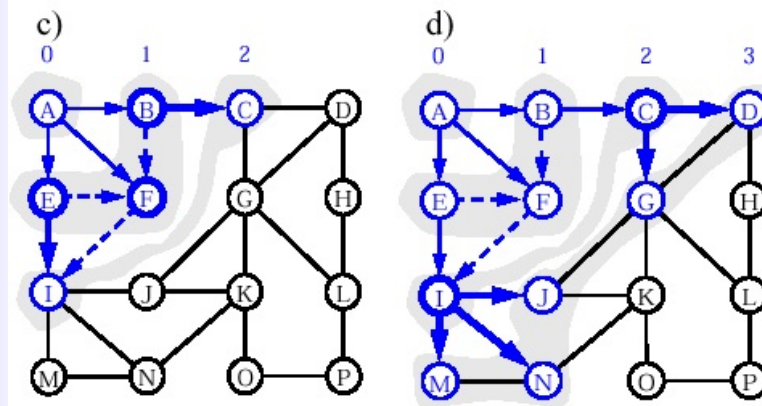
Khái niệm

- Ở lượt thứ hai, tất cả các đỉnh mới cách mức 2 cạnh sẽ được thăm và được đặt ở mức 2
- Quy trình này tiếp tục cho đến khi tất cả các đỉnh được thăm (được gán vào một mức nào đó).
- Nhãn của mọi đỉnh **v** tương ứng với đường đi qua ít cạnh nhất (ngắn nhất) từ **s** đến **v**.

Breadth-First Search (BFS)



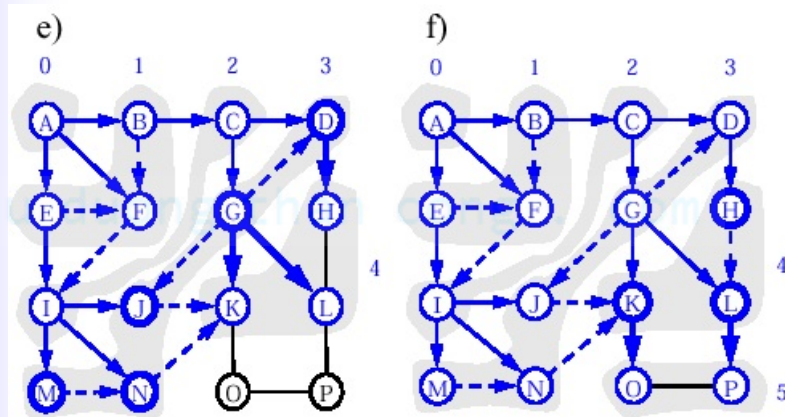
Breadth-First Search (BFS)



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

41

Breadth-First Search (BFS)



Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

42

Thuật toán Breadth-First Search

Algorithm BFS(v);

Input : Một đỉnh v của đồ thị

Output : Một cách gán nhãn cho các cạnh của đồ thị để tìm kiếm “**nỗ lực khám phá**” hoặc “**crossedge**”

Khởi tạo hàng đợi L_0 để chứa đỉnh s

$i \leftarrow 0$;

while $L_i \neq \emptyset$ **do**

 Tạo $L_{i+1} = \emptyset$

for mỗi đỉnh $v \in L_i$ **do**

if cạnh e kề với v **then**

 Gọi w là đỉnh khác của e

if đỉnh w là đỉnh mới **then**

 Gán nhãn e là “**được khám phá**”

 Chèn w vào L_{i+1}

else

 Gán nhãn e là “**crossedge**”

$i \leftarrow i + 1$

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

43

Các tính chất của BFS

- **Mệnh đề:** Gọi G là một đồ thị vô hướng trên đó thực hiện phép duyệt **BFS** bắt đầu từ đỉnh s . Ta có:
 - Phép duyệt sẽ thăm tất cả các đỉnh cùng thành phần liên thông với s .
 - Các cạnh có nhãn “**đã thăm**” sẽ tạo ra một cây tối đại của thành phần liên thông chứa s mà ta sẽ gọi là cây BFS.

Dương Anh Đức – Nhập môn Cấu trúc Dữ liệu và Giải thuật

44

Các tính chất của BFS

- Với mỗi cạnh v tại mức i , đường đi trên cây **BFS** giữa s và v qua i cạnh, và bất kỳ đường đi nào khác trên G giữa s và v dài tối thiểu i cạnh.
- Nếu (u, v) là một cạnh không nằm trên cây **BFS**, thì mức của u và v sai lệch không quá 1.

Các tính chất của BFS

- **Mệnh đề:** Gọi G là một đồ thị vô hướng với n đỉnh và m cạnh. Một phép duyệt **BFS** trên G tốn chi phí $O(n + m)$. Ngoài ra, tồn tại các thuật toán $O(n + m)$ dựa trên nền tảng BFS để giải các bài toán sau:
 - Kiểm tra tính liên thông của G .
 - Xác định cây tối đại G .
 - Xác định các thành phần liên thông của G .
 - Xác định, với mỗi cạnh $v \in G$, số cạnh tối thiểu cần đi từ s đến v .