图论(02)-最短路

DijikstraFloyd BellmanFord SPFA 差分约束

陈锋(sukhoeing@qq.com, qq:2533620)

Floyd算法 & 传递闭包

```
for(int k = 0; k < n; k++)
  for(int i = 0; i < n; i++)
    for(int j = 0; j < n; j++)
      d[i][j] = min(d[i][j],  d[i][k] + d[k][j]);
```

DP：

- 图不是线性空间，也未必是DAG，无法线性DP
- 定义D(i,j,k)为i到j之间只允许借道点集[1,k]的最短路径
- 初始$D(i,j,0) = W_{i,j}$
- D(i,j,k) = min(D(i,j,k-1), D(i,k,k-1)+D(k,j,k-1))，不借道k，或者借道 k → j
- 从k=0依次到n-1递推即可，$O(n^3)$

- 传递闭包
  - 有向图只考虑连通性，使用D(i,j) = 1,0表示连通与否
  - $D(i,j,k) = \min(D(i,j,k-1), D(i,k,k-1)+W_{k,j})$ -> D(i,j,k) = D(i,j,k-1) || (D(i,k,k-1)&&D(k,j,k-1))
- 算法特点
  - 常数较小/可以适应负权边/无法解决负权环

例题11-4 电话圈(Calling Circles, ACM/ICPC WF 1996, UVa247)

如果两个人相互打电话(直接或间接)，则说他们在同一个电话圈里。例如，a打给b，b打给c，c打给d，d打给a，则这4个人在同一个圈里；如果e打给f但f不打给e，则不能推出e和f在同一个电话圈里。输入n(n≤25)个人的m次电话，找出所有电话圈。人名只包含字母，不超过25个字符，且不重复。

【分析】

1. 计算出传递闭包
2. 对于每个结点u，查找所有{v|满足u->v并且v->u都连通}

例题 11-5 噪音恐惧症(Audiophobia, UVa10048)

输入一个C个点S条边（C≤100，S≤1000）的无向带权图，边权表示该路径上的噪声值。当噪声值太大时，耳膜可能会受到伤害，所以当你从某点去往另一个点时，总是希望路上经过的最大噪声值最小。输入一些询问，每次询问两个点，输出这两点间最大噪声值最小的路径。例如，在图11-8中，A到G的最大噪声值为80，是所有其他路径中最小的（如ABEG的最大噪声值为90）。

【分析】

1. 求任意两点的距离，故Floyd，但是把加法操作变成max即可
2. D(i,j,k) = min(D(i,j,k-1), max(D(i,k,k-1), D(k,j,k-1))

习题 11-2 奶酪里的老鼠(Say Cheese, ACM/ICPC World Finals 2001, UVa1001)

无限大的奶酪里有n（0<=n<=100）个球形的洞。你的任务是帮助小老鼠A用最短的时间到达小老鼠O所在位置。奶酪里的移动速度为10秒一个单位，但是在洞里可以瞬间移动。洞和洞可以相交。输入n个球的位置和半径，以及A和O的坐标，求最短时间。

【分析】

1. 每个洞看做一个结点
2. 两点之间的边权为球心距离减去两者半径(可能<0，就变成0)*10
3. Floyd就可以，$n \leqslant 100$, $O(n^3)$就行

Dijkstra (Single Source Shortest Path)

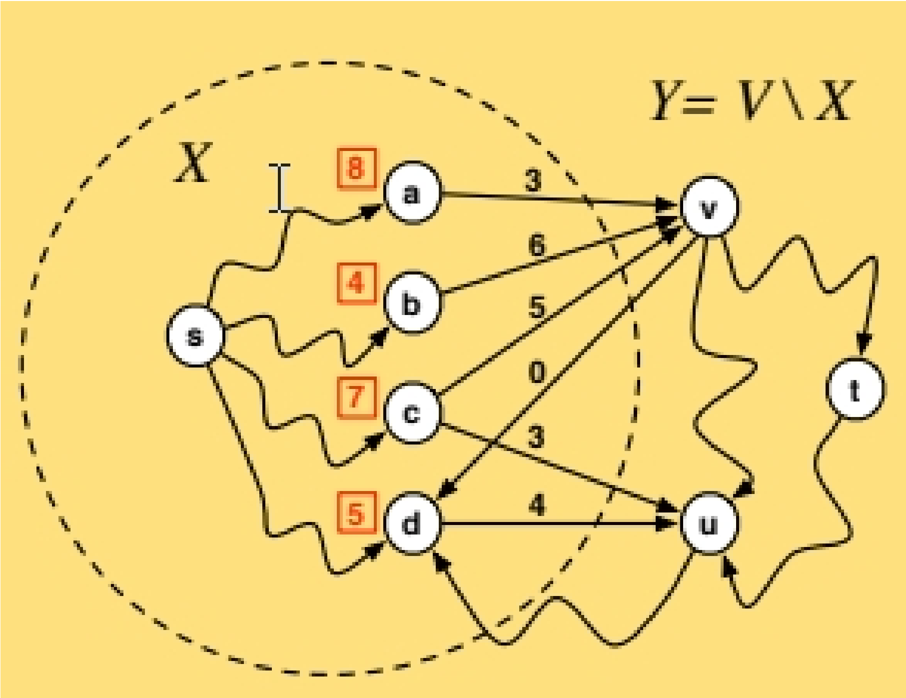
- 算法描述
  - 源点s, d[u]: u到s的最短距离,
  - X为已经计算出d[u]的{u}
  - Pa[u]:最短路上u的父结点
  - 初始: d[u≠s]=∞, d[s] = 0, X={s}

```
while X ≠ V
  set u = argmin(d[u], u ∈ V-S), 将u加入S
  // 这里argmin可以使用优先级队列实现
  for each v (e(u, v) < ∞) // 松弛操作
    d[v] = min(d[v], d[u] + e(u,v)), P[v] = u // 重新入队
```

- 证明，需要证明任意时刻，对于u∈S, d[u]就是最短路径，归纳法即可
- 子问题最优->最终问题最优
- BFS的queue -> priority_queue
- 不支持负权边，**复杂度O(mlogn)**

归纳法证明

1. 假设X集合中的点dist值都已经是正确最短路长度，假设Y中点u的dist值是最小的，也就是说它将被放进X里
2. 如果要维持这个不变式，此时dist(u)必须等于v正确的最短路径长度。
3. 反证法：假设有另外一条路径P s→v→u 更短。这条路径，必须先从s走到frontier里的某个点，再走到u，不然就违背了dist(u)最小的假设。但如 果|s→v→u | < |s→u |, dist(v) < dist(u)，这与dist(u)最小的假设违背。所以dist(u)已经是u的最短路径值。



路径统计

1. 枚举两点之间的所有最短路
   1. 先求出所有点到目标点s的最短路长度d[i]
   2. 起点开始出发行走，但只沿着d[i] = d[j] + w(i,j)的边走
   3. 只有这样走到目标，走出一定是最短路，走错一次都不行。
2. 统计最短路的条数, DP逻辑
   1. f_i表示从i到目标点的最短路的条数
   2. $f_i = \Sigma\{f_j \mid d[i] = d[j] + w(i,j)\}$
   3. 终点t : f(t) = 1

例题11-6 这不是bug，而是特性（It's not a Bug, it's a Feature!, UVa 658）

补丁在修正bug时，有时也会引入新的bug。假定有n(n≤20)个潜在bug和m(m≤100)个补丁，每个补丁用两个长度为n的字符串表示，其中字符串的每个位置表示一个bug。第一个串表示打补丁之前的状态(“-”表示该bug必须不存在，“+”表示必须存在，0表示无所谓)，表示打补丁之后的状态（“-”表示不存在，“+”表示存在，0表示不变）。每个补丁都有一个执行时间，你的任务是用最少的时间把一个所有bug都存在的软件通过打补丁的方式变得没有bug。一个补丁可以打多次。

【分析】

1. bug的$N=2^{20}$种状态，可以认为是N个点，很多状态是不可能达到的，所以没必要预处理所有边。
2. 从S开始Dijkstra即可，每次动态扩展即可。

例题5-11 机场快线(Airport Express, UVa11374)

在Iokh市中，机场快线是市民从市内去机场的首选交通工具。机场快线分为经济线和商业线两种，线路、速度和价钱都不同。你有一张商业线车票，可以坐一站商业线，而其他时候只能乘坐经济线。假设换乘时间忽略不计，你的任务是找一条去机场最快的线路。

【分析】

1. 枚举商业线到底坐那条路u-v
2. 然后需要求s-u, v-t
3. 这是无向图D(v,t) = D(t,v)，可以分别以s,t为源求两次Dijkstra，则s-u,v-t就可以常数时间求出来。
4. mlog(n)+m

例题5-12 林中漫步（A Walk through the Forest, UVa10917)

Jimmy下班需要穿过一个森林。劳累一天后在森林中漫步是件非常惬意的事，所以他打算每天沿着一条不同的路径回家，欣赏不同的风景。但他也不想太晚回家，因此他不打算走“回头路”。换句话说，他只沿着满足如下条件的道路(A,B)走：存在一条从B出发回家的路径，比所有从A出发回家的路径都短。你的任务是计算一共有多少条不同的回家路径。

【分析】

1. 实际上是要求回家走路时d(B)<d(A), dijkstra
2. 构造一个有向图，当且仅当d(B)<d(A)时加入有向边A->B
3. 这个图必然是DAG
4. DP求出从起点出发回家的最短路个数即可

例题5-13 战争和物流(Warfare And Logistics, LA4080)

给出一个n个结点m条边($1<n\leq 100$, $1<=m\leq 1\ 000$)的无向图，每条边上有一个正权。令c等于每对结点的最短路长度之和。

n = 3时: c=d(1,1)+d(1,2) +d(1,3)+d(2,1)+d(2,2)+d(2,3)+d(3,1)+d(3,2)+d(3,3)

要求删除一条边后使得新的c值c'最大。不连通的两点的最短路长度视为L。

【分析】

1. 尝试每删除一条边，floyd暴力计算，重新计算，复杂度$O(mn^3)$
2. 遍历点u，对于指定的u遍历删除的边，每次使用dijkstra，为$O(m^2 n\log n)$，但是floyd常数小，也差不多。
3. 对于每个点u，dijkstra之后会形成一个最短路树(DT)，只要DT的边不被删除，c值不变，复杂度为$O(n^2)$
4. 最短路树。用Dijkstra算法可以求出单源最短路树，方法是在发现$d[i] + w(i,j) < d[j]$时除了更新$d[j]$之外还要设置$p[i] = j$。这样，把p看成是父指针，则所有点形成了一棵树。这样，要从起点出发沿最短路走到任意其他点，只需要顺着树上的边走即可。因为连通且有$n - 1$条边, 注意起点的p值等于它自己，其他每个点u对应一条边$p[u]\rightarrow u$。
5. 如果删了在DT中的的边，需要重新计算，复杂度降低为$O(n^2 m\log n)$
6. 陷阱：注意重边，删除第1条边，用第2条边代替

习题11-7 电梯换乘(Lift Hopping, UVa10801)

在一个假想的大楼里，有编号为0~99的100层楼，还有n（n<=5）座电梯。你的任务是从第0楼到达第k楼。每个电梯都有一个运行速度，表示到达一个相邻楼层需要的时间(单位：秒）。由于每个电梯不一定每层都停靠，有时候需要从一个电梯换到另一个电梯。换电梯时间总是1分钟，但前提是两座电梯都能停靠在换乘楼层。大楼里没有其他人和你抢电梯，但你不能使用楼梯(这是一个假想的大楼，你无需关心它是否真实存在)。

【分析】

1. 一层楼内，每个电梯在该层的出口看作一个顶点
2. 假如电梯 x在第 i 层 停靠，把 x 在 i 层的出口看作一个点：x*100 + i。
3. 对于 x 的下一个停靠层 j ，x*100 + j →x*100+i 连一条边，权值是电梯从 i 到 j 层运行时间。
4. 同样在 i层停靠的每对电梯 x,y ，x*100+i → y*100 + i 连一条边，权值为60(同层换乘)。
5. 遍历所有的u=x*100和v=y*100+k，dijkstra 求出min(d(u,v))即可

例题18 低价空中旅行(Low Cost Air Travel, WF2006，LA 3561)

- 票1：1→3→4，票价225
- 票2：1→2，票价200
- 票3：2→3，票价50

很多航空公司都会出售一种联票，要求从头坐，上飞机时上缴机票，可以在中途任何一站下飞机。比如，假设你有一张“1→2→3”的联票，你不能用来只从2飞到3（因为必须从头坐），也不能先从城市1飞到城市2，再用其他票飞到其他城市玩，回到城市2后再用原来的机票飞到城市3（因为机票已经上缴）。

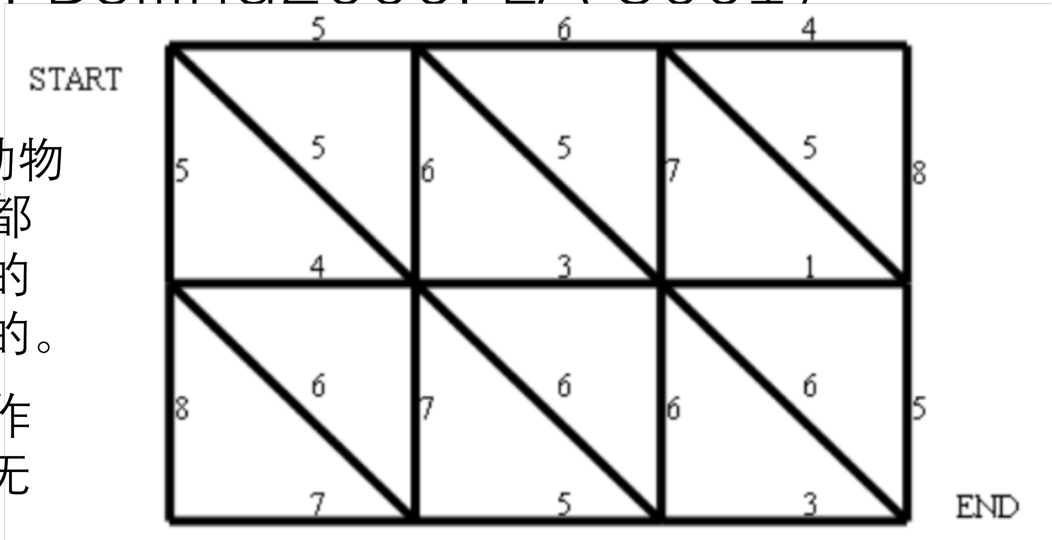
这里有一个例子。假设有3种票，每种票的情况如右所示。

你想从1飞到3，有两种方法可以选择。买票1，只飞第一段；买票2和3，通过城市2中转。显然，第一种方法比较省钱，虽然浪费了一段。给出票的信息，以及一个或多个行程单，你的任务是买尽量少的票（同一种票可以买多张），使得总花费最小。输入保证行程总是可行的。行程单上的城市必须按顺序到达，但中间可以经过一些辅助城市。

【分析】

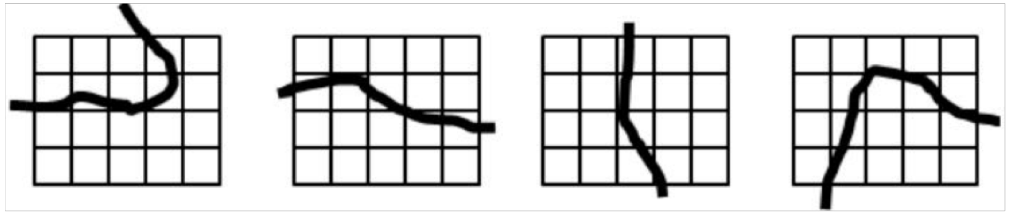
1. “目前已经经过了行程单上的几个城市”i（不超过10）作为状态的一部分，再加上当前城市编号j（最多可能有400个城市），
2. 以每个(i,j)为结点构图。
3. 考虑每张机票的每种用法(飞前面几段)，计算出新的状态。

例题5-19 动物园大逃亡(Animal Run, Beijing2006, LA 3661)



由于控制程序出了bug，动物园的笼子无缘无故被打开，所有动物展开了一次大逃亡。整个城市是一个网格，另外每个单位方格都有一条从左上到右下的对角线，其中动物园在左上角，动物们的目的地是右下角。所有道路（即网格的边和对角线）都是双向的。

每条道路上都有一个正整数权值，代表拦截这条边所需要的工作人员数，如图所示。你的任务是派尽量少的工作人员，使动物无法从动物园走到目的地（动物只能经过没有被拦截的边）。



【分析】

1. 可以看做网络流中的最小割问题，但数据规模太大了。
2. 左上角无法走到右下角，等价于障碍物从“左/下”连到“右/上”。
3. 分别建两个虚拟节点，对应左/下，右/上，Dijkstra即可。

Bellman-Ford

- 现实意义: 贸易利润模型，权值就是投入or产出费用
- 如果最短路存在，一定存在一个不含环的最短路。
  - 正圈和零圈可以去掉，负圈会直接导致最短路不存在。
- 源点s，定义d(i, k)为s->i的最多经过k条边的最短路长度，则算法逻辑如下：
  - $d(i, 1) = w_{s,i}$
  - 松弛n-1次：for each e(u,v,w) : d(v,k)=min(d(v,k-1), d(u,k-1) + w)
    - 决策逻辑：要不要用边e呢?
    - 最短路最多只经过n-1个顶点，所以只用松弛n-1次
  - 复杂度O(nm)

Bellman-Ford的队列优化: SPFA

SPFA : shortest past fast algorithm

1. 松弛操作必定只会发生在SP前导节点relax成功过的节点上
2. 用一个队列queue记录relax过的节点
3. 避免了冗余计算
4. 复杂度可以到O(mn)，证明略
5. 如何判断负圈，松弛>n-1次说明还能松弛
6. 采用队列实现，当某个结点入队了n次时可以判断出存在负圈

例题15 在环中(Going in Cycle!!, UVa11090)

给定一个n个点m条边的加权有向图，求平均权值最小的回路。

【分析】

1. 对平均权值W进行二分，看看有没有平均权值 $<W$ 的回路
2. 若存在这样的回路，回路上边权为 $(w_1, w_2 \cdots w_k)$，平均权值 $<W$ 意味着 $\Sigma w_i < W*k$，也就是说 $\Sigma(w_i - W) < 0$
3. 只要把每条边权值减W，然后用SPFA判断负权圈即可

例题16 Halum操作(Halum, UVa11478)

n个点和m条边(n≤500, m≤2700)的带权有向图, 每条边都有一个权值d(-10000≤d≤10000)。每次可以选择一个结点v和一个整数d, 把所有以v为终点的边的权值减小d, 把所有以v为起点的边的权值增加d, 最后要让所有边权的最小值非负且尽量大。

对于每组数据, 输出边权最小值的最大值。 如果无法让所有边权都非负, 输出"No Solution "; 如果边权最小值可以任意大, 输出"Infinite"。

【分析】

1. 不同的操作互不影响，顺序无关。同一个结点的多次操作也可以合并。
2. S(u)为作用于点u之上的d之和。目标转化为确定所有的S(u), 使得操作之后所有边权最小值尽量大。
3. "最小值最大" →二分答案x, 问题转化为是否可以让操作完毕后每条边的权值均≥x。
4. 边a→b操作完毕后权值为w(a,b)+S(a)-S(b)，每条边a→b都可以列出一个不等式w(a,b) + S(a) – S(b)≥x,移项得S(b) – S(a)≤w(a,b) – x。得到一个差分约束系统(system of difference constraints/SDC)。
5. 如果SDC无解，原问题无解。记原图最大边权为maxb，如果x=maxb+1时有解，说明可以让所有边权增加，重复多次操作，一直到+∞。

差分约束系统

1. 指一个k个不等式的不等式组,每个形如x_j-x_i≤b_k, 这里的b_k是一些事先已知的常数。
2. 类似于最短路中的不等式d(v)≤d(u) + w(u,v)。
3. 对于x_u-x_v≤b_k,新建边u→v, 权值为b_k, 对于x_u, x_v有。d(j)≤d(i) + w(i,j) → x_u≤x_v +b_k→x_u-x_v≤b_k
4. 运行Bellman-Ford, 则源点s→u的最短路就是x_u
5. 负权圈则表示最短路无限小，那么x_u-x_v≤b_k中的b_k无限小， x_u-x_v的最大值不存在
6. 不连通：s无法到达u， x_s-x_u之间有约束关系，这种x_u-x_v的最大值是∞，取值无穷多种

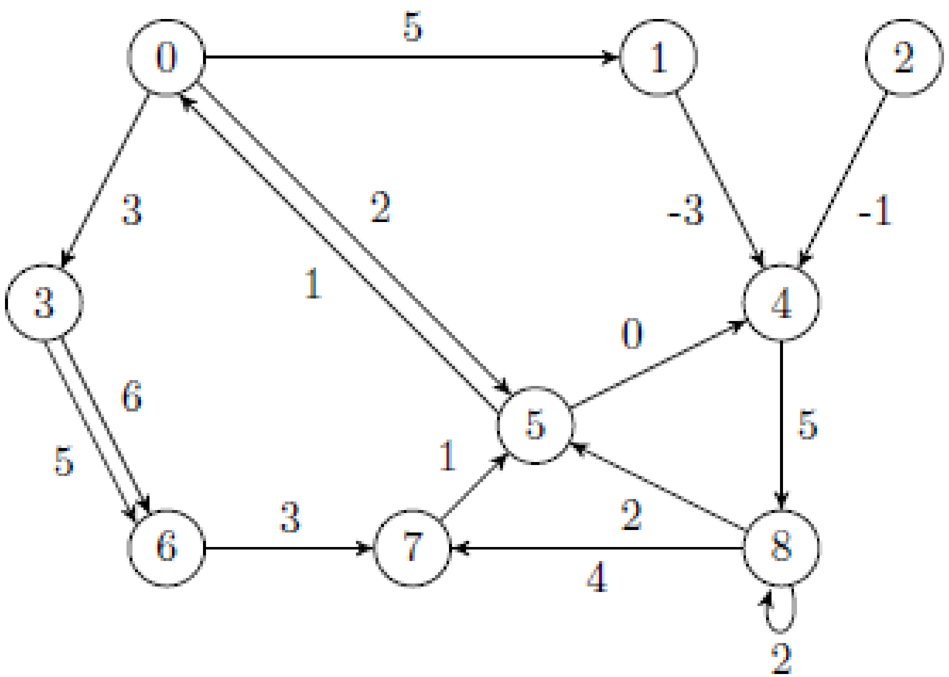
x0-x5≤1, x1-x0≤5, x3-x0≤3,

x4-x1≤-3, x4-x2≤-1, x4-x5≤0,

x5-x0≤2, x5-x7≤1, x5-x8≤2,

x6-x3≤5, x6-x3≤6, x7-x6≤3,

x7-x8≤4, x8-x4≤5, x8-x2≤2



作业

习题11-12 岛屿(Islands, ACM/ICPC CERC2009, UVa1665)

习题11-14 乱糟糟的网络(Network Mess, ACM/ICPC Tokyo 2005, UVa1667)

例题17 蒸汽式压路机（Steam Roller, LA4128）

例题14 过路费(加强版)(Toll! Revisited, UVa10537)

以下难题随意:

例题11-11 UVa12661 Funny Car Racing 特殊图的Dijkstra算法

*例题11-14 UVa1279 Asteroid Rangers 动点的最小生成树