图匹配问题模板

一般图最大匹配

```
/* luoguP6113 【模板】一般图最大匹配
输入：
10 10
4 3
3 1
4 7
2 10
2 9
3 10
5 9
4 6
1 10
1 7
输出：
4
7 9 10 6 0 4 1 0 2 3
*/
#include <bits/stdc++.h>
using namespace std;
struct blossom {  // Blossom Algorithm
    int n, vis_t;
    vector<vector<int>> E;
    vector<int> match, label, org, vis, parent;
    queue<int> Q;
    blossom(int _n) {
        n = _n;
        E = vector<vector<int>>(n, vector<int>());
        match.assign(n, -1);
        label.resize(n);
        org.resize(n);
        iota(org.begin(), org.end(), 0);
        parent.assign(n, -1);
        vis.assign(n, 0);
        vis_t = 0;
    }
    void addEdge(int u, int v) {
        E[u].emplace_back(v);
```

```
        E[v].emplace_back(u);
    }
    int lca(int v, int u) {
        vis_t++;
        while (true) {
            if (v != -1) {
                if (vis[v] == vis_t) {
                    return v;
                }
                vis[v] = vis_t;
                if (match[v] == -1) {
                    v = -1;
                } else {
                    v = org[parent[match[v]]];
                }
            }
            swap(v, u);
        }
    }
    void agument(int v) {
        while (v != -1) {
            auto pv = parent[v];
            auto nxt = match[pv];
            match[v] = pv;
            match[pv] = v;
            v = nxt;
        }
    }
    void flower(int v, int u, int a) {
        while (org[v] != a) {
            parent[v] = u;
            u = match[v];
            if (label[u] == 1) {
                label[u] = 0;
                Q.emplace(u);
            }
            org[v] = org[u] = a;
            v = parent[u];
        }
    }
    bool bfs(int root) {
        fill(label.begin(), label.end(), -1);
        iota(org.begin(), org.end(), 0);
        while (!Q.empty()) {
            Q.pop();
        }
        Q.emplace(root);
        label[root] = 0;
        while (!Q.empty()) {
            auto u = Q.front();
            Q.pop();
            for (auto v : E[u]) {
                if (label[v] == -1) {
                    label[v] = 1;
                    parent[v] = u;
                    if (match[v] == -1) {
                        agument(v);
                        return true;
```

```
                    }
                    label[match[v]] = 0;
                    Q.push(match[v]);
                    continue;
                } else if (label[v] == 0 && org[v] != org[u]) {
                    auto a = lca(org[u], org[v]);
                    flower(v, u, a);
                    flower(u, v, a);
                }
            }
        }
        return false;
    }
    void solve() {
        for (int i = 0; i < n; ++i) {
            if (match[i] == -1) {
                bfs(i);
            }
        }
    }
};

void Solve() {
    int n, m;
    cin >> n >> m;
    blossom G(n);
    for (int i = 0; i < m; ++i) {
        int u, v;
        cin >> u >> v;
        u--, v--;
        G.addEdge(u, v);
    }
    G.solve();
    int ans = 0;
    for (int i = 0; i < n; ++i) {
        if (G.match[i] != -1) {
            ans++;
        }
    }
    cout << ans / 2 << "\n";
    for (int i = 0; i < n; ++i) {
        cout << G.match[i] + 1 << " ";
    }
}

signed main(){
    int T=1;
    while(T--){
        Solve();
    }
    return 0;
}
```

一般图最大权匹配

```
/* luoguP6699 【模板】一般图最大权匹配
输入：
7 20
5 7 9
3 7 4
3 6 6
2 5 8
5 1 9
1 3 6
6 5 1
2 7 4
2 3 5
6 4 2
7 1 5
5 4 4
4 1 3
5 3 9
7 6 4
2 1 3
4 3 9
6 2 7
4 2 8
6 1 10
输出：
28
6 0 4 3 7 1 5
*/
#include<bits/stdc++.h>
#define int long long
#define inf 0x3f3f3f
using namespace std;
const int N=805;
struct E{
    int u,v,w;
    E(){};
    E(int u,int v,int w):u(u),v(v),w(w){}
}g[N][N];
int n,n_x;
int lab[N];
int match[N],slack[N],st[N],pa[N];
int flower_from[N][N],S[N],vis[N];
vector<int>flower[N];
queue<int>q;

inline int e_delta(const E &e){ // does not work inside blossoms
    return lab[e.u]+lab[e.v]-g[e.u][e.v].w*2;
}

inline void update_slack(int u,int x){
    if(!slack[x]||e_delta(g[u][x])<e_delta(g[slack[x]][x]))slack[x]=u;
}
inline void set_slack(int x){
    slack[x]=0;
    for(int u=1;u<=n;++u)
        if(g[u][x].w>0&&st[u]!=x&&S[st[u]]==0)update_slack(u,x);
```

```
}
void q_push(int x){
    if(x<=n)q.push(x);
    else for(size_t i=0;i<flower[x].size();i++)q_push(flower[x][i]);
}
inline void set_st(int x,int b){
    st[x]=b;
    if(x>n)for(size_t i=0;i<flower[x].size();++i)
            set_st(flower[x][i],b);
}
inline int get_pr(int b,int xr){
    int pr=find(flower[b].begin(),flower[b].end(),xr)-flower[b].begin();
    if(pr%2==1){//檢查他在前一層圖是奇點還是偶點
        reverse(flower[b].begin()+1,flower[b].end());
        return (int)flower[b].size()-pr;
    }else return pr;
}

inline void set_match(int u,int v){
    match[u]=g[u][v].v;
    if(u>n){
        E e=g[u][v];
        int xr=flower_from[u][e.u],pr=get_pr(u,xr);
        for(int i=0;i<pr;++i)set_match(flower[u][i],flower[u][i^1]);
        set_match(xr,v);
        rotate(flower[u].begin(),flower[u].begin()+pr,flower[u].end());
    }
}

inline void augment(int u,int v){
    for(;;){
        int xnv=st[match[u]];
        set_match(u,v);
        if(!xnv)return;
        set_match(xnv,st[pa[xnv]]);
        u=st[pa[xnv]],v=xnv;
    }
}

inline int get_lca(int u,int v){
    static int t=0;
    for(++t;u||v;swap(u,v)){
        if(u==0)continue;
        if(vis[u]==t)return u;
        vis[u]=t;//這種方法可以不用清空v陣列
        u=st[match[u]];
        if(u)u=st[pa[u]];
    }
    return 0;
}

inline void add_blossom(int u,int lca,int v){
    int b=n+1;
    while(b<=n_x&&st[b])++b;
    if(b>n_x)++n_x;
    lab[b]=0,S[b]=0;
    match[b]=match[lca];
    flower[b].clear();
```

```
    flower[b].push_back(lca);
    for(int x=u,y;x!=lca;x=st[pa[y]])
        flower[b].push_back(x),flower[b].push_back(y=st[match[x]]),q_push(y);
    reverse(flower[b].begin()+1,flower[b].end());
    for(int x=v,y;x!=lca;x=st[pa[y]])
        flower[b].push_back(x),flower[b].push_back(y=st[match[x]]),q_push(y);
    set_st(b,b);
    for(int x=1;x<=n_x;++x)g[b][x].w=g[x][b].w=0;
    for(int x=1;x<=n;++x)flower_from[b][x]=0;
    for(size_t i=0;i<flower[b].size();++i){
        int xs=flower[b][i];
        for(int x=1;x<=n_x;++x)
            if(g[b][x].w==0||e_delta(g[xs][x])<e_delta(g[b][x]))
                g[b][x]=g[xs][x],g[x][b]=g[x][xs];
        for(int x=1;x<=n;++x)
            if(flower_from[xs][x])flower_from[b][x]=xs;
    }
    set_slack(b);
}

inline void expand_blossom(int b){
    for(size_t i=0;i<flower[b].size();++i)
        set_st(flower[b][i],flower[b][i]);
    int xr=flower_from[b][g[b][pa[b]].u],pr=get_pr(b,xr);
    for(int i=0;i<pr;i+=2){
        int xs=flower[b][i],xns=flower[b][i+1];
        pa[xs]=g[xns][xs].u;
        S[xs]=1,S[xns]=0;
        slack[xs]=0,set_slack(xns);
        q_push(xns);
    }
    S[xr]=1,pa[xr]=pa[b];
    for(size_t i=pr+1;i<flower[b].size();++i){
        int xs=flower[b][i];
        S[xs]=-1,set_slack(xs);
    }
    st[b]=0;
}
inline bool on_found_edge(const E &e){
    int u=st[e.u],v=st[e.v];
    if(S[v]==-1){
        pa[v]=e.u,S[v]=1;
        int nu=st[match[v]];
        slack[v]=slack[nu]=0;
        S[nu]=0,q_push(nu);
    }else if(S[v]==0){
        int lca=get_lca(u,v);
        if(!lca)return augment(u,v),augment(v,u),true;
        else add_blossom(u,lca,v);
    }
    return false;
}

inline bool matching(){
    memset(S+1,-1,sizeof(int)*n_x);
    memset(slack+1,0,sizeof(int)*n_x);
    q=queue<int>();
    for(int x=1;x<=n_x;++x)
```

```
        if(st[x]==x&&!match[x])pa[x]=0,S[x]=0,q_push(x);
    if(q.empty())return false;
    for(;;){
        while(q.size()){
            int u=q.front();q.pop();
            if(S[st[u]]==1)continue;
            for(int v=1;v<=n;++v)
                if(g[u][v].w>0&&st[u]!=st[v]){
                    if(e_delta(g[u][v])==0){
                        if(on_found_edge(g[u][v]))return true;
                    }else update_slack(u,st[v]);
                }
        }
        int d=inf;
        for(int b=n+1;b<=n_x;++b)
            if(st[b]==b&&S[b]==1)d=min(d,lab[b]/2);
        for(int x=1;x<=n_x;++x)
            if(st[x]==x&&slack[x]){
                if(S[x]==-1)d=min(d,e_delta(g[slack[x]][x]));
                else if(S[x]==0)d=min(d,e_delta(g[slack[x]][x])/2);
            }
        for(int u=1;u<=n;++u){
            if(S[st[u]]==0){
                if(lab[u]<=d)return 0;
                lab[u]-=d;
            }else if(S[st[u]]==1)lab[u]+=d;
        }
        for(int b=n+1;b<=n_x;++b)
            if(st[b]==b){
                if(S[st[b]]==0)lab[b]+=d*2;
                else if(S[st[b]]==1)lab[b]-=d*2;
            }
        q=queue<int>();
        for(int x=1;x<=n_x;++x)
            if(st[x]==x&&slack[x]&&st[slack[x]]!=x&&e_delta(g[slack[x]][x])==0)
                if(on_found_edge(g[slack[x]][x]))return true;
        for(int b=n+1;b<=n_x;++b)
            if(st[b]==b&&S[b]==1&&lab[b]==0)expand_blossom(b);
    }
    return false;
}

inline pair<long long,int> weight_blossom(){
    memset(match+1,0,sizeof(int)*n);
    n_x=n;
    int n_matches=0;
    long long tot_weight=0;
    for(int u=0;u<=n;++u)st[u]=u,flower[u].clear();
    int w_max=0;
    for(int u=1;u<=n;++u)
        for(int v=1;v<=n;++v){
            flower_from[u][v]=(u==v?u:0);
            w_max=max(w_max,g[u][v].w);
        }
    for(int u=1;u<=n;++u)lab[u]=w_max;
    while(matching())++n_matches;
    for(int u=1;u<=n;++u)
        if(match[u]&&match[u]<u)
```

```
            tot_weight+=g[u][match[u]].w;
    return make_pair(tot_weight,n_matches);
}
inline void init_weight_graph(){
    for(int u=1;u<=n;++u)
        for(int v=1;v<=n;++v)
            g[u][v]=E(u,v,0);
}

signed main(){
    int m;
    scanf("%lld%lld",&n,&m);
    init_weight_graph();
    for(int i=0;i<m;++i){
        int u,v,w;
        scanf("%lld%lld%lld",&u,&v,&w);
        g[u][v].w=g[v][u].w=w;
    }
    printf("%lld\n",weight_blossom().first);
    for(int u=1;u<=n;++u)printf("%lld ",match[u]);
    puts("");
    return 0;
}
```

例题

[zoj3316 Game]

题意：一个棋盘上有n个棋子，两个人做游戏，拿的棋子和上一个被拿的棋子的曼哈顿距离不能超过L，问后手能否赢。

思路：最大匹配等于n的时候才能赢，不然就会输。

[HNCPC Multi-university Training Round3 G - Cooking]

题意：给定n$(n \leq 10)$个厨师可做的菜数、每道菜所需要做的次数$a_i(a_i \leq 50)$。每次可以选两道菜并花$c_{i,j}(c_{i,j} < 100)$的时间去做，问最小需要的总时间，如果无解输出-1。

思路：数据很小，拆成$\sum a$个点然后两两做最小权匹配，套个带花树的板子即可。

参考资料

https://blog.csdn.net/birdmanqin/article/details/100160999