پیدا کردن مؤلفه‌های همبند

تعریف

در گراف‌های بدون‌جهت، دو رأس متعلق به یک مؤلفه همبندی هستند اگر و تنها اگر یک مسیر بین آن دو با استفاده از یال‌های گراف باشد. مسئله پیدا کردن و افراز به این مؤلفه‌هاست. این مسئله در گراف‌های جهت‌دار را مؤلفه‌های قویاهمبند می‌نامند.

الگوریتم

کافیست روی گراف پیمایش کنیم و از آنجا که پیمایش با هر دو الگوریتم جست‌وجوی عمق‌اول و جست‌وجوی سطح‌اول تمام رأس‌های مؤلفه‌ی رأس آغازین را میدهد، پس کافیست به ازای تمام رأس‌هایی که مؤلفه‌ی آن‌ها پیدا نشده است، الگوریتم را اجرا کنیم و در طول الگوریتم آن‌ها را با رنگ خاصی رنگ کنیم.

پیچیدگی الگوریتم

همان پیچیدگی الگوریتم‌های پیمایش است که در نتیجه برابر $O(n+e)$ میشود.

پیاده‌سازی اولیه

در پیاده‌سازی فرض کرده‌ایم که شماره اولین رأس، ۰ است و تعداد رأس‌ها n و لیست مجاورت رأس‌ها نیز داده شده است.

```
#include <iostream>
#include <vector>

const int MAXN = 100 * 1000 + 10;
using namespace std;

bool mark[MAXN];
vector<int> comp[MAXN];  //  لیست مؤلفه های همبندی در این آرایه قرار می‌گیرد
vector <int> adj[MAXN];
int n;   // تعداد رأس‌ها
int m;   // تعداد یال‌ها

void dfs(int v, int c) {
    mark[v] = 1;
    comp[c].push_back(v);
    for(int i = 0; i < adj[v].size(); i++) {
        int u = adj[v][i];
        if(mark[u] != 1)
            dfs(u, c);
    }
}

int find_comp() {
    int cnt = 0;  //  تعداد مؤلفه‌ها
    for(int i = 0; i < n; i++)
        if (mark[i] == 0)
            dfs(i, cnt++);
    return cnt;
}


void input()
{
```

```
    cin >> n >> m;
    for (int i = 0; i < m; i++) {
        int v, u;
        cin >> v >> u;
        adj[--v].push_back(--u);
        adj[u].push_back(v);
    }
}

int main()
{
    input();
    cout << find_comp() << endl;    // تعداد مؤلفه‌ها
}
```

مراجع

مؤلفه‌های همبند در ویکی‌پدیا [illegible]