Mô tả thuật toán tìm đường đi Hamilton

Đường đi Hamilton là đường đi trong đồ thị có hướng hoặc vô hướng, đi qua mỗi đỉnh đúng một lần.

Ở đây, em xây dựng thuật toán tìm đường đi Hamilton bằng cách sử dụng backtracking.

```
vector<vector<int> > graph;
vector<int> path;
vector<int> rs;
int N, M;
vector<bool> visited;
bool flag = false;
```

Đầu tiên chúng ta sẽ khai báo các biến cần sử dụng:

- Biến graph là một vector hai chiều dùng để lưu đồ thị.
- Biến path là một vector số nguyên dùng để lưu đường đi có thể sau mỗi lần đệ quy.
- Biến rs là biến sẽ lưu lại đường đi Hamilton đầu tiên chúng ta tìm được và là kết quả của bài toán.
- N là số đỉnh của đồ thị và M là số đường đi.
- vector<bool> visited là vector dùng để kiểm tra một đỉnh đã được đi đến hay chưa trong một nhánh đệ quy.
- Và biến flag dùng để kiểm tra xem đã tìm được đường đi Hamilton hay chưa.

```
void HamiltonPath (int u) {
    if (path.size() == N) {
        rs = path;
        flag = true;
        return;
    }
```

Chúng ta có hàm tìm đường đi Hamilton như sau:

- Đầu tiên chúng ta sẽ có base case như trên: Khi path.size() == N tức là chúng ta đã đi đến hết tất cả các đỉnh. Chúng ta sẽ lưu lại path tại thời điểm đó và đưa flag về true để dừng việc đệ quy lại.

```
    for (int v : graph[u]) {
        if (!visited[v]) {
            visited[v] = true;
            path.push_back(v);
            HamiltonPath(v);
            if (flag)
                return;
            path.pop_back();
            visited[v] = false;
        }
    }
```

Và các bước đệ quy như sau:

- Chúng ta sẽ đi lần lượt các đỉnh v nằm trong graph[u] ở đây hiểu là các đỉnh có thể đi từ đỉnh u.
- Nếu đỉnh v chưa đi đến tức là visited[v] == false thì chúng ta sẽ đi đến đỉnh v. Nếu không bỏ qua.
- Chúng ta sẽ đánh visited[v] = true và đưa v vào path cho lần đệ quy tiếp theo trên nhánh đó.
- Sau khi đệ quy xong, nếu flag == true thì tức là chúng ta đã tìm được đường đi Hamilton chúng ta sẽ kết thúc đệ quy.
- Và khi đệ quy xong một nhánh (khi vẫn chưa tìm được đường đi Hamilton tức flag = false), để bảo đảm lần đệ quy của một nhánh khác được chính xác theo lý thuyết của backtracking, chúng ta sẽ bỏ đỉnh v vừa thêm vào vector path. Và đánh lại visited[v] = false.

Đó là hàm tìm đường đi Hamilton.

Và cuối cùng dưới đây là hàm main: (Ở trang sau)

```cpp
int main() {
    cin >> N >> M;
    graph.assign(N, vector<int>());
    for (int i = 0; i < M; ++i) {
        int a, b;
        cin >> a >> b;
        graph[a-1].push_back(b-1);
        graph[b-1].push_back(a-1);
    }
    visited.assign(N, false);
    visited[0] = true;
    path.push_back(0);
    HamiltonPath(0);

    for (int i = 0; i < rs.size(); ++i) {
        cout << rs[i] + 1 << " ";
    }
    cout << endl;
    return 0;
}
```

Khi chạy test case:

```
PS C:\Users\tbm23\OneDrive\Desktop\algorithm\PTTKTT> g++ .\A30080_TranBinhMinh_0938325211_Bai1_DuongDiHamiltion.cpp
PS C:\Users\tbm23\OneDrive\Desktop\algorithm\PTTKTT> .\a.exe
5 6
1 2
1 4
4 5
2 5
2 3
3 5
1 2 3 5 4
PS C:\Users\tbm23\OneDrive\Desktop\algorithm\PTTKTT>
```