TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN



Tên môn học: Cấu trúc dữ liệu và giải thuật

Tên Project: Các thuật toán tìm kiếm trên đồ thị

Thành viên thực hiện:

1. Trần Đức Trung 21280115

2. Lê Hồ Hoàng Anh 21280085

Giảng viên phụ trách: Nguyễn Bảo Long

Mục lục:

1. [Giới thiệu thuật toán DFS, BFS, UCS](#_1._Giới_thiệu) 
   1. Ý tưởng chung
   2. Mã giả
   3. Đánh giá thuật toán
      1. Tính đầy đủ
      2. Tính tối ưu
      3. Độ phức tạp
   4. Ví dụ

2. So sánh giữa các thuật toán

1. So sánh DFS và BFS
2. So sánh UCS và Dijkstra

3. Phân chia công việc và tiến độ hoàn thành

4. Tự đánh giá

# 

# 1. Giới thiệu thuật toán:

A. DFS

a. Ý tưởng chung:

- Từ một đỉnh v đã cho trước, chiến lược tìm kiếm theo chiều sâu (DFS) của quá trình duyệt đồ thị dọc theo đường dẫn từ v càng sâu vào đồ thị trước khi sao lưu. Sau khi thuật toán DFS qua một đỉnh, đỉnh liền kề có thể chưa được thăm.

- Ý tưởng DFS:

+ Bắt đầu từ đỉnh được chọn

+ Tại mỗi đỉnh bất kỳ v

* + - Duyệt đỉnh v
    - Sau đó lần lượt đi tới những đỉnh liền kề với v và chưa được duyệt và lặp lại các thao tác trên đối với những đỉnh này

+ Quay lại đỉnh trước của v

b. Mã giả

*//Traverses a graph beginning at vertex v*

*//by using a depth-first search*

dfs (v: Vertex)

         s = a new empty stack

*// Push v onto the stack and mark it*

            s.push(v)

            Mark v as visited

*// Loop invariant: there is a path from vertex v at the*

*// bottom of the stack s to the vertex at the top of s*

            While(!s.isEmpty()){

                if(no unvisited vertices are adjacent to the vertex on the top of the stack)

                 s.pop() *//Backtrack*

                else{

                  Select an unvisited vertex u adjacent to the vertex on the top of the stack

                  s.push(u)

                  Mark u as visited

                }

        }

}

B. BFS

a. Ý tưởng chung:

- Ý tưởng BFS:

+ Bắt đầu tại đỉnh được chọn

+ Tại mỗi đỉnh bất kỳ v

* Duyệt đỉnh v
* Sau đó đi đến và duyệt các đỉnh liền kề nó
* Tiếp tục lặp lại chiến lược cho các đỉnh liền kề nó

b. Mã giả

*// Traverses a graph beginning at vertex v by using a*

*// breadth-first search: Iterative version.*

bfs(v: Vertex)

q = a new empty queue

*// Add v to queue and mark it*

q.enqueue(v)

Mark v as visited

while (!q.isEmpty()){

q.dequeue(w)

*// Loop invariant: there is a path from vertex w to every vertex in the queue q*

for (each unvisited vertex u adjacent to w){

Mark u as visited

q.enqueue(u)

}

}

# 2. So sánh giữa các thuật toán:

a. Giữa DFS và BFS

|  |  |  |
| --- | --- | --- |
| Tiêu chí | DFS | BFS |
| CTDL được sử dụng | Stack | Queue |
| Định nghĩa | Tiến hành duyệt các nodes xa  nhất có thể cho đến khi không  còn node xung quanh chưa được duyệt | Duyệt toàn bộ nodes trên cùng một mức trước khi di chuyển   tới mức tiếp theo |
| Kỹ thuật | Phải duyệt qua nhiều cạnh để  duyệt đến đỉnh cần tìm từ một  đỉnh nguồn | Có thể sử dụng để tìm kiếm đường đi ngắn nhất trong đồ thị  không trọng số. Vì trong BFS,  có thể duyệt đến một đỉnh với  số cạnh nhỏ nhất từ một đỉnh |
| Thời gian | O(V+E) :Khi là một danh sách kề  O(V^2): Khi là ma trận kề  Với V cạnh và E đỉnh | O(V+E) :Khi là một danh sách kề  O(V^2): Khi là ma trận kề  Với V cạnh và E đỉnh |
| Bộ nhớ | DFS sử dụng ít bộ nhớ hơn | BFS sử dụng nhiều bộ nhớ hơn |
| Space complexity | DFS có ít hơn vì trong cùng  một thời gian DFS chỉ lưu trữ một đường đi đơn từ gốc đến  node lá | Nhiều hơn |
| Tốc độ | DFS nhanh hơn so với BFS | BFS chậm hơn so với DFS |