图论

北京大学 洪华敦

图的基本概念

- 根据边的方向分为两种：有向图，无向图
- 路径：从某个点出发经过若干条边到达另一个点
- 简单路径：不重复经过某个点的路径
- 度数：无向图中一个点连出去的边的个数
- 连通：两个点通过路径相连
- 拓扑图：没有环的图

最短路问题

- 给定一张n个点m条边的有向图，每条边的长度为1，求1号点到其他点最短的路径

- 如果 1 到 x 的最短路是 L，那么肯定存在一个 y，使得 1 到 y 的最短路是 L-1，且 y 到 x 有边

- 用队列维护这个“扩展”的过程

- 时间复杂度：O(n+m)

最短路问题

- 给定一张n个点m条边的有向图，每条边的长度是 0 或者 1，求1号点到其他点最短的路径
- 魔改一下 BFS

最短路问题

- 给定一张n个点m条边的有向图，每条边的长度为1 或者2，求1号点到其他点最短的路径

- 把长度为2的边拆成2条长度为1的边

- 时间复杂度：O(n+m)

最短路问题

- 给定一张n个点m条边的正权有向图，求1号点到每个点的最短路
- 受到BFS算法的启发，我们只要按照最短路的大小从小到大加入到队列里就行了，需要用优先队列维护
- 时间复杂度：O(n+mlogn) 或者 O(n^2+m)

最短路问题

- 给定一张n个点m条边的有向图，求1号点到每个点的最短路
- Bellman-ford 算法：每次拿每条边去更新最短路
-
```
for(int k = 1 ; k <= n - 1 ; k ++)
{
    for(int i = 1 ; i < m ; i ++)
    {
        if(dis[v[i]] > dis[u[i]] + w[i])
            dis[v[i]] = dis[u[i]] + w[i] ;
    }
}
```

SPFA

- 一条边 (u,v)，在 dis[u] 没变的情况下不应该重新更新

- 用队列记录要进行扩展的点，每次 dis[x] 被更新后就把 x 扔进队列里去更新其他点的最短路

-
```
for(int k = 1 ; k <= n - 1 ; k ++)
{
    for(int i = 1 ; i < m ; i ++)
    {
        if(dis[v[i]] > dis[u[i]] + w[i])
            dis[v[i]] = dis[u[i]] + w[i] ;
    }
}
```

如何卡SPFA

- 建一个n*m的网格图，其中行数很少，列数很多
- 之后行与行之间的边权都较小，列与列之间的边权都较大即可

判断负环

- 一个有向图有负环等价于某个点的最短路长度>n
- 方法1：某个点入队次数>n时就有负环
- 方法2：记录 len[x]，表示 1 到 x 的最短路的点的个数，len[x]>=n 时存在负环
- 时间复杂度：O(nm)

最短路

- 给定一张 n 个点 m 条边的正权有向图，求每两个点之间的最短路

- 设 F(K,X,Y) 表示 X 到 Y 的路径中，满足路径上的点的标号都不超过 K 的最短路径

- F(K,X,Y)=Min(F(K-1,X,Y) , F(K-1,X,K)+F(K-1,K,Y))

- 时间复杂度：O(n^3)

差分约束问题

- 假设对于一张图来说，1号点到 x 的最短路长度为 d[x]
- 则我们有：d[y]<=d[x]+w[x][y]
- 最短路问题可以给出这一类不等式的最大解，取个反可以得到最小解

差分约束的应用

- 给定n,m和m个三元组(l,r,k)，求一个01串 s，使得s[l…r] 中1的个数至少有 k 个，求 1 最少的满足条件的01串

- n,m<=10000

差分约束的应用

- 设 f[i] 为 s[1…i] 中 1 的个数
- f[i]<=f[i-1]+1
- f[i-1]<=f[i]
- f[l-1]<=f[r]-k
- 等价于求该不等式组中 f[n] 的最大解

次短路

- 如何求出 1 号点到 N 号点的次短路
- 定义：若 d[v]=d[u]+w(u,v)，则称 (u,v) 是最短路图上的边
- 一条次短路一定至多经过一条非最短路图上的边

例题

- 给定一张有向带正权拓扑图，求有几条 1 到 N 的路径的长度<= 1 到 N 的最短路+K

- N,M<=10^5，K<=100，边权<=10^9

最短路结合动态规划

- 定义一条路径 (X…Y) 的冗余度为它的长度减去 X…Y 的最短路长度
- 则题目要求的是 1…N 冗余度不超过 K 的路径长度
- 考虑路径 (1,a,b,N)
- 冗余度为 $w(1,a)+w(a,b)+w(b,N)-d(1,N)$
- $=w(1,a)+w(a,b)+w(b,N)-d(a,N)-d(1,N)+d(a,N)$
- $=[w(1,a)+d(a,N)-d(1,N)]+w(a,b)+w(b,N)-d(a,N)$
- 设 $p(x,y)=w(x,y)+d(y,N)-d(x,N)$，则冗余度变成了 a…N 的冗余度加上 $p(1,a)$

最短路结合动态规划

- 设 F(X,L) 表示 1 到 X 有几条路径满足 p 的和为 L
- F(X,L)=sum(F(Y,L-P(Y,X)))
- 时间复杂度：O(mlogn+mk+n)

图的连通性问题

- 无向图中的连通：两个点存在一条连接它们的路径
- 连通块：里面的点两两连通的块

维护连通性

- 对于每个连通块维护一棵有根树，F(x) 表示点 x 的父亲
- 则假设我们要添加一条边 U,V，首先求出U,V所在连通块的有根树树根 X,Y，然后令 F(X)=Y
- GetRoot(X)：若 F(X)=0 则 return X，否则 return F(X)=GetRoot(F(X))

桥

- 对于一个连通无向图，定义一条边 (u,v) 是桥，当且仅当断开这条边后图变得不连通

- 图的大致形状：O-O，则中间的边就是桥

- 强连通分量：没有桥的连通块

Tarjan算法

- 对整个图进行 dfs，设 dfn[x] 表示点 x 是第几个被搜到的
- low[x] 表示：x 通过非返祖边，且至多通过一条非树边能到达的最小 dfn

```
void tarjan(int x){
    instack[x]=1;
    dfn[x]=low[x]=++tot;
    for(int y:go[x])if(!dfn[y]){
        tarjan(y);
        low[x]=min(low[x],low[y]);
    }
    else{
        if(instack[y])low[x]=min(low[x],dfn[y]);
    }
    instack[x]=0;
}
```

Tarjan算法求桥

- 非树边一定不是桥
- 对于树边 (U,V)，它是桥等价于 low[V]>dfn[U]
- 去掉所有的桥后，剩下的每个连通分量都是强连通分量

有向图的强连通分量

- 在做 Tarjan算法时，如果 tarjan(x) 后发现 dfn[x]=low[x]，则 x 的子树里的剩下的所有点构成一个强连通分量

例题

- 有N个人，给你M对整数 (a,b)，表示第 a 个人认为 b 很厉害，而这种关系具备传递性，也就是如果 a 认为 b 厉害，且 b 认为 c 厉害，则 a 认为 c 厉害
- 求有多少人被所有人都觉得很厉害
- N,M<=10^5

例题

- 等价于求有几个点，使得他们能到达所有点
- 对于无环图来说非常简单
- 可以发现每个强连通分量中的人都是互相认为对方很厉害的，所以可以把它们看成一个人
- 求出强连通分量并缩点，之后得到一个无环图就能做了

欧拉回路问题

- 给定一张有向图，如何求出一条经过每条边恰好一次的回路
- 必要条件：
- 1. 这张图是个强连通分量
- 2. 每个点的出度等于入度
- 可以发现这两个条件同时也是充分的

圈套圈算法

- 任选一个起点，从起点开始 dfs，每条边只能被走一遍，当没有边可以走的时候把 x 压入答案的队列中
- 最后的答案是反着的欧拉回路

二进制

- 有 n 个灯泡，灯泡有两种状态：开和关，每次可以操作某个灯泡，使得它状态取反
- 定义一个局面为每个灯泡的状态以及最后一个被操作的灯泡的编号，可以将一个局面看成一个二元组
- (0…2^N-1,0…N-1)
- 现在你可以选定任何初始局面，求最少几步遍历所有局面
- N<=20

二进制

- 建立 2^N 个点，将每个局面看成边，求欧拉回路

树

- 树：N 个点 N-1 条边的连通无向图，分为有根树和无根树
- 树的叶子：度数为 1 的点

最小生成树

- 给定一张 n 个点的带权无向图，求权值和最小的生成树
- Kruskal 算法：将边按照权值大小从小到大排序，之后能加就加，用并查集维护
- 证明：证明权值最小的边一定在最小生成树中，然后归纳法

Prufer序列

- 将一棵树变成一个序列：
- 每次选择树上标号最小的叶子，删掉它，将与它相连的那个点的标号加到序列里，直到只剩下 2 个点
- 可以证明：任意一个长度为 n-2 的 1…n 的序列都是某棵树的 Prufer 序列
- 所以可以推出：n 个点的无根树个数为 n^(n-2)

例题

- 给定每个点的度数 d[i]，求有几棵这样的无根树。

例题

- Prufer 序列中，度数为 k 的点一共会出现 k-1 次
- 相当有给定了序列中每个数的出现次数，求有几种合法的序列
- 根据排列组合知识，答案是：(n-2)!/((d[1]-1)!(d[2]-1)!…(d[n]-1)!)

二分图

- 可以分成两部分，使得这两部分内部没有边的图
- 一个图是二分图等价于该图没有奇环

二分图匹配

- 给定一张二分图，求它的最大匹配

- 匈牙利算法：每次找一条增广路

```
bool findpath(int x){
    vis[x]=1;
    for(int y:go[x]){
        if(!mat[y]){
            mat[y]=x;
            return 1;
        }
        else{
            int z=mat[y];
            if(!vis[z])if(findpath(z)){
                mat[y]=x;
                return 1;
            }
        }
    }
}
```

-

二分图

- 最小顶点覆盖：选最少的点覆盖所有边
- |二分图最小顶点覆盖|=|二分图最大匹配|
- 最大独立集：选最多的点使得它们两两没边相连
- |二分图最大独立集|=总点数-|二分图最小顶点覆盖|

三元环计数

- 求无向图的三元环个数
- 将点按照度数为第一关键字，标号为第二关键字从小到大排序，定义排序后每个点的序为 pos[x]
- 对于每条无向边，变成有向边：pos较小的点连向pos较大的点
- 这样连完后每个点的出度都最多只有 N^0.5

三元环计数

- 枚举三元环中 pos 最小的点 x，然后枚举 x 的出边 y，再枚举 y 的出边 z，如果 z 也在 x 的出边中的话就得到一个三元环

- 时间复杂度：O(N^1.5)

习题1

- 给定 N 和一个数组 a[1…N]，定义边 (i,j) 的边权为 a[i] xor a[j]，求这张完全图的最小生成树的边权和
- N<=10^5，a[i]<=10^9

习题2

- 有一个 N*M 的矩形，其中有 K 个格子中有病毒，现在你可以进行若干次消毒，每次你可以选择一个任意大小的子矩形进行消毒，假设是 A*B 的矩形，则代价是 min(A,B)，要求你用最少的代价进行消毒

- N,M,K<=5000

习题3

- 给定一个 N*N 的 01 矩阵，你可以任意进行 行交换操作和列交换操作，问是否能使这个矩阵的主对角线上都是 1
- N<=300

习题4

- 有 N 张卡牌，每张卡牌正反面各有一个数，求有几种翻面的方式使得这 N 个数字互不相同
- N<=10^5

习题5

- 你现在很想知道一个数列 A[1…N] 是啥，但是需要花费代价去获取情报，你可以花费 Cost[L][R] 的值去得到 A[L…R] 的和，给定 Cost，求最少花费多少代价才能确定 A[1…N]

- N<=1000，Cost[L][R]<=10^9

习题6

- 给定一个无向图，求它的四元环个数
- N<=50000

习题7

- 有 N 块农田，你可以花费 W[i] 在第 i 块田上建一个水库，也可以花费 P[i][j] 从第 j 块田上把水引到第 i 块田上，求使得所有田都有水的最小代价
- N<=500