最短路

PKU Hzyoi

01 SPFA

三角不等式&松弛操作

- 令s为单源最短路问题的源点。
- 对于任意边(u, v) ∈ E，有dist(s, v) ≤ dist(s, u) + cost(u, v)，称为三角不等式。
- 三角不等式为dist(s,t)是s到t的最短路的充分必要条件。

- 对于每个顶点v ∈ V，用d[v]描述s到v的最短路径上界，称为最短路估计。
- 对于一条边(u, v) ∈ E，若d[u] + cost(u, v) < d[v]，则用d[u] + cost(u, v)更新d[v]的值。
- 这一操作称为松弛操作，原理为三角不等式，本质为：不断寻找当前状态与目标状态之间的矛盾并调整，直到找不到矛盾，即达到最优状态。

bellman-ford

- 用于处理单源最短路问题。
- 每轮暴力队每条边进行一次松弛操作，因为最短路的边数不会超过n，只要做n轮，就能满足三角不等式
- 时间效率：O(NM)。
- 允许有负权边，且可以判断有无负环。

SPFA

- 过程中，我们要维护一个队列，开始时将源顶点置于队首，然后反复进行这样的操作，直到队列为空：
- (1)从队首取出一个顶点u，扫描所有由u顶点可以一步到达的顶点，具体的扫描过程，随存储方式的不同而不同；
- (2)一旦发现有这样一个顶点，记为v，满足$D_v > D_u + L(u,v)$，则将D_v的值减小到$D_u + L(u,v)$。即松弛操作。
- (3)上一步中，我们认为我们“改进了”结点v的最短路径，结点v的当前路径长度D_v相比以前减小了一些，于是，与v相连的一些结点的路径长度可能会相应地减小。注意，是可能，而不是一定。但即使如此，我们仍然要将v加入到队列中等待处理，以保证这些结点的路径值在算法结束时被降至最优。当然，如果连接至v的边较多，算法运行中，结点v的路径长度可能会多次被改进，如果我们因此而将v加入队列多次，后续的工作无疑是冗余的。这样，就需要我们维护一个bool数组Inqueue，来记录每一个结点是否已经在队列中。我们仅需将尚未加入队列的点加入队列。

```
void SPFA(int s){
    memset(dist,0x3f,sizeof dist);
    memset(inq,0,sizeof inq);
    queue<int>Q;
    dist[s]=0;
    while (!Q.empty()){
        int u=Q.front();inq[u]=0;
        for (int i=head[u];i!=-1;i=next[i]){
            int v=to[i];
            if (dist[u]+len[i]<dist[v]){
                dist[v]=dist[u]+len[i];
                if (!inq[v]){Q.push(v);inq[v]=1;}
            }
        }
    }
}
```

- 队列优化无法改变Bellman-Ford算法的复杂度上界。
- 最坏情况下，一个点最多入队n次，因此基于队列优化的Bellman-Ford算法的复杂度上界为O(nm)。
- 网上还有各种玄学优化如SLF，LLL，堆优化SPFA等。
- 这些优化在无负权边的图上加上也不如Dijkstra
- 而若有负权边，前两者写法因人而异不论，所谓堆优化SPFA复杂度会达到指数级
- 网格图可以把SPFA卡到上界O（nm）
- 在nm能过的情况下无所谓
- 但无负权边尽量写Dijkstra

如何判断负环

- 如果没有负环，那么任意两点之间的最短路只会经过至多N – 1条 边。
- 所以当N次松弛后还有更新，那就出现了负环。
- bellman-ford ： 直接判
- SPFA：判断一个点是否入队超过n次
- 常见情况：
  - 判断全局有无负环：直接令所有点初始距离为0。
  - 判断从S出发的所有路径中有无负环：只令这个点初始距离为0。
  - 判断从S到T的所有路径有无负环：将能从S走到且能走到T的点拿 出来，重新构图跑。

02 差分约束

差分约束系统

- 有N个变量x1, x2, .., xN，有M条不等式关系，形如： $x_j - x_i \le w$
- 我们要求解xt − xs的最大值。
- 假设有xt − xs ≤ a ≤ b这两个限制，那xt − xs的最大值必然是a，即较小的那一个。 所以我们可以把每一个变量xi视为图上一个点。
- 对于每一个限制，就让xi向xj连一条长度为w的边。
- 直接从s至t跑最短路，就可以得到xt − xs的最大值。
- 因为有负权边，所以用SPFA搞定。

求解xt − xs的最小值

- 把(−xi)视为一个点，则要求的就是(−xt) − (−xs)的最大值。
- 对于限制xj − xi ≤ w，令(−xj)向(−xi)连一条长度为w的边。
- 跑最短路即可。
- 注意连边的技巧。比如要求Xi=Xj，则连 （Xi,Yj,0）和（Yj,Xi,0）。

HDU 1529 Cashier Employment

- 有一个超市，在２４小时对员工都有一定需求量，表示为rｉ,意思为在ｉ这个时间至少要有rｉ个员工，现在有ｎ个员工来应聘，每一个员工开始工作的时间为ｔｉ∈[0,23]，并持续８小时，问最少需要多少员工才能达到每一个时刻的需求
- 前一天16点后的人统计入下一天

- 记sm[i]表示0…i中加了sm[i]个人
- 则sm[i]-sm[i-8]>=R[i]   i∈[8,23]
- sm[i]+s[23]-s[16+i]>=R[i] i∈[0,8]
- sm[i]-sm[i-1]<=i时刻来的人 sm[0]<= 0时刻来的人
- sm[23]就是答案
- 但其中有式子有3个变量，但都包括sm[23]
- 发现sm[23]越大边权越小，正环越不可能出现
- 考虑二分sm[23]=mid，这样根据差分约束建出来的图若有解答案就能更小

Johnson

- Johnson算法用于解决稀疏图的任意点对最短路问题。
- 可以带负权，可以判断负环。
- 效率为：O(N 2 log N + NM log N)或者O(N 2 log N + NM)。

- 首先试想，如果没有负权边，则我们可以对于每一个点作为起点， 跑一遍dij算法。
- 为了处理负权，我们需要对图进行一些变换。

- 我们给每一个点u一个权值h(u)。
- 对于每一条边(u, v,w)，令它新的权值为w ′ = w + h(u) − h(v)。
  - 这样对于一个环，权值还是不变。
  - 对于一条路径v0, v1, ..., vk，权值会增加h(v0) − h(vk)；因为只和两端点的权值有关，所以最短路依旧不会变。
- 现在的问题是如何求出h(u)，使得w ′都大于等于0；这样我们就可以用dij做了。

- 我们在图上加入一个新的点S，令S向每一个点连一条权为0的有向边。
- 用SPFA算法求出S到每一个点的最短路，记为dis(u)。
- 对于原图上每一条边(u, v,w)，因为三角形不等式，所以有：
- $dis(u) + w \geq dis(v)$ 令$h(u) = dis(u)$，
- 则有： $w + h(u) - h(v) \geq 0$
- 这个h(u)就满足要求了！

K短路

- 首先对于第K优解的问题，常见的算法就是A*。
- 即我们建立堆，对于堆中每一个元素，有两个值：当前值和估价值。
- 为了快速拓展出每一个解，我们需要令每一个估价都表示一条可行的路径。

- 先用dij算法求出每一个点到T的最短距离；并由此建立最短路树。
- 那么每一个点沿着树边到T必然是当前的一条最短路。
- 我们不妨令当前值为这个方案已经走的路径的长度，估价值表示这个方案所在点走到T的最短路。
- 最初的时候，估价就是该点最短路树上到T的距离，即最短路。

- 考虑稍微劣一些的解，必然从当前点沿着树边走，然后走一条非树边，接着继续沿着树边走。
- 对于最短路上每一个点，我们都有d(u)，表示u点到T的距离。
- 那么一条非树边(u, v,w)，我们就令w ′ = w + d(v) − d(u)，表示走这条边后的最短路径长度。
- 暴力的想法，可以对于每一个点u，把它在树上所有的祖先f，连出去的非树边都存下来。 排序后，必然是依次选择拓展。

- 容易发现，我们可以用主席树，或者可持久化堆，维护它祖先的所有出边。每次拓展，就可以轻松维护出新的估价。
- 朴素实现，效率：$O((N + M + K) \log N)$。

循环DP

- 假如将每个DP状态当做图上的一个点，将DP的转移作为点与点之间的边处理。
- 常规DP的图都是拓扑图，但是存在一类奇葩DP的状态转移会成环
- 对于基于贪心的DP来说，往往都可以用最短路的做法跑，当然如果DP状态形成的图是个拓扑图的话不需要这么麻烦。
- 对于基于计数的DP来说，一般边权都不会大于1，否则就会出现结果无限大，所以能用图论解决的都是那种概率DP
- 这种时候就需要算出每个状态转移到自己的贡献，并且去掉这部分贡献之后再转移到其他状态。

[AHOI2014/JSOI2014]骑士游戏

- 有n个怪物，可以消耗k的代价消灭一个怪物或者消耗s的代价将它变成另外一个或多个新的怪物，求消灭怪物1的最小代价

- DP方程
- F[x]=min(k[x],sum(F[y]))   (x->y∈E)
- 转移方程有环
- 初始的dis设为k[x]，全部进队
- 跑SPFA，若当前x点的出边能使y变得更优，更新y并把y入队

02 杂题们

[HAOI2012]Road

- 给你一个n个点、m条边的有向图，问你每条边被多少条不同的最短路经过，答案对10^9+7取模，其中n<=1500，m<=5000。
-

- 注意任意两个点之间的路径都要计算。所以首先，不可避免地，我们枚举路径的初始点 S并进行最短路的求解。
- 对于一条边(x,y)，它 的方案数实际上是S~x的方案数num1[x]乘上 y~T的方案数num2[y]。
- 比如我们现在要计算 num1[x]。注意到num1[x]在更新的时候必须在 最短路上，所以我们可以直接按之前最短路跑的结果升序排序去转移每一个点的num1[x] （其实就是拓扑序）；
- 同理，计算num2的时 候降序处理，而且因为T不固定，每一个点都 可以作为T。（计算某个点时先把它num2++)

[Cerc2012]Farm and factory

- 给出N个点和M条边。
- 要求新增 一个N+1号点并向每一个点连直接相邻的边 （边权自定，可以为小数），使得N+1到1~N个点的边权的和最小。
- 同时必须满足从1或2开始的到N个点的最短路都不变。N,M<=10^5

- 根据题意转化为对于任意的x，
- Dis[n+1][x]>=max{|dis[1][x]-dis[1][n+1]|, |dis[2][x]-dis[2][n+1]}。
- 然后求∑Dis[n+1][x] 的最小值。
- 再转化一下：平面上有N个定点(Ai,Bi)，要指定一个点(dis1,dis2)使得：∑max{|Ai-dis1|,|Bi-dis2|}最小。
- 这是平面的切比雪夫距离。通过转45° 坐 标，可以变成曼哈顿距离。
- 曼哈顿距离中，我们只需指定dis1’为x’ 的中位数，dis2’为y’的中位数即可。

NOIP2017逛公园

- 策策同学特别喜欢逛公园。公园可以看成一张N个点M条边构成的有向图，且没有 自环和重边。其中1号点是公园的入口，N号点是公园的出口，每条边有一个非负权值，代表策策经过这条边所要花的时间。
- 策策每天都会去逛公园，他总是从1号点进去，从N号点出来。
- 策策喜欢新鲜的事物，它不希望有两天逛公园的路线完全一样，同时策策还是一个 特别热爱学习的好孩子，它不希望每天在逛公园这件事上花费太多的时间。如果1号点 到N号点的最短路长为d，那么策策只会喜欢长度不超过d+K的路线。
- 策策同学想知道总共有多少条满足条件的路线，你能帮帮它吗？
- 为避免输出过大，答案对P取模。
- 如果有无穷多条合法的路线，请输出−1
- K 50 N 1e5

- 考虑最短路树
- 往上叠k层
- 边权大于0时转移只会从低层到高层
- 只有出现0边时会有同一层的转移，按拓扑序转移即可
- 特判0环

非常感谢您的观看

THANK YOU FOR YOUR WATCHING