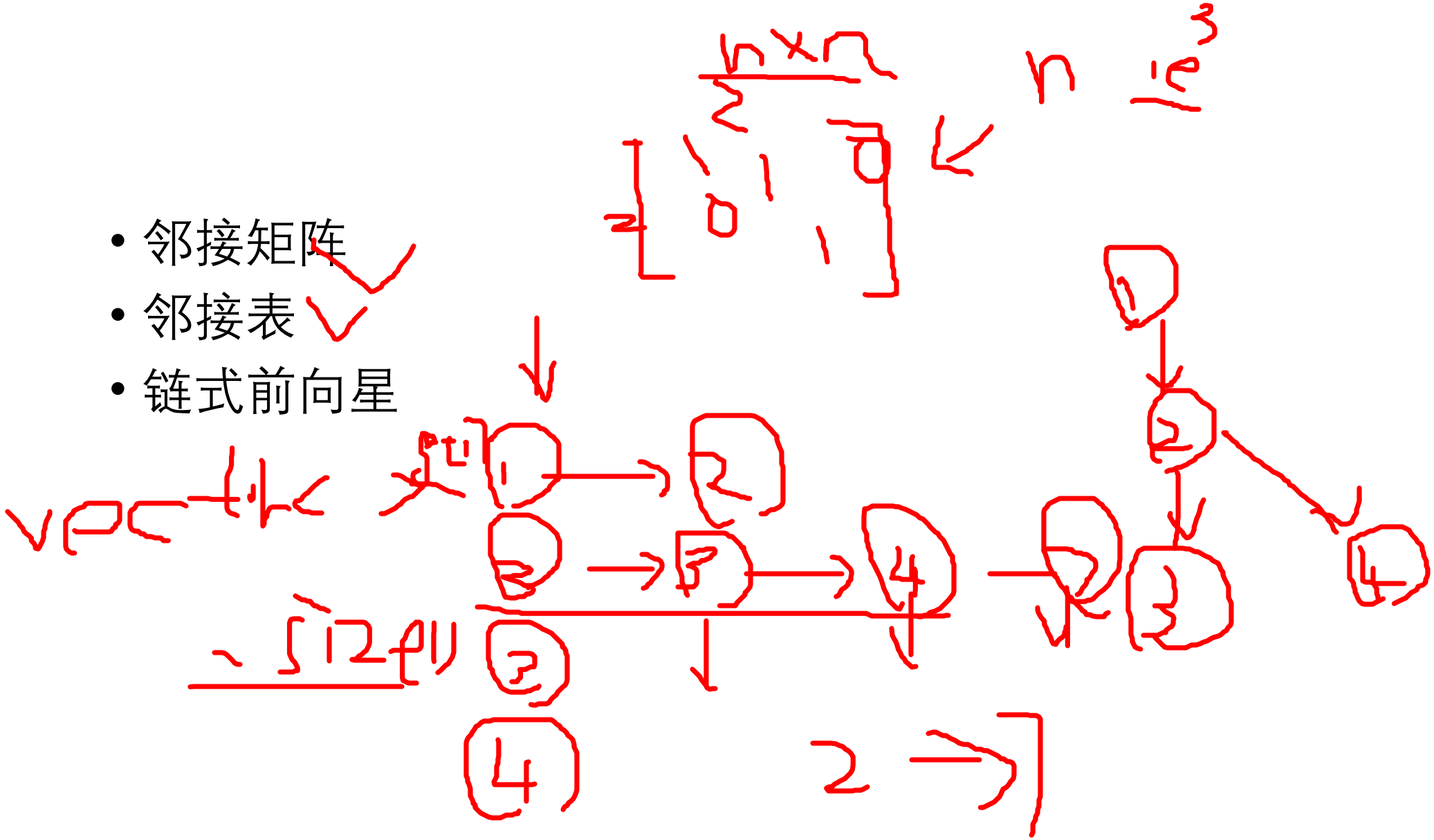
建图

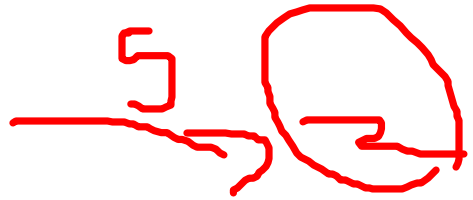
- 邻接矩阵
- 邻接表
- 链式前向星



邻接矩阵

- int e[V][V]
- e[i][j]：i到j的边长
- 适用于稠密图
- 加一条从f到t，长度为l的单向边：e[f][t]=l;(若可能有重边，则根据题意取max或min)
- 加双向边：加两条单向边

邻接表（最常用）

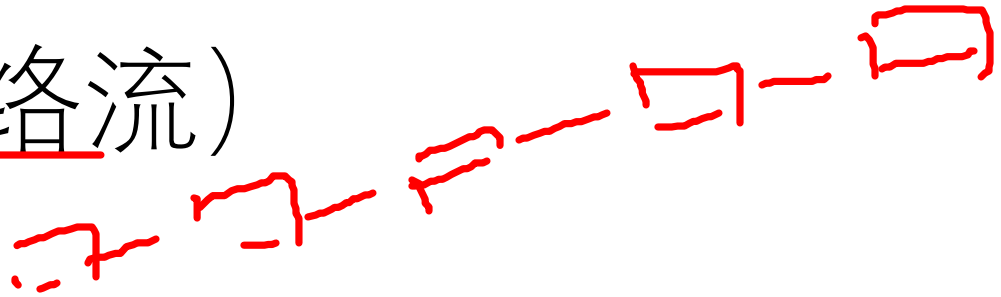


- vector<Edge> e[V];
- e[i[中存放由i出发的所有边
- Edge：结构体，储存边的信息
- eg: 对于有长度的边
- struct Edge{
  - int to,len;
  - Edge(){}
  - Edge(int t,int l){to=t,len=l;}
- }
- 加f到t,长度为l的单向边：e[f].push_back(Edge(t,l));
- e[f]中有从f出发的所有边。

简单直接的定义：

```
#include <vector>
using namespace std;
struct edge{int to;int cost;};
vector<edge>g[50005];
```

链式前向星（多用于网络流）



- 一般情况下可用邻接表代替
- 本质是缠在一起的很多个链表
- int head[V],nxt[E],to[E],len[E],cnte=0;
- 加从f到t长度为l的单向边：
- void addedge(int f,int t,int l){
  - to[cnte]=t;
  - len[cnte]=l;
  - nxt[cnte]=head[f];
  - head[f]=cnte++;
- }
- 使用前需将head清空为-1，cnte赋值为0

- 遍历从f出发的所有边：
- for(int i=head[f];i!=-1;i=nxt[i]){
  - //此边指向to[i]，长为len[i]
- }

并查集

ufs

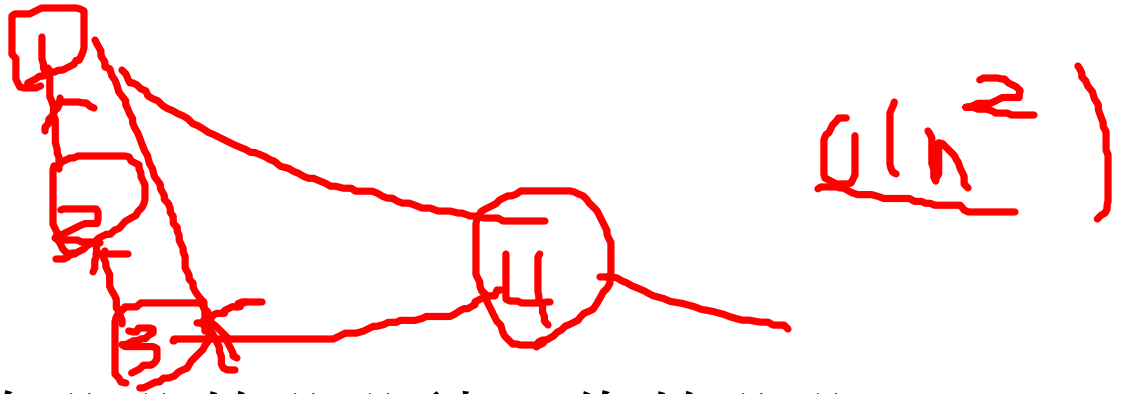
并查集的一般功能

- 连接两个节点
- 判断两个节点是否连通

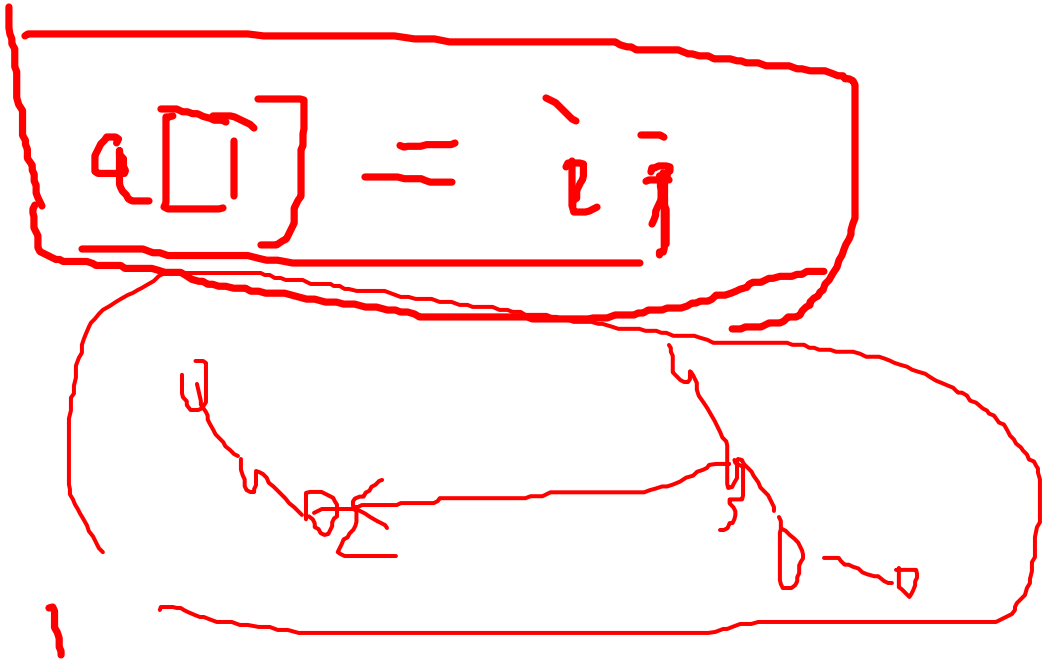
朴素写法

- $O(n^2)$？
- 不放代码了，没人这么写。

路径压缩优化



- 如果你不是你的爸爸，那么你爸爸的爸爸就是你的爸爸。
- int find(int o)//查找根
- {
- if(fa[o]!=o)fa[o]=find(fa[o]);
- return fa[o];
- }



模板

```
struct UFS
{
    int fa[MAX];
    void init(int n)//初始化
    {
        for(int i=0;i<=n;++i)fa[i]=i;
    }
    int find(int o)//查找根
    {
        if(fa[o]!=o)fa[o]=find(fa[o]);
        return fa[o];
    }
    int unite(int u,int v)//连接
    {
        u=find(u);v=find(v);
        if(u!=v)fa[u]=v;
    }
    bool united(int u,int v)//是否连通
    {
        return find(u)==find(v);
    }
};
```

时间复杂度

- O(nlogn)
- 对，只有路径压缩优化的并查集能被卡到O(nlogn)，但一般没人卡。
- 要想得到真正$O(n\alpha(n))$的并查集需要按秩合并（然而我不知道此时应该按长度合并还是按重量合并，我也没写过两种优化都加的并查集）。

按秩合并

- 可按大小、深度合并

代码

```
struct UFS
{
    int fa[MAX],siz[MAX];
    void init(int n)//初始化
    {
        for(int i=0;i<=n;++i)
        {
            fa[i]=i;
            siz[i]=1;
        }
    }
    int find(int o)//查找根
    {
        while(fa[o]!=o)o=fa[o];
        return o;
    }
    int unite(int u,int v)//连接
    {
        u=find(u);v=find(v);
        if(u!=v)
        {
            if(siz[u]>siz[v])swap(u,v);
            fa[u]=v;
            siz[v]+=siz[u];
        }
    }
    bool united(int u,int v)//是否连通
    {
        return find(u)==find(v);
    }
};
```

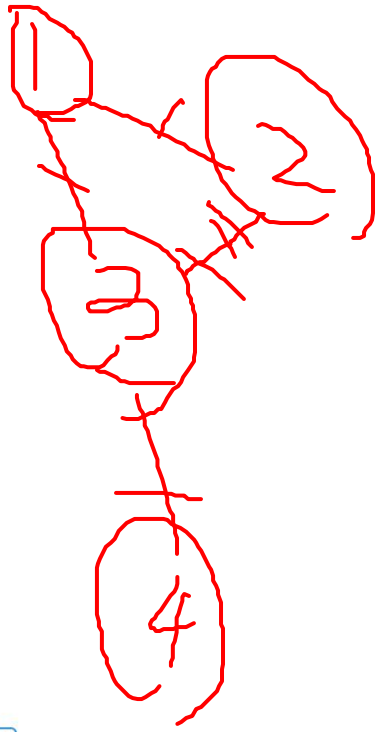
```
struct UFS
{
    int fa[MAX],dep[MAX];
    void init(int n)//初始化
    {
        for(int i=0;i<=n;++i)
        {
            fa[i]=i;
            dep[i]=1;
        }
    }
    int find(int o)//查找根
    {
        while(fa[o]!=o)o=fa[o];
        return o;
    }
    int unite(int u,int v)//连接
    {
        u=find(u);v=find(v);
        if(u!=v)
        {
            if(dep[u]>dep[v])swap(u,v);
            fa[u]=v;
            if(dep[u]==dep[v])++dep[v];
        }
    }
    bool united(int u,int v)//是否连通
    {
        return find(u)==find(v);
    }
};
```

- 只有按秩合并优化的并查集时间复杂度为O(nlogn)。
- 按秩合并的并查集支持可回退、可持久化、kruskal重构树等操作。

P3366 【模板】最小生成树

题目描述

如题，给出一个无向图，求出最小生成树，如果该图不连通，则输出 `orz`。



输入格式

第一行包含两个整数 N, M，表示该图共有 N 个结点和 M 条无向边。

接下来 M 行每行包含三个整数 X_i, Y_i, Z_i，表示有一条长度为 Z_i 的无向边连接结点 X_i, Y_i。

输出格式

如果该图连通，则输出一个整数表示最小生成树的各边的长度之和。如果该图不连通则输出 `orz`。

输入输出样例

输入 #1 复制

```
4 5
1 2 2
1 3 2
1 4 3
2 3 4
3 4 3
```

输出 #1 复制

```
7
```

```
#include <bits/stdc++.h>
using namespace std;
const int maxn=200005;
int n,m;
int sum=0;
int fat[maxn];
struct node
{
    int u;
    int v;
    int w;
}e[maxn];
bool cmp(struct node x,struct node y)
{
    return x.w<y.w;
}
int father(int x)
{
    //寻找团体的一个头头
    if(fat[x]!=x)
  fat[x]=father(fat[x]);
    return fat[x];
}
  void join(int x,int y)
  {
    fat[father(x)]=father(y);
  }
int main()
{
    ios::sync_with_stdio(false);
    cin>>n>>m;
    for(int i=1;i<=m;i++)
    {

cin>>e[i].u>>e[i].v>>e[i].w;
    }
    sort(e+1,e+1+m,cmp);
    for(int i=1;i<=n;i++)
      fat[i]=i;
    int sum=0,tot=0;
    for(int i=1;i<=m;i++)
    {
        if(tot==n-1)
          break;
        int u=e[i].u;
        int v=e[i].v;
        int w=e[i].w;
        if(father(u)!=father(v))
        {
          join(u,v);
          sum+=w;
          tot++;
        }
    }
    cout<<sum<<endl;
    return 0;
}
```

作业题

- 林大网站上的作业　7道
- 洛谷上的作业　5道

并查集

2020年大一7月13日作业 5题32人

作业：2020年大一7月13日作业

- #A **P3366** 【模板】最小生成树
- #B **P1547** [USACO05MAR]Out of Hay S
- #C **P1195** 口袋的天空
- #D **P1536** 村村通
- #E **P2820** 局域网

2020暑假集训-并查集　System Time : 2020年7月13日 09:13:45

Information | Status | Rank List | 补题提交 | 排行榜

No.	Title	AC/Submit
A	畅通工程并查集版	0/0
B	小希的迷宫	0/0
C	湖南修路	0/0
D	最小树1	0/0
E	修路工程	0/0
F	一道图论一	0/0
G	藤原千花的星星图	0/0

练习

- luoguP3367 【模板】并查集
- Kruskal最小生成树算法
- NOI2018归程 前68pts
- CSP-S 2019 树上的数（难度极大）

进阶操作练习　重点要会带权并差集

- 可回退并查集（loj#121离线可过动态图连通性）
- 可持久化并查集（洛谷模板、NOI2018归程）
- 带权并查集（NOI2001食物链、NOI2002银河英雄传说）矢量法
- Kruskal重构树（IOI2018狼人、NOI2018归程）
- 下面是我的带权并差集的博客
- https://www.luogu.com.cn/blog/xinganheixiong/p2014-si-wu-lian
- https://www.luogu.com.cn/blog/xinganheixiong/p1196-noi2002-yin-he-ying-xiong-zhuan-shuo