


图论

wjh

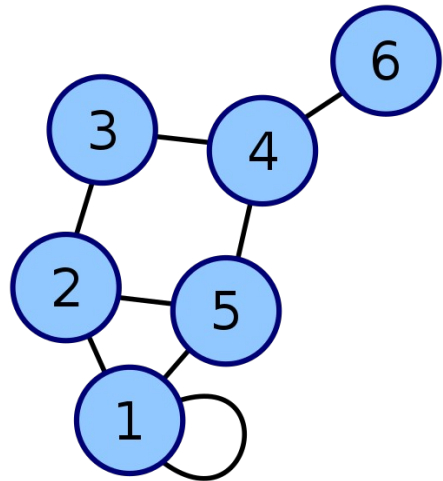




拓扑排序 最小生成树 最短路

图的概念

- 一张图 G 是一个二元组 (V, E) ，其中 V 称为顶点集， E 称为边集
- 边集 E 的元素是二元组数对，用 (x, y) 表示，其中 $x, y \in V$ ，代表有一条从 x 到 y 的边
- 有向图：边有方向， (x, y) 和 (y, x) 不表示一条边
- 无向图：边无方向， (x, y) 和 (y, x) 表示一条边，实际编程中常常认为每一条边都是两条有向边
- 重边：两点之间有多条边，有的题目会保证无重边
- 自环：边 (x, x) ，如右图点 1 处有一个自环，有的题目会保证无自环
- 边权：每条边有一个权值 c ，常常代表距离或者费用
- OI 中不需要拘泥于严格的定义，理解即可



图的存储

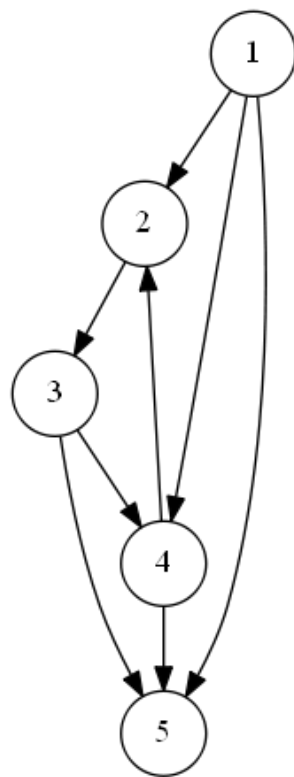
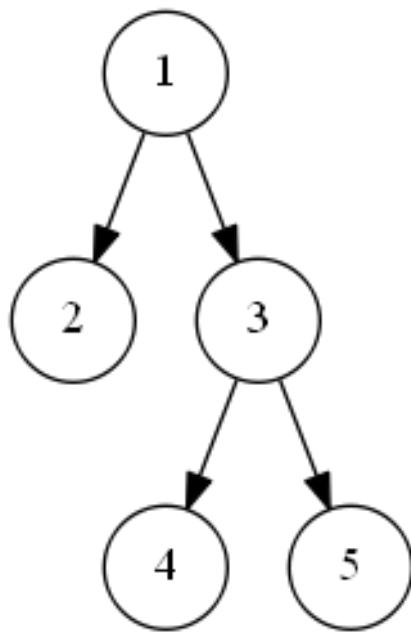
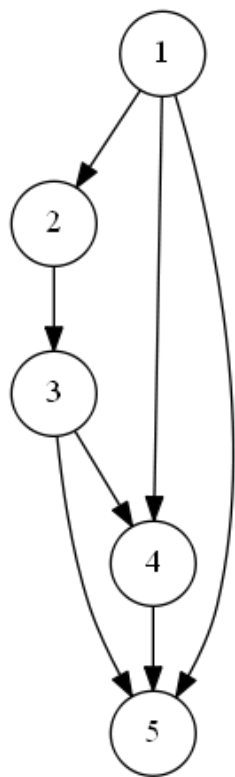
- 邻接矩阵：二维数组 `g[MAXN][MAXN]`，`g[i][j]=1` 代表存在一条 (i,j) 的边
空间复杂度较大，但是有些算法要求这种存储方式
- 邻接表：保存每个点相连的边，具体实现方式有多种。
边多使用结构体定义，`Edge {int from, to, dist;}`
若只需要保存边的终点可以用 `int` 只保存 `to`
 - `vector(1_vector.cpp)`：
`vector<Edge> g[MAXN];`
`g[i]` 是一个 `vector`，保存和点 `i` 相连的边
插入时直接将边 `push_back` 进起点的 `vector` 中，遍历时直接使用 `iterator`，若支持 C++11 还可以使用 `auto` 关键字。
代码很短，常数在 STL 上，开 O2 跑的飞快，不开也没被卡过，强烈推荐
 - 链式前向星 (`2_lfs.cpp`)：
简单的单项链表实现，代码长度还行，跑的快占用空间小，常用，推荐

INF

- 在图论中有时需要表示边权，对于未连通的边，可以将其边权视为无限大
- 在编程时需要将其赋值为一个数据范围外的特殊值
- -1 ：非负权值时通用，但是每次需要做特判，可以 `memset`
- `0x7fffffff`：理论上最大，但是相加会溢出，仍然要特判
- `0x3fffffff`：较大，好处是两个 `INF` 相加不会溢出， $\min(\text{INF}, \text{INF} + \text{INF}) = \text{INF}$
- `0x3f3f3f3f`：较大，相加不会溢出，可以 `memset`，最常使用
- 注意数据范围，图论题常常需要 `long long`，对应的为 `0x3f3f3f3f3f3f3f3fLL`

名词解释

- （点的）度：对于无向图，和某个点相连的边条数
- 入度：对于有向图，终点是该点的边条数
- 出度：对于有向图，起点是该点的边条数
- （两点间）路径：从起点点依次沿着边移动到下一个点，直到终点所经过的点和 / 或边
若为有向图要求只能从边的起点移动到边的终点
- 圈：从一个点出发到自己的路径，常常被称作环
- 有向无环图（ DAG ）：不含有环的有向图

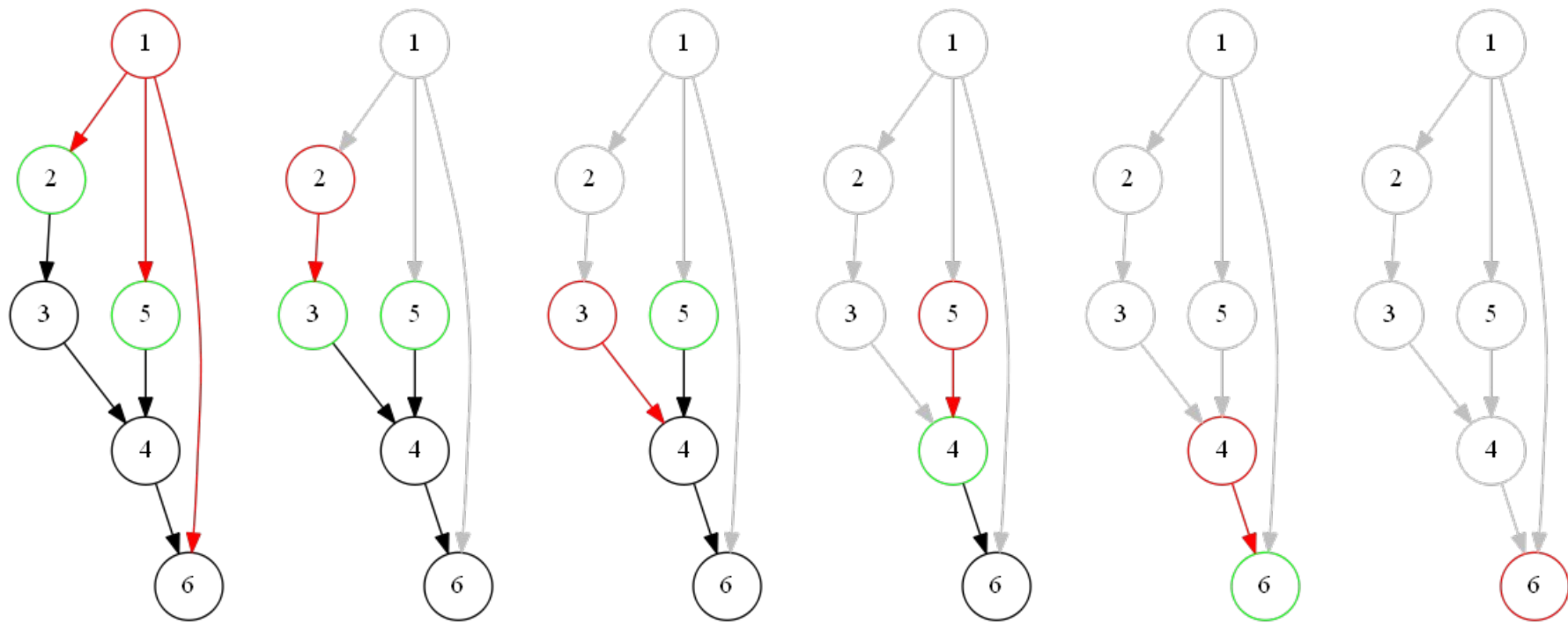


拓扑排序

- 和数组的排序没什么关系
- 对 DAG 的顶点进行排序，结果要求
 - 每个顶点出现且仅出现一次
 - 对于顶点对 (u, v) ，若排序后 u 在 v 前，则不存在 v 到 u 的路径
- 可以理解为，能够到达某个顶点 u 的所有点都在 u 前面出现的一种访问顺序
- 一般是随着排序过程处理节点的信息，不需要显式得出结果

算法流程

- 首先在建图时记录每个点的入度
- 建立一个队列，把接下来需要访问的点加入队列
- 最开始时所有入度为 0 的点都可以访问，加入队列
- 依次从队列中取出每个点 u ，枚举其出边，边的终点设为 v
 - 此处进行各种 $u \rightarrow v$ 的信息更新
- 因为 u 信息已经计算过了，相当于从图中删去 u ，将其 v 入度 -1
- 此时 v 若入度为 0 则说明前置信息处理完成，加入队列



NOIP2015 P2661 信息传递

题目描述

有 n 个同学（编号为 1 到 n ）正在玩一个信息传递的游戏。在游戏里每人都有一个固定的信息传递对象，其中，编号为 i 的同学的信息传递对象是编号为 T_i 的同学。

游戏开始时，每人都只知道自己的生日。之后每一轮中，所有人会同时将自己当前所知的生日信息告诉各自的信息传递对象（注意：可能有人可以从若干人那里获取信息，但是每人只会把信息告诉一个人，即自己的信息传递对象）。当有人从别人口中得知自己的生日时，游戏结束。请问该游戏一共可以进行几轮？

NOIP2015 P2661 信息传递

- 每个点只有一条出边
- 传递到自己的情况是出现环
- 环上的点的出边一定指向环上的下一个点，其它点指向关系是一颗树，树根指向环上的点
- 这种有向图一般被称为基环内向树
- 题目中的关系可能不只有一个环，也就不只有一颗基环内向树
- 先把树的部分全都删掉，剩下环计算长度就可以了
- 删的方式类似拓扑排序

例题： P1983 车站分级

题目描述

一条单向的铁路上，依次有编号为 $1, 2, \dots, n$ 的 n 个火车站。每个火车站都有一个级别，最低为 1 级。现有若干趟车次在这条线路上行驶，每一趟都满足如下要求：如果这趟车次停靠了火车站 x ，则始发站、终点站之间所有级别大于等于火车站 x 的都必须停靠。（注意：起始站和终点站自然也算作事先已知需要停靠的站点）

例如，下表是 5 趟车次的运行情况。其中，前 4 趟车次均满足要求，而第 5 趟车次由于停靠了 3 号火车站（2 级）却未停靠途经的 6 号火车站（亦为 2 级）而不满足要求。

车站编号	1		2		3		4		5		6		7		8		9
车站级别	3		1		2		1		3		2		1		1		3
车次																	
1	始	→	→	→	停	→	→	→	停	→	终						
2					始	→	→	→	停	→	终						
3	始	→	→	→	→	→	→	→	停	→	→	→	→	→	→	→	终
4							始	→	停	→	停	→	停	→	停	→	终
5					始	→	→	→	停	→	→	→	→	→	→	→	终

现有 m 趟车次的运行情况（全部满足要求），试推算这 n 个火车站至少分为几个不同的级别。

模型转化

- 考虑一条火车线路会带来什么信息
- 经过的站点编号大于未经过的站点
- 每个站点建立一个节点，每个“大于”建立一条有向边
- 问题转化为 DAG 上的最长链
- 记录 $l[i]$ 为终点为 i 的最长链， $l[i] = \max\{l[j]\} + 1, (j, i) \in E$
- 类似 DP 的式子，为了保证计算顺序需要按照拓扑序遍历

优化

说明

对于20%的数据, $1 \leq n, m \leq 10$;

对于 50%的数据, $1 \leq n, m \leq 100$;

对于 100%的数据, $1 \leq n, m \leq 1000$ 。

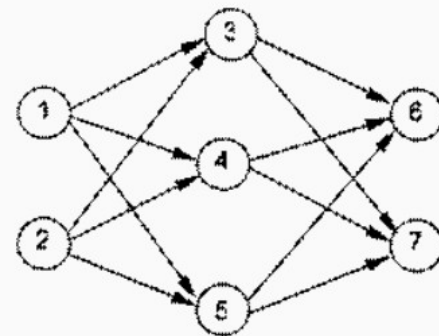
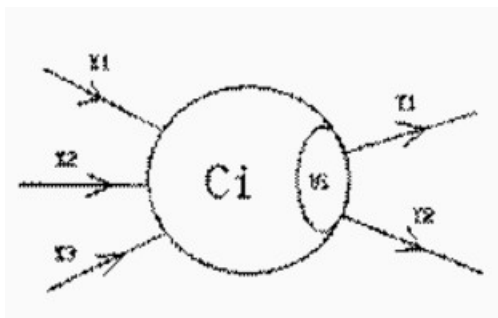
- 连边太慢
- 考虑将每条火车经过的站点中优先级最低的节点作为这条火车的优先级
- 若两条火车运行区间有重叠, 考虑重叠的这部分区间
- 若经过站点不相同, 则一定是某条优先级较高火车经过的站点的包含另一条的站点
- 因此将每条火车建点, 仍然将大于关系建边求最长链

偏序关系

- 从自反性、对称性、传递性三个方面考虑集合的某个关系
- 类似小于等于
 - $\forall a \in S, a \leq a$
 - $\forall a, b \in S, a \leq b$ 且 $b \leq a$, 则 $a=b$
 - $\forall a, b, c \in S, a \leq b$ 且 $b \leq c$, 则 $a \leq c$
- 类似小于
 - $\forall a \in S, a \not\leq a$
 - $\forall a, b \in S, a < b$, 则 $b \not< a$
 - $\forall a, b, c \in S, a < b$ 且 $b < c$, 则 $a < c$
- 这类关系被称为偏序关系, 可以将集合中元素建点, 将偏序关系建有向边转化为图论问题, 然后通过拓扑排序等有向图算法求解。

NOIP2003 P1038 神经网络

- 在兰兰的模型中，神经网络就是一张有向图，图中的节点称为神经元，而且两个神经元之间至多有一条边相连，下图是一个神经元的例子：
- 图中， X_1-X_3 是信息输入渠道， Y_1-Y_2 是信息输出渠道， C_i 表示神经元目前的状态， U_i 是阈值，可视为神经元的内在参数。
- 神经元按一定的顺序排列，构成整个神经网络。在兰兰的模型之中，神经网络中的神经元分为几层；称为输入层、输出层，和若干个中间层。每层神经元只向下一层的神经元输出信息，只从上一层神经元接受信息。下图是一个简单的三层神经网络的例子。



NOIP2003 P1038 神经网络

兰兰规定, C_i 服从公式: (其中 n 是网络中所有神经元的数目)

$$C_i = \sum_{(j,i) \in E} W_{ji} C_j - U_i$$

$$C_i = \sum_{j,i \in E} W_{ji} C_j - U_i$$

公式中的 W_{ji} (可能为负值) 表示连接 j 号神经元和 i 号神经元的边的权值。当 C_i 大于 0 时, 该神经元处于兴奋状态, 否则就处于平静状态。当神经元处于兴奋状态时, 下一秒它会向其他神经元传送信号, 信号的强度为 C_i 。

如此, 在输入层神经元被激发之后, 整个网络系统就在信息传输的推动下进行运作。现在, 给定一个神经网络, 及当前输入层神经元的状态 (C_i), 要求你的程序运算出最后网络输出层的状态。



NOIP2003 P1038 神经网络

- 由于节点是分层的，所以一定无环
- 每个节点的信息取决于前一层的节点，所以需要一层一层递推计算
- 计算顺序由拓扑排序决定
- 注意这种分层的有向图可以使用拓扑排序确定每个节点的层数

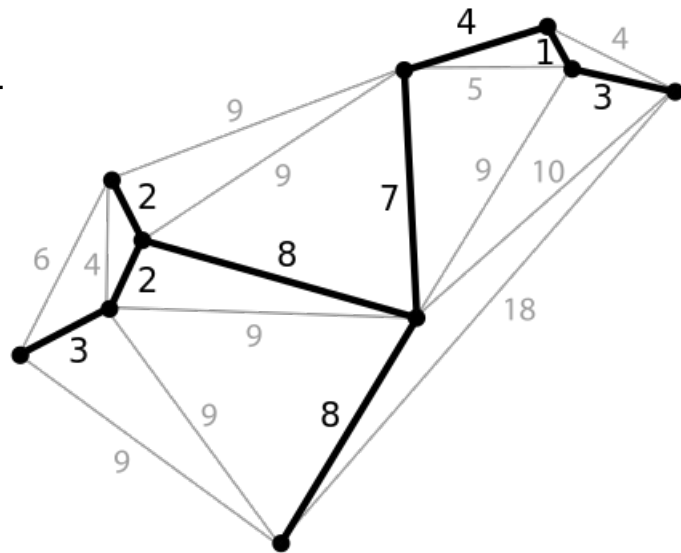


名词解释

- 连通图：任意两点间存在路径
- 树：以下定义等价
 - 任意两点间只存在一条路径的图
 - 无环的连通图
 - n 个点 $n-1$ 条边的连通图
 - n 个点 $n-1$ 条边，无环的图
- 子图：图 G' 称作图 G 的子图如果 $V(G') \subseteq V(G)$ 以及 $E(G') \subseteq E(G)$ 。
也就是从原图中选出一些点以及和一些边组成的新的图
- 生成子图：指满足条件 $V(G')=V(G)$ 的 G 的子图 G' 。
也就是选出所有的点和一些边组成的新的图，生成树则是指子图 G' 是一颗树

名词解释

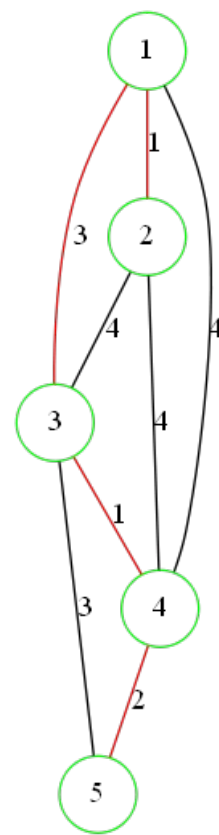
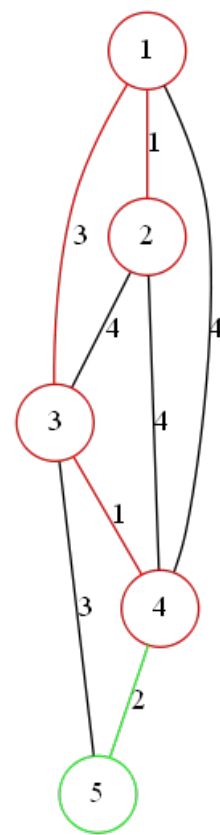
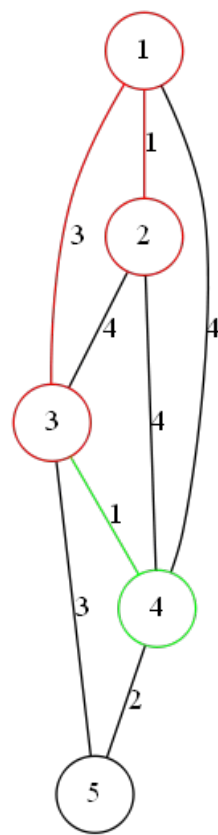
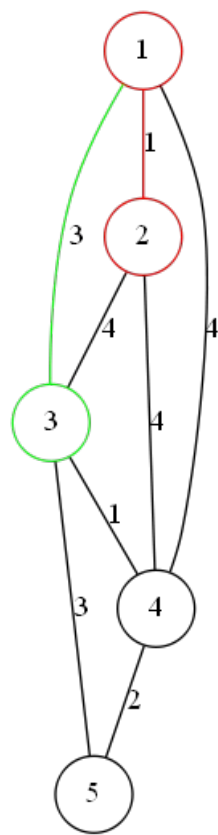
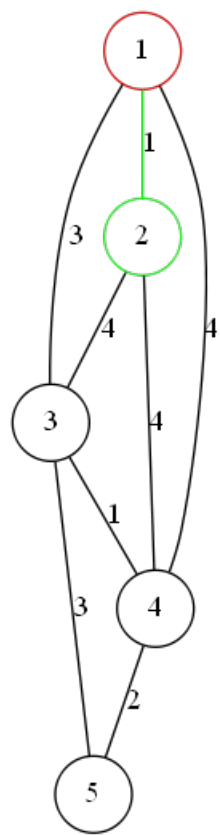
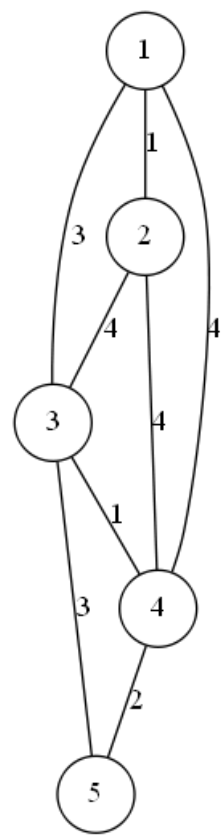
- 最小生成树：对于带权图，权值和最小的生成树
- 最小瓶颈生成树：对于带权图，最大权值最小的生成树
- 最小生成树一定是最小瓶颈生成树
- 可以使用 Prim 算法或者 Kruskal 算法，复杂度均为 $O(m \log n)$
- 推荐使用后者，无需建图





Prim 算法流程

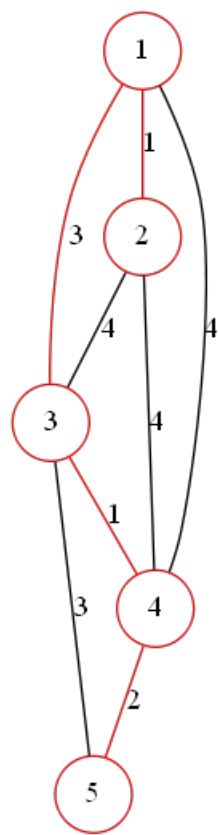
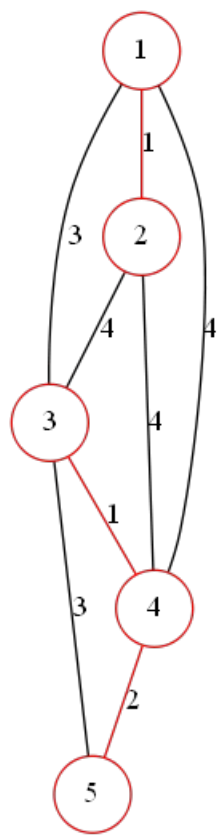
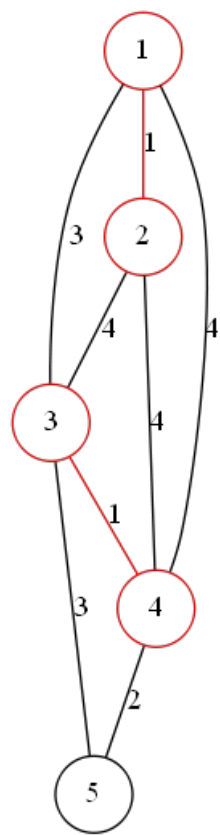
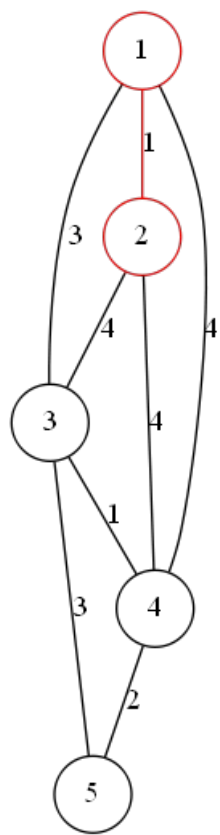
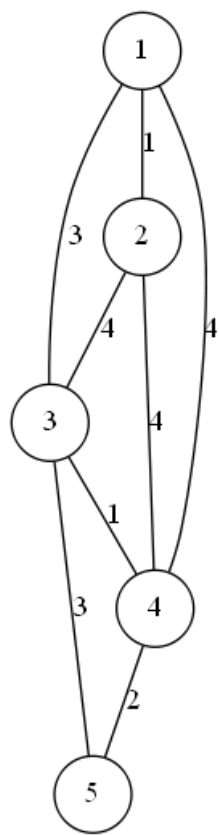
- 随意选取一个点作为已访问集合的第一个点，并将所有相连的边加入堆中
 - 从堆中找到最小的连接集合内和集合外点的边，将边加入最小生成树中
 - 将集合外点标记为已访问，并将相连边加入堆
- 重复以上过程直到所有点都在访问集合中





Kruskal 算法流程

- 将边按照权值排序
- 依次枚举每一条边，若连接的两点不连通则加入最小生成树中
- 使用并查集维护连通性



模板题 P3366

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

输入输出格式

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。（ $N \leq 5000$ ， $M \leq 200000$ ）

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：

输出包含一个数，即最小生成树的各边的长度之和；如果该图不连通则输出orz



题目建模

- 如果只考最小生成树的话模型会比较简单，只要找到题目中哪些元素对应点，哪些费用对应边建图就可以了，可能用到拆点拆边之类的技巧
- 另外也有可能求最小生成树之后再做树上操作，可能后续需要树型 DP 或者 LCA/ 倍增之类的算法。这种题目中求最小生成树是第一步，考虑到图上是否有多余信息就自然会想到求生成树。
- 总的来说最小生成树的题比较裸

例题 P1194 买礼物

题目描述

又到了一年一度的明明生日了，明明想要买 B 样东西，巧的是，这 B 样东西价格都是 A 元。

但是，商店老板说最近有促销活动，也就是：

如果你买了第 I 样东西，再买第 J 样，那么就可以只花 $K_{I,J}$ 元，更巧的是， $K_{I,J}$ 竟然等于 $K_{J,I}$ 。

现在明明想知道，他最少要花多少钱。



问题分析

- 都要买，问题在需要确定一个购买顺序
- 用了某个优惠关系就在两点间连一条边，最后出来是一颗树
- 那么用所有优惠关系建图，最后求最小生成树即可



最短路问题

- 求从 s 到 t 权值和最小的路径
- Floyd 算法：
 - 多源最短路，求出所有点对的最短路长度
 - 时间复杂度： $O(n^3)$
- Dijkstra 算法：
 - 单源最短路，求出某个点 s 到所有点的最短路长度
 - 时间复杂度： $O(n^2)/O(m\log n)$
 - 无法处理负权
- SPFA 算法，即队列优化的 Bellman-Ford 算法：
 - 单源最短路，求出某个点 s 到所有点的最短路长度
 - 时间复杂度：声称 $O(m)$ ，最坏 $O(nm)$ ，容易卡到最坏
 - 可以处理负权边，可以判断负权环

Floyd 算法

- 设 $d[i][j][k]$ 为从 i 到 j ，仅通过编号为 $1-k$ 的中间节点的最短路径距离
- $d[i][j][k] = \min(d[i][j][k-1], d[i][k][k-1] + d[k][j][k-1])$
- 初始值 $d[i][j][0]$ 为两点之间边权值，未连通为 INF
- 从 1 到 n 枚举 k ，然后枚举 (i, j)
- 为了方便可以不开第三维，在原地迭代

单源最短路

- 维护一个 $dis[0..MAXN]$ 数组, $dis[i]$ 代表 s 到 i 的最短路径长度
- $dis[s]=0$, 其他为 INF
- 松弛操作: 通过某条路径更新 $dis[v]$ 的值
 - if ($dis[v] > dis[u] + e.dist$) $dis[v] = dis[u] + e.dist$
 - 尝试使用 s 到 u 的最短路加上边 (u,v) 的长度来更新 s 到 v 的最短路

模板题 P3371

题目描述

如题，给出一个有向图，请输出从某一点出发到所有点的最短路径长度。

输入输出格式

输入格式：

第一行包含三个整数 N 、 M 、 S ，分别表示点的个数、有向边的个数、出发点的编号。

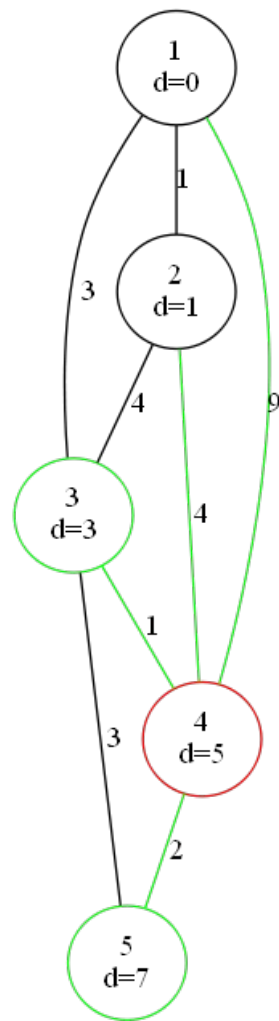
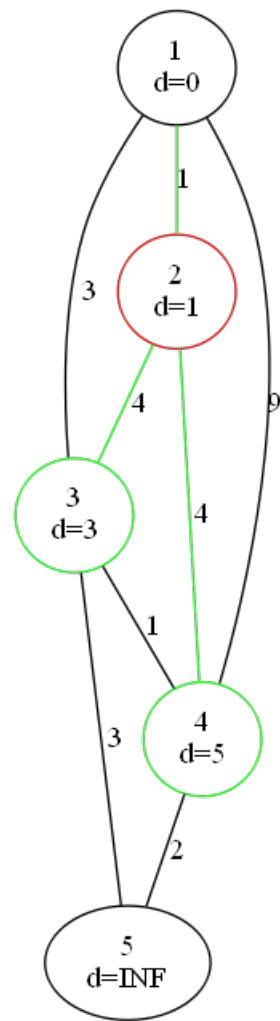
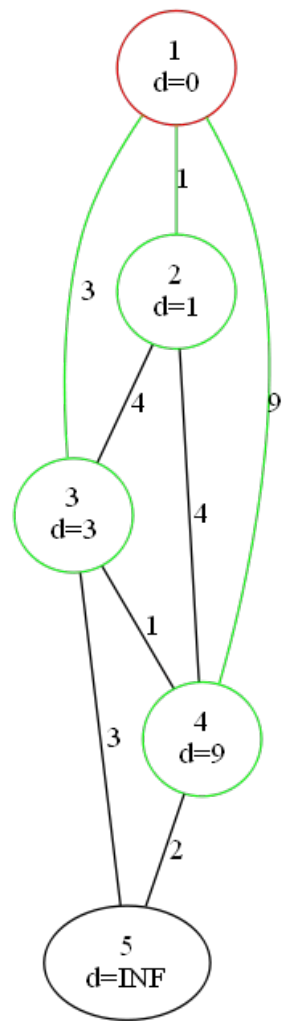
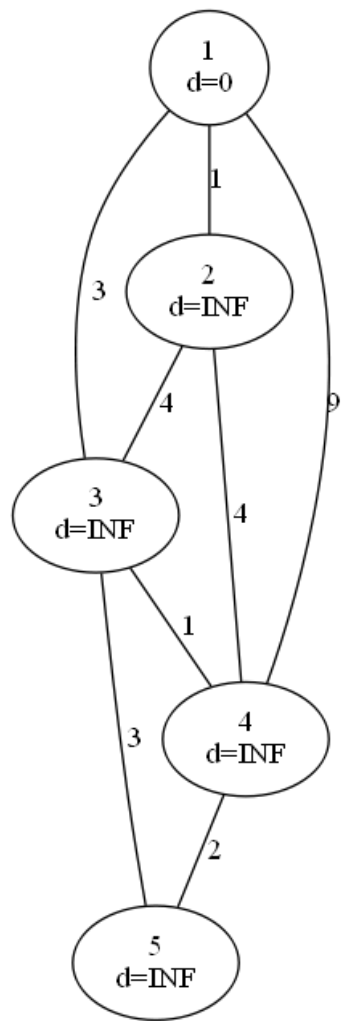
接下来 M 行每行包含三个整数 F_i 、 G_i 、 W_i ，分别表示第 i 条有向边的出发点、目标点和长度。

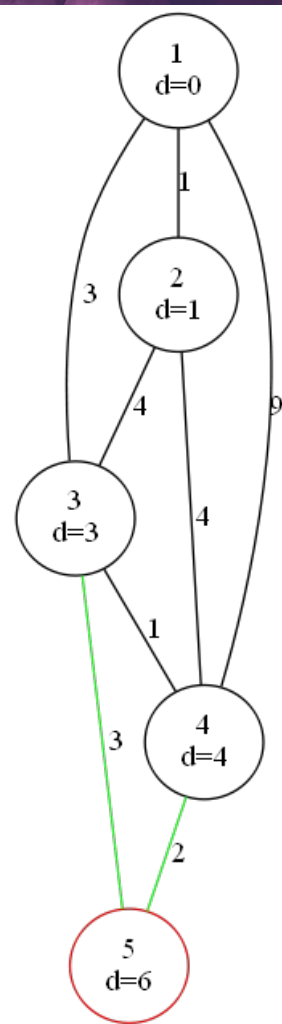
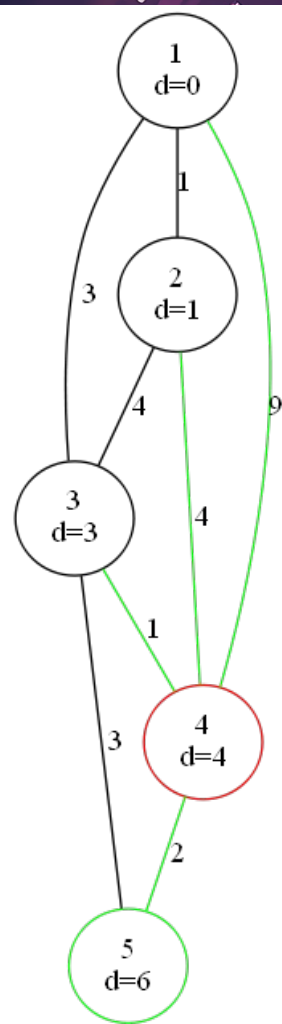
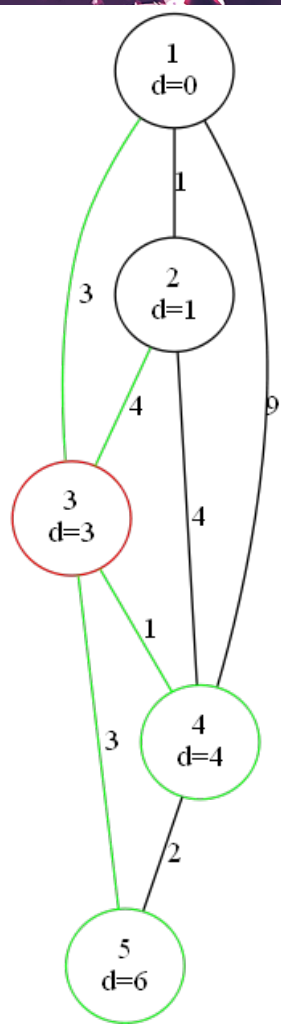
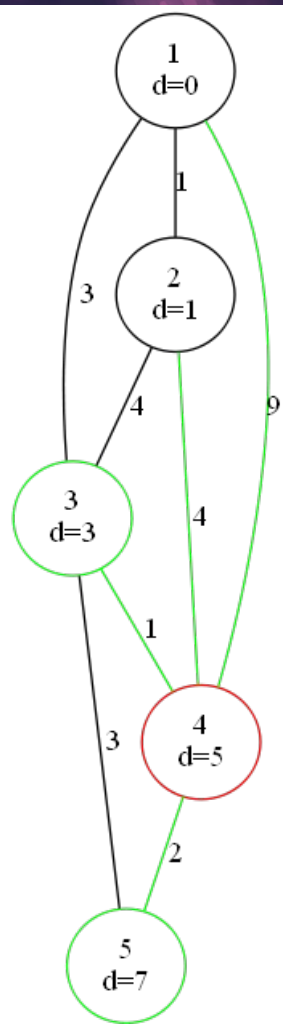
输出格式：

一行，包含 N 个用空格分隔的整数，其中第 i 个整数表示从点 S 出发到点 i 的最短路径长度（若 $S=i$ 则最短路径长度为 0 ，若从点 S 无法到达点 i ，则最短路径长度为 2147483647 ）

SPFA

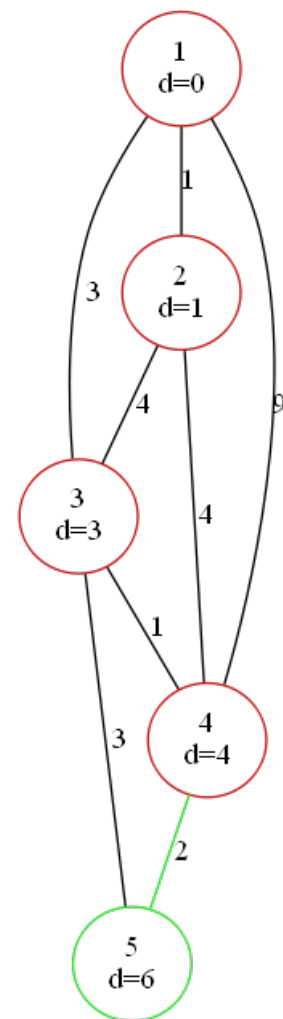
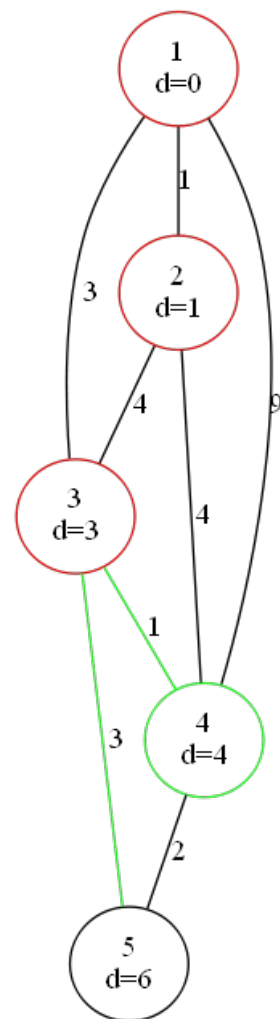
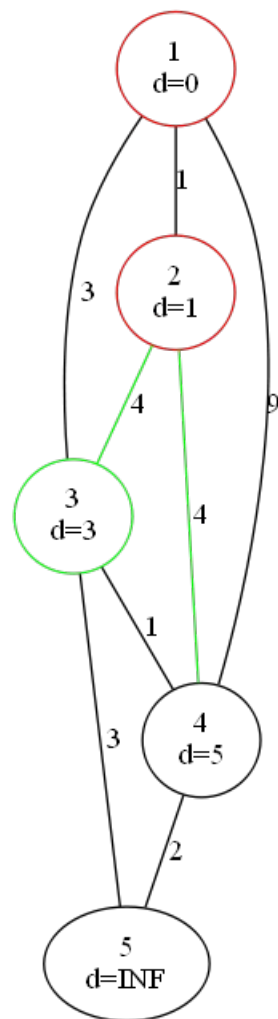
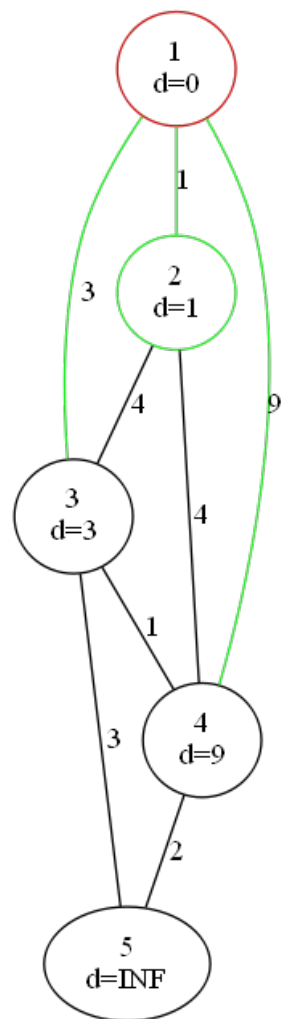
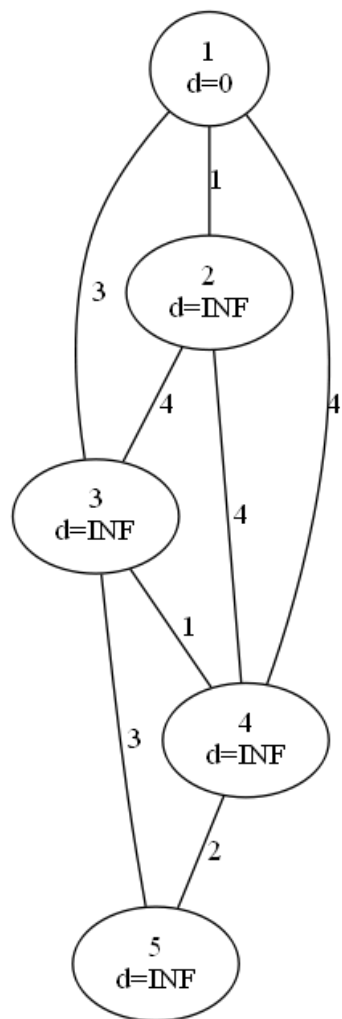
- Bellman-Ford：对整张图进行 $n-1$ 次松弛，每次枚举每条边进行松弛，最后一定能得出最优解
- SPFA：在上述过程中避免无意义的松弛
- 只有成功的松弛操作才会对那个点产生影响，所以使用队列维护等待松弛的点，每次取出一个点进行松弛，对于所有松弛成功的点加入队列
- 判负环：某个点松弛了第 n 次，说明有负环





Dijkstra

- 起点作为已访问集合的第一个点，并更新相连的点的 dis
 - 找到未被访问的 dis 最小的点，标记访问，用这个点更新相连的点 dis
- 重复上述过程直到所有的点均被访问
- 问题在于“找到未被访问的 dis 最小的点”这一步，两种不同的实现方法会带来两种复杂度
- 枚举每个点
- 当点 u 的距离更新时，将 $(dis[u], u)$ 插入堆中，这样堆中可能有多个 u ，此时取出后面的点时，会发现 u 已经被访问过不再处理



例题 P1119 灾后重建

题目背景

B 地区在地震过后，所有村庄都造成了一定的损毁，而这场地震却没对公路造成什么影响。但是在村庄重建好之前，所有与未重建完成的村庄的公路均无法通车。换句话说，只有连接着两个重建完成的村庄的公路才能通车，只能到达重建完成的村庄。

题目描述

给出 B 地区的村庄数 N ，村庄编号从 0 到 $N - 1$ ，和所有 M 条公路的长度，公路是双向的。并给出第 i 个村庄重建完成的时间 t_i ，你可以认为是同时开始重建并在第 t_i 天重建完成，并且在当天即可通车。若 t_i 为 0 则说明地震未对此地区造成损坏，一开始就可以通车。之后有 Q 个询问 (x, y, t) ，对于每个询问你要回答在第 t 天，从村庄 x 到村庄 y 的最短路径长度是多少。如果无法找到从 x 村庄到 y 村庄的路径，经过若干个已重建完成的村庄，或者村庄 x 或村庄 y 在第 t 天仍未重建完成，则需要返回 -1 。

对于100%的数据，有 $N \leq 200$ ， $M \leq N \times (N - 1)/2$ ， $Q \leq 50000$ ，所有输入数据涉及整数均不超过100000。



题目分析

- 询问可离线
- 询问只经过前 t 天的点的多源最短路
- 注意到和 Floyd 的过程一致，将枚举 k 的顺序改为排序后的顺序即可，一边松弛一边处理询问
- 需要离线处理，将询问和点一起按照时间戳排序

例题 [P1629](#) 邮递员送信

题目描述

有一个邮递员要送东西，邮局在节点1.他总共要送 $N-1$ 样东西，其目的地分别是 $2\sim N$ 。由于这个城市的交通比较繁忙，因此所有的道路都是单行的，共有 M 条道路，通过每条道路需要一定的时间。这个邮递员每次只能带一样东西。求送完这 $N-1$ 样东西并且最终回到邮局最少需要多少时间。

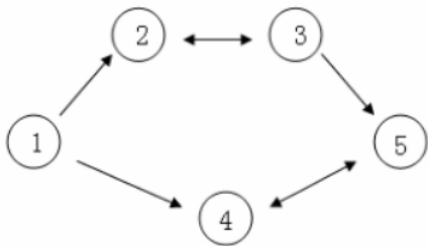


题目分析

- 从起点跑单源最短路可以得到起点到所有点的最短路
- 如何得到所有点到起点的最短路?
- 反向建图从起点跑单源最短路

NOIP2009 P1073 最优贸易

- $n(≤100000)$ 个城市之间用单向边或双向边相连，现在要从 1 号点走到 n 号点，路径可以有回路
- 每个城市有一个水晶球售价，现在要在旅途中买卖一次，求最大的收益



假设 1 n 号城市的水晶球价格分别为 4, 3, 5, 6, 1。

阿龙可以选择如下一条线路：1->2->3->5，并在 2 号城市以 3 的价格买入水晶球，在 3 号城市以 5 的价格卖出水晶球，赚取的旅费数为 2。

阿龙也可以选择如下一条线路 1->4->5->4->5，并在第 1 次到达 5 号城市时以 1 的价格买入水晶球，在第 2 次到达 4 号城市时以 6 的价格卖出水晶球，赚取的旅费数为 5。

思路 1

- 有费用的路径问题可以考虑最短路
- 分为三个阶段，起点到购买点、购买点到售卖点、售卖点到终点
- 使用分层图思想，将原图复制为三份对应三个阶段
- 从第 1 层图到第 2 层图对应购买，对于每个点从第 1 层到第 2 层连边，权值为在这个点购买的费用
- 从第 2 层图到第 3 层图对应售卖，同理连边，权值为负的费用
- 由于移动不需要费用，三层图内部的边权为 0
- 之后求出来的最短路，层内部的对应移动，跨层的对应购买或者售卖操作