NOIP知识点串讲 图论（一）

Colin

NOIP中涉及的数学知识

图论

组合数学

…

道路与回路

基本概念
图的代数表示
道路矩阵与Warshall算法
BFS、DFS
欧拉回路
哈密顿回路
旅行商问题
最短路
*差分约束
关键路径

树

树的等价定义
DFS序
最近公共祖先
哈夫曼树
最小生成树
生成树计数
prufer序列
*斯坦那树

NOIP知识点串讲 **图论（一）**

一些概念

一些概念

道路（链）、回路

有向道路（回路）

无向道路（回路）

简单道路（回路）

初级道路（回路） 和简单道路的关系？

简单图

连通图

连通支

二分图

DAG（有向无环图）

二分图

若图G是二分图，则G 中的回路都是偶回路

其实是个充要条件：当且仅当图G是二分图时，G 中的回路都是偶回路

二分图具备很多特殊的性质

故我们常要对原图进行二分图染色

DAG（有向无环图）

工程各步骤先后顺序，无矛盾

与树的关系

与有向树的关系

可以在线性复杂度内拓扑排序

通常可以按照拓扑序进行DP

NOIP知识点串讲 图论（一）

图的代数表示

图的代数表示

邻接矩阵

权矩阵

关联矩阵

边列表

邻接表

什么时候用什么表示

NOIP知识点串讲 图论（一）

连通性判断

例1.

已知一个n个点的简单图G的邻接矩阵为A，图G的边权均为1。求所有点对间长度为l的路径条数。

$n \le 100$，$l \le 10^9$

如何计算点对之间的路径条数

设A是图G的邻接矩阵，则A^l可以表示点对之间长为l的路径条数。

两个顶点 v_i, v_j 之间存在道路，最简单的情形是这两点有边相连，即 $a_{ij} = 1$。

另一种情形是 v_i 可以经过某个顶点 v_k 到达 v_j，也就是 $a_{ik} = a_{kj} = 1$，我们发现这等价于 $a_{ik}a_{kj} = 1$。由于 k 是任意的，我们可以对所有情况求和，因此 v_i, v_j 有长为 2 的道路等价于 $\sum_{k=1}^{n} a_{ik}a_{kj} > 0$。设 $A^2 = (a_{ij}^{(2)})$，那么 v_i, v_j 有长为 2 的道路等价于 $a_{ij}^{(2)} > 0$。

类似地 v_i 到 v_j 的一条道路序列 $(v_i, v_{k_1}, v_{k_2}, ..., v_{k_l}, v_j)$ 存在当且仅当 $a_{ik_1}a_{k_1k_2}a_{k_2k_3}...a_{k_lj} = 1$。定义 $A^l = (a_{ij}^{(l)})$，那么 v_i, v_j 有长为 l 的道路等价于 $a_{ij}^{(l)} > 0$，并且 a_{ij} 实际上就是长为 l 的道路条数。

例1.解

求A^l

矩阵乘法具有结合律，可以快速幂

复杂度O（n^3logl）

例2.

已知一个n个点的简单图G的邻接矩阵为A，求所有联通的点对。

$n \leq 500$

只想知道点对之间是否连通？

道路矩阵P

$P = \sum_{k=1}^{n-1} A^k$ （为什么只要加到$n-1$）

复杂度为$O(n^4)$

逻辑运算 $a_{ij}^{(l)} = \bigvee_{k=1}^{n} (a_{ik}^{(l-1)} \bigwedge a_{kj})$

相应的 $P = A \bigvee A^{(2)} \bigvee \ldots \bigvee A^{(n-1)}$

复杂度仍为$O(n^4)$

WARSHALL算法

Algorithm 1 Warshall 算法

矩阵 $P \leftarrow A$
for all $i = 1 \rightarrow n$ **do**
    for all $j = 1 \rightarrow n$ **do**
        for all $k = 1 \rightarrow n$ **do**
            $p_{jk} \leftarrow p_{jk} \vee (p_{ji} \wedge p_{ik})$
        end for
    end for
end for

证明:

对 i 进行归纳:

当 $i=1$ 时，我们注意到

$$p_{jk}^{(1)} = p_{jk} \vee (p_{j1} \wedge p_{1k}) \tag{6}$$

如果 $p_{jk}^{(1)}=1$，那么 $p_{jk}=1$ 或者 $p_{j1}=1 \wedge p_{1k}=1$。其中 $p_{jk}=1$ 表明 j,k 之间有边相连，后者表明 j,k 间存在通过 1 为中转的道路。因此，$p_{jk}^{(1)}=1$ 等价于结点集合 $\{v_j,v_1,v_k\}$ 之间有 v_j 到达 v_k 的道路。

归纳假设 $i=t-1$ 时，$p_{jk}^{(i)}=1$ 当且仅当结点集合 $\{v_j,v_k,v_1,v_2,...,v_{t-1}\}$ 之间存在 v_j 到达 v_k 的道路。

对于 $i=t$ 的情形，由于

$$p_{jk}^{(t)} = p_{jk}^{(t-1)} \vee (p_{jt}^{(t-1)} \wedge p_{tk}^{(t-1)}) \tag{7}$$

如果 $p_{jk}^{(t)}=1$，那么 $p_{jk}^{(t-1)}=1$ 或者 $p_{jt}^{(t-1)}=1 \wedge p_{tk}^{(t-1)}=1$，其中 $p_{jk}^{(t-1)}=1$ 等价于 $\{v_j,v_1,v_2,...,v_t,v_k\}$ 中存在不经过 v_t 的从 v_j 到 v_k 的道路，后者等价于 $\{v_j,v_1,v_2,...,v_t,v_k\}$ 中存在经过 v_t 的从 v_j 到 v_k 的道路。故 $p_{jk}^{(i)}=1$ 当且仅当结点集合 $\{v_j,v_k,v_1,v_2,...,v_t\}$ 中存在 v_j 到达 v_k 的道路。

□

WARSHALL算法

正确性（动态规划思想）

类似于求最短路的Floyd算法

复杂度$O(n^3)$

BFS & DFS

常用的对图进行遍历的方法

广度（宽度）优先搜索算法

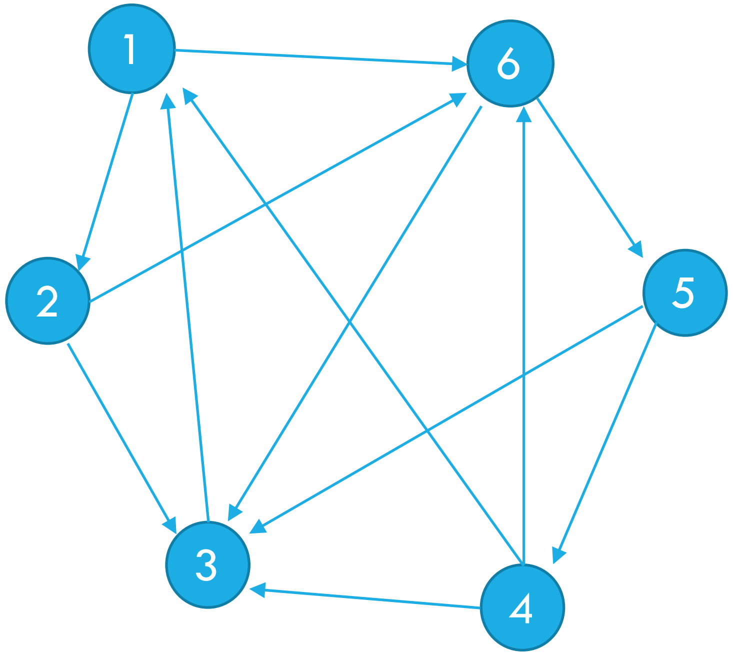
深度优先搜索算法

复杂度均为O(m)

区别在于访问顶点的顺序不同

常用部分分算法

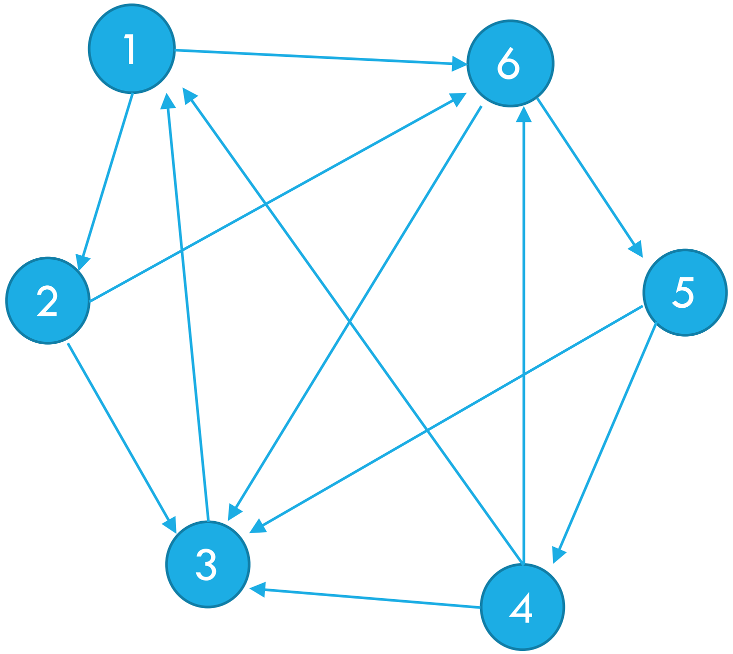
BFS



Algorithm 2 BFS

for all $u \in V(G)$ **do**
    $vis[u] \leftarrow false$
end for
任选一个节点 v_0 加入队列尾端
$vis[v_0] \leftarrow true$
while 队列不为空 **do**
    弹出队列首端元素 u
    for all $(u, v) \in E(G)$ **do**
        if $vis[v] = false$ **then**
            将 v 加入队列尾端
            $vis[v] \leftarrow true$
        end if
    end for
end while

DFS



Algorithm 3 DFS

for all $u \in V(G)$ **do**
    $vis[u] \leftarrow false$
end for
任选一个节点 v_0 加入队列首端
$vis[v_0] \leftarrow true$
while 队列不为空 **do**
    弹出队列首端元素 u
    for all $(u, v) \in E(G)$ **do**
        if $vis[v] = false$ **then**
            将 v 加入队列首端
            $vis[v] \leftarrow true$
        end if
    end for
end while

NOIP知识点串讲 图论（一）

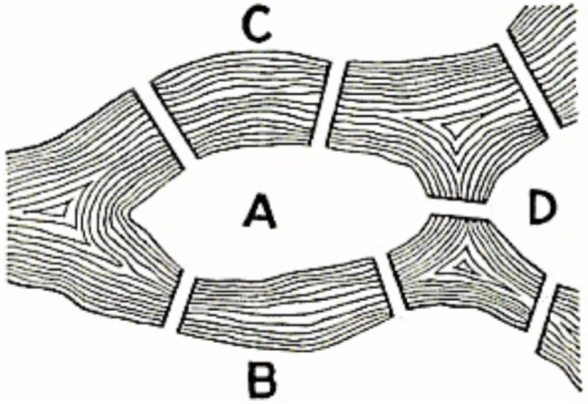
欧拉回路

欧拉回路与道路

七桥问题

一笔画问题

欧拉回路（道路）定义



欧拉回路的判定

无向连通图G 存在欧拉回路C 的充要条件是G 中每个结点的度数均为偶数。

证明：

必要性： 简单而言，有进必有出。对任意一个点 u，C 经由 e 进入 u，必定通过另一条边 f 离开。我们把这样的边配对，那么一定会有结点 u 的度数为偶数。

充分性：（构造法）我们仍选一个点出发，每次沿未访问过的边前往下一个与之相邻的结点。因为每个结点度为偶数，故最后一定会回到最开始的节点，即最后一定会连成一个回路 C_1。

如果这个回路中包含原图所有边，那么这个回路就是我们要求的欧拉回路。

如果不是，那么由于原图是连通的，C_1 和原图的其它部分（即 $G-C_1$）必然有公共顶点。由于在 $G-C_1$ 中每个节点度数任然是偶数，故从这个公共顶点出发，在原图的剩余部分中将得到另一回路。两个回路相连得到一个更长的回路，经过若干步后我们一定能将原图所有边包含进来，此时我们得到原图的欧拉回路。 □

一些推论

有向连通图G 中存在欧拉回路的充要条件是每个结点的正度等于负度。

若无向连通图G 有且仅有两个结点的度为奇数，则G中存在欧拉道路。

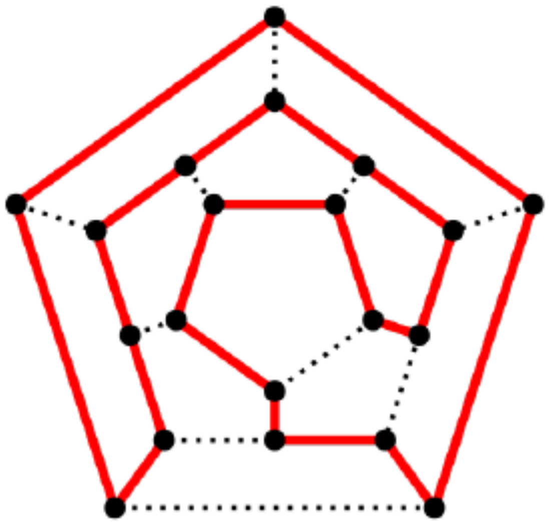
设连通图G 中有k 个度为奇数的顶点，那么G 可以被划分为$k/2$条简单道路。

NOIP知识点串讲 图论（一）

哈密顿回路

哈密顿回路与道路

与欧拉回路（道路）问题非常类似的是哈密顿回路（道路）问题。该问题起源于英国数学家威廉·哈密顿于*1857* 年提出的一个关于正十二面体的有数学游戏：他把*12* 面体的*20* 个顶点比作世界上的*20* 个城市，*30* 条棱比作表示这些城市之间的交通路线。哈密顿提出能否周游世界，即从某个城市出发，经过每个城市一次且一次最后返回出发地。



哈密顿回路（道路）

无向图G 的一条经过全部节点的初级道路（回路）称为G 的哈密顿道路（回路），简记为H 道路（回路）

哈密顿回路是初级回路，欧拉回路是简单回路，在特殊情况下，图G的一条哈密顿回路也恰好是欧拉回路。

因为对于H 回路研究的是初级回路，对于一般图G 删掉其重边和自环，不会影响H 回路的存在性，因此我们一般针对简单图研究H 回路存在性问题。

H回路存在的一个必要条件

如果简单图G 中存在H 回路，从G 中删去若干个点v_{i1}、$v_{i2} \dots v_{ik}$及与他们相邻的边后得到图G'，则图G'的连通支个数不超过k。

证明：设G 的H 回路为C，那么将v_1、$v_2 \dots v_k$去掉后，C 至多分为k段，因此图G'的连通支个数不超过k。

二分图存在H回路（道路）的一个必要条件

设 $G=(X,Y,E)$ 是一个二分图。若 $|X|$ 与 $|Y|$ 的差值大于 1，则 G 中不存在 H 道路；若 $|X|\neq|Y|$,则 G 中不存在 H 回路。

H道路存在的一个充分条件

如果简单图G 中任意两结点v_i、v_j 之间恒有$d(v_i) + d(v_j) \geq n - 1$，则$G$ 中存在H 道路。

H道路存在的一个充分条件

证明:

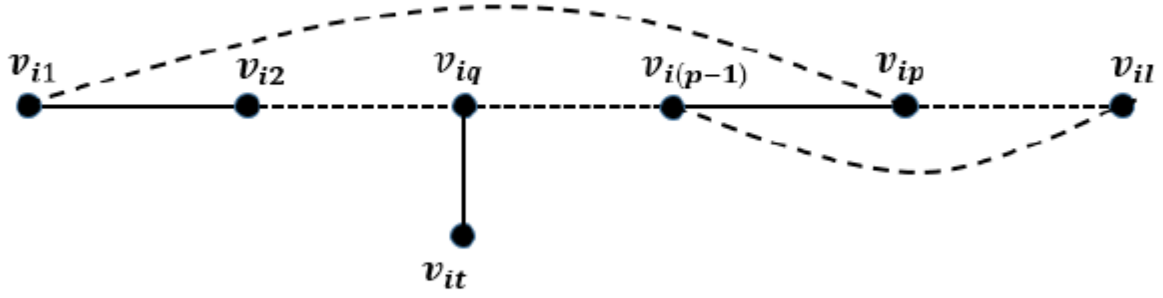
反证法证明 G 为连通图

若 G 非连通，则至少存在两个连通支 H_1, H_2，记其结点数目分别为 n_1, n_2。从 H_1, H_2 中各任取一个结点 v_i, v_j，因为 G 是简单图，故 $d(v_i) \leq n_1 - 1, d(v_j) \leq n_2 - 1$。由此不难推出 $d(v_i) + d(v_j) < n - 1$，与题意矛盾。故图 G 一定是联通图。

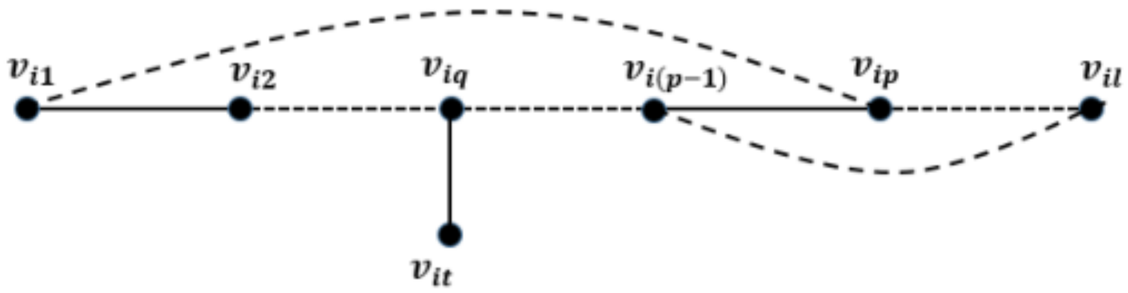
构造法证明 G 中存在 H 道路

构造 G 中一条极长的初级道路 $P = (v_{i1}, v_{i2}, ..., v_{il})$。

若 $l = n$，则 P 就是一条 H 道路。



H道路存在的一个充分条件



若 $l<n$，我们可以通过反证法证明 G 中一定存在经过节点 $v_{i1}, v_{i2}, ..., v_{il}$ 的初级回路 C。假设 $l<n$ 且 G 中不存在经过节点 $v_{i1}, v_{i2}, ..., v_{il}$ 的初级回路 C。因为 P 为极长的初级回路，所以 v_{i1}, v_{il} 的所有邻点都在 P 上[4]。因此若边 $(v_{i1}, v_{ip}) \in E(G)$，则 $(v_{il}, v_{i(p-1)}) \notin E(G)$[5]。又因为简单图中不存在自环，故 $d(v_{il}) \leq l-(d(v_{i1})+1)$，即 $d(v_{i1})+d(v_{il}) \leq l-1<n-1$，与题设矛盾。故 G 中一定存在经过节点 $v_{i1}, v_{i2}, ..., v_{il}$ 的初级回路 C。

又由于 G 连通，因此存在 C 之外的结点 v_t 与 C 中的某点 v_{iq} 相邻，删掉 $(v_{i(q-1)}, v_{iq})$，我们可以得到一条比 P 更长的初级道路。因为 G 为有穷图，重复上述过程，我们最终一定可以得到一条包含 G 中全部结点的初级道路，即 H 道路。 $\square$

一个推论

若简单图 G 中存在 H 道路，但不存在 H 回路，不妨设其 H 道路的两端点为 v_{i1} 和 v_{in}，则 $\mathrm{d}(v_{i1}) + \mathrm{d}(v_{in}) \leq n-1$。

证明： 设 $P = (v_{i1}, v_{i2}, ..., v_{in})$ 为 G 中的 H 道路。若边 $(v_{i1}, v_{ip}) \in E(G)$，则 $(v_{in}, v_{i(p-1)}) \notin E(G)$（否则删去 $(v_{i(p-1)}, v_{ip})$ 我们便得到了 G 中的 H 回路，与题设矛盾）。又因为简单图中不存在自环，故 $d(v_{in}) \leq n-(d(v_{i1})+1)$，即 $d(v_{i1}) + d(v_{in}) \leq n-1$。 □

H回路存在的一个充分条件

若简单图 G 的任意两结点 v_i, v_j之间恒有 $d(v_i) + d(v_j) \geq n$，则G中存在H回路。

H回路存在的充要条件

设G为简单图，v_i, v_j不相邻，且满足 $d(v_i) + d(v_j) \geq n$。则 G 存在 H 回路的充要条件是$G + (v_i, v_j)$ 有H回路。

证明：

必要性：若 G 存在 H 回路则 $G+(v_i, v_j)$ 一定存在 H 回路。

充分性：若 $G+(v_i, v_j)$ 存在 H 回路。假设 G 不存在 H 回路，则 $G+(v_i, v_j)$ 的 H 回路一定经过边 (v_i, v_j)。删去 (v_i, v_j)，得到 G 中的一条以 v_i, v_j 为端点的 H 道路，由引理*1*知 $d(v_i)+d(v_j) \leq n-1$，与条件矛盾，故 G 中存在 H 回路。 □

闭合图与哈密顿回路

若 v_i 和 v_j 是简单图 G 的不相邻结点，且满足 $d(v_i)+d(v_j)\geq n$，则令 $G'=G+(v_i,v_j)$，对 G' 重复上述过程，直至不再有这样的点对为止。最终得到的图称为 G 的**闭合图**，记做 $C(G)$。

简单图 G 存在 H 回路的充要条件是其闭合图存在 H 回路。

哈密顿回路(道路)

一般情况下判断一个图是否存在哈密顿回路是NP-complete问题

一种解决方式是搜索法

最坏复杂度$O(n!)$

NOIP知识点串讲 图论（一）

旅行商问题

旅行商问题

定义：给定一个正权完全图，求其总长最短的哈密顿回路

NP-complete问题

精确法 和 近似法

精确法：搜索$O(n!)$ 和 动态规划$O(n^2 2^n)$

近似法：贪心、模拟退火、遗传算法

现有最好记录： 精确法 85900个点

近似法 $2\% - 3\%$的误差范围内解决百万个点的情况

分支与界法

分支与界法是解决旅行商问题的一个确定性算法。算法基本思想是对边按权值排序后，按顺序搜索所有可能成为解的H回路，并通过已有解的上界对搜索树进行剪枝。

最坏复杂度为$O(n!)$

可以解决$40-70$个点的情况

Algorithm 4 分支与界法

```
1:  将边按权值从小到大排序
2:  d_0 ← +∞
3:  while 栈不为空 do
4:      if 能够按顺序继续选边 then
5:          按顺序选边加入栈
6:      else
7:          将栈中最长边删去,转3
8:      end if
9:      d(s) ← 栈中边的权值和
10:     if d(s) ≥ d_0 then
11:         将栈中最长两条边删去,转3
12:     end if
13:     if 栈中边数达到 n 条 then
14:         if 栈中边能构成 H 回路 then
15:             d_0 ← d(s)
16:             将栈中最长两条边删去,转3
17:         else
18:             将栈中最长边删去,转3
19:         end if
20:     else
21:         if 栈中边出现回路，或存在度数 > 3 的点 then
22:             将栈中最长边删去,转3
23:         end if
24:     end if
25: end while
```

便宜算法

- 针对边权符合三角不等式的无向正权图
- 基于的贪心的思想
- 在最初时维护的是一个边权和为 0 的自环
- 每次选取一个未在回路中且距离回路最近的点，贪心地将其加入回路
- 重复这个过程，直到得到 H回路

Algorithm 5 便宜算法

$T \leftarrow \{(1,1)\}$
while T 不是 H 回路 **do**
  $u \leftarrow$ 不在 T 中且距离 T 最近的点
  $v \leftarrow$ 在 T 中且距离 u 最近的点
  $v_1, v_2 \leftarrow v$ 在 T 中相邻的节点
  if $w(u, v_1) - w(v, v_1) \leq w(u, v_2) - w(v, v_2)$ **then**
    $T \leftarrow T - \{(v_1, v)\} + \{(u, v), (u, v_1)\}$
  else
    $T \leftarrow T - \{(v_2, v)\} + \{(u, v), (u, v_2)\}$
  end if
end while

便宜算法

设正权完全图的边权满足三角不等式，其旅行商问题的最佳解释 O_n，便宜算法的最优解是 T_n，则 $\frac{T_n}{O_n} < 2$。

NOIP知识点串讲 图论（一）

最短路

最短路

按照实际问题的模型可分为三类:

(1) 某两结点之间的最短路径

(2) 某结点到其他各结点的最短路径

(3) 任意两结点之间的最短路径

模型(2)如果能够解决，模型(1)和(3)自然可以解决

最短路

依据边权值的特点，有以下几种情况：

(1) 均大于零（正权图）

(2) 均等于1

(3) 为任意实数

最短路

v_1到v_i的最短路径中，是否会出现回路?

- 若回路长度为正，则删掉回路可以得到更短路径
- 若回路长度为负，则不存在最短路径

只讨论无负长回路的图

例3.

已知n个点的无向图G的边权均为1，求编号为1的节点到其余所有节点的最短路。

$n \leq 10^5, m \leq 10^7$

BFS

边权为1

距离源点近的点先被访问

用近的点去更新远的点

例4.

已知n个点的无向图G的边权均为1到5之间的整数，求编号为1的节点到其余所有节点的最短路。

$n \le 10^5, m \le 10^6$

BFS?

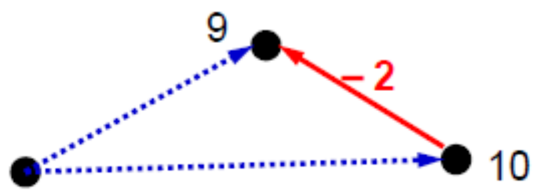
添加虚拟节点

DIJKSTRA算法

Dijkstra算法思想是由近及远扩展

得到某点的最短路径后不会再变

但有负权边时（不一定存在负长回路），Dijkstra算法可能会失效



Dijkstra算法只适用于正权图

DIJKSTRA算法

1.设从源结点沿已知最佳路径到本结点的距离$\pi(i)$

2.初始时，所有$\pi(i)$均为无穷大；

3.将源结点的$\pi(i)$设为0，令其为工作结点；

4.检查与工作结点i相邻的结点j，若$\pi(i)+w(i,j)\leq\pi(j)$，则更新$\pi(j)$

5.每次选择未当过工作节点的 $\pi(i)$最小 的结点，令其为下一轮工作结点；

重复第4、5步，直到目标结点成为工作结点。

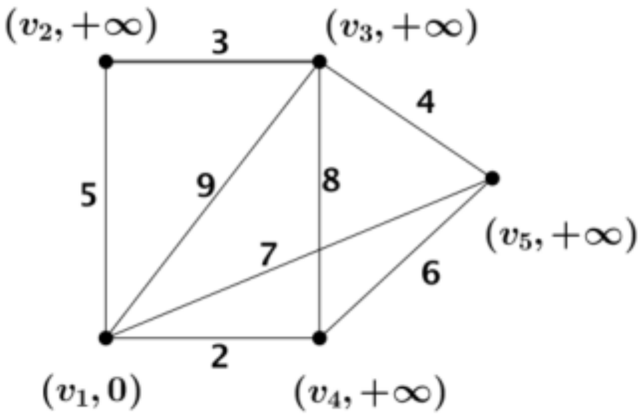
DIJKSTRA算法

Algorithm 7 Dijkstra 最短路

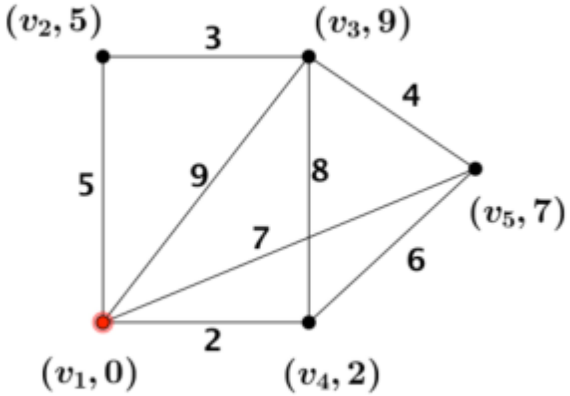
Input: v_s: 原点
Input: V: 点集
Input: E: 边集

$v_{now} \leftarrow v_s$
$\pi(v_i) = +\infty, \ \pi(v_s) = 0$
for $i = 1 \ to \ |V|$ **do**
    标记 v_{now}
    for all $v_i \in V$ **and** $(v_{now}, v_i) \in E$ **do**
        $\pi(v_i) \leftarrow min\{\pi(v_i), \pi(v_{now}) + w(v_{now}, v_i)\}$
    end for
    $v_{min} \leftarrow \pi$ 值最小且未被标记的 v_i
    $v_{now} \leftarrow v_{min}$
end for

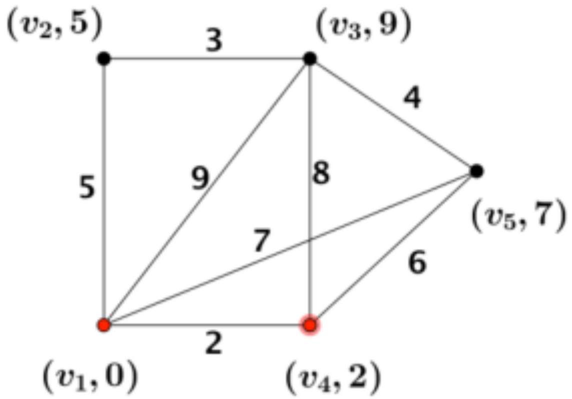
DIJKSTRA算法



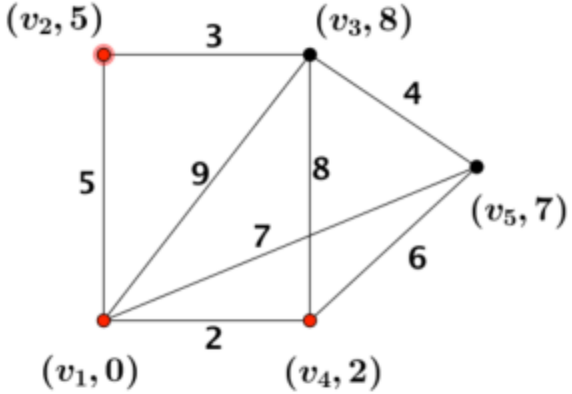
(c) 步骤 1



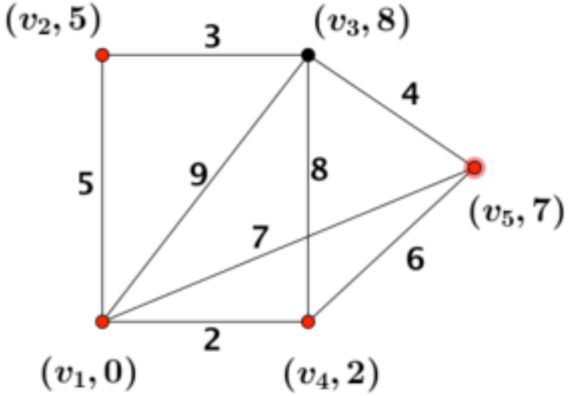
(d) 步骤 2



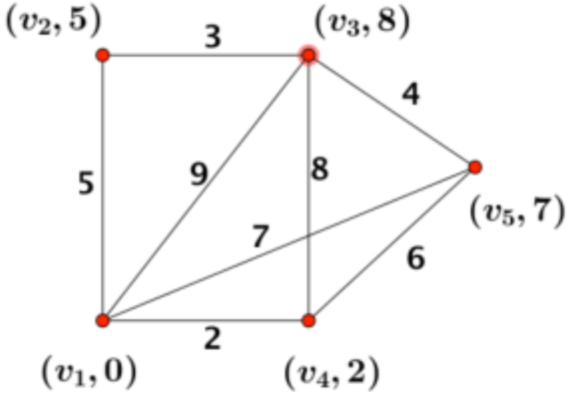
(e) 步骤 3



(f) 步骤 4



(g) 步骤 5



(h) 步骤 6

DIJKSTRA算法

从时间复杂度角度来看，我们发现，算法在稠密图（边数较多）上运行时效率不错，但是在稀疏图（边数较少）上效果不佳. 此时时间复杂度的瓶颈在于 $O(n^2)$，即计算最小值上，我们只要用高效的数据结构维护最小值就能突破瓶颈. 使用堆、平衡树等数据结构后，时间复杂度变为 $O(mlogn)$.

FORD算法

每次枚举每条边尝试更新，直到$\pi(i)$值不再发生变化

Algorithm 8 Ford 最短路

Input: v_s: 原点
Input: V: 点集
Input: E: 边集

$\pi(v_i) = +\infty,\ \pi(v_s) = 0$
repeat
  for all $(v_i, v_j) \in E$ **do**
    $\pi(v_j) \leftarrow min\{\pi(v_j), \pi(v_i) + w(v_i, v_j)\}$
  end for
until 没有 π 值被更新

FORD算法

最多更新$n-1$轮，否则存在负环

复杂度为O（mn）

SPFA算法

Ford 算法的迭代过程中有许多边的枚举是无意义的

只有在前一次迭代时被更新过的点才有更新其他点的可能

对Ford算法进行优化，得到SPFA算法

Algorithm 9 SPFA 最短路

Input: v_s: 原点
Input: V: 点集
Input: E: 边集

$\pi(v_i) = +\infty$，$\pi(v_s) = 0$
把 v_s 加入队列 Q，并标记 v_s
while Q *is not empty* **do**
  $v_i \leftarrow Q$ 的队首元素
  弹出 Q 的队首元素
  for all $(v_i, v_j) \in E$ **do**
    if $\pi(v_i) + w(v_i, v_j) < \pi(v_j)$ **then**
      $\pi(v_j) \leftarrow \pi(v_i) + w(v_i, v_j)$
      if v_j 未被标记 **then**
        把 v_j 加入队列 Q，并标记 v_j
      end if
    end if
  end for
  取消 v_i 标记
end while

注：队列是一种先进先出的数据结构

SPFA算法

同样，如果某个点进入队列超过 n 次，说明存在负环

本质上来说，SPFA 算法只是 Ford 算法的一个常数优化

在特定图，例如网格图上，SPFA 算法会退化到 $O(nm)$ 的时间复杂度

在一般的图上，SPFA 算法比Ford 算法要快得多

特别地，如果图是随机生成的，SPFA 算法的效率接近于 $O(mlogn)$

最常用的一种求最短路的算法

适用范围广泛，实现简单，效率惊人

但效率不稳定

对于特殊的稠密图

若为正权图

进行Dijkstra算法

若为负权图

可先删去负边，进行Dijkstra算法

再加上负边，进行SPFA算法

最短路

最长路？小于等于变大于等于，负环变正环

最短路问题难点不在于算法本身，而是在于建图

例5.

墨墨突然对等式很感兴趣，他正在研究$a_1x_1+a_2x_2+\cdots+a_Nx_N=B$存在非负整数解的条件。他想知道，给定$N$、$\{an\}$、以及$B$的取值范围，有多少$B$可以使等式存在非负整数解。

$N\le 12$，$0\le ai\le 5\times10^5$，$1\le B_{min}\le B_{max}\le 10^{12}$

例5.解

设$amin = \min(a_i)$

若b能被表示出，则$b + amin$也能被表示出

只需求出模$amin$意义下，每个剩余类中最小能被表示出的数，就能算出答案

建$amin$个点，$dis[i]$分别表示模$amin$意义下，i所在剩余类的最小能被表示出的数

对于每一个a_i，从b向$(b + a_i)\%amin$连一条边权为a_i的边

求最短路，得到$dis[i]$，即可求出答案

NOIP知识点串讲 图论（一）

差分约束

差分约束

n个变量$x_1, x_2, \dots, x_n$

m个形如$x_i - x_j \geq a_k$约束

其中a_k是一个常数，每个约束的常数可以不同

求满足约束的解

解的特点

有无数组解

已知一组解，每个变量加上同一个数也是一组可行解

因此，我们一般是求最小的非负解

观察不等式

$x_i - x_j \geq a_k$

剧透：$x_i \geq x_j + a_k$

再剧透：和最短（长？）路有关

解法

不等式和最短路中的三角形不等式相似

对于一条边$i \to j: w$，最后一定满足$dist[j] \le dist[i] + w$

不过此处我们是大于号，所以求的应该是最长路

对于每一个变量设一个点

不等式$x_i - x_j \ge a_k$对应边$j \to i: a_k$

由于所有变量非负，所以我们另设一个点S作为起点，$dist[S] = 0$

再向每个点连一条边权为0的边

即添加了不等式$x_i - x_S \ge 0$以及$x_S = 0$

解法

大功告成！只要用你喜欢的算法求最长路就行了，注意边权是否有负值

如何判是否有解？有正环就意味着无解

如果求最大解？不等式改成小于等于号，求最短路就好了

注意所有的不等式都应该是同样的符号，可以通过乘-1改变符号

如果是等式，可以用一个大于等于和一个小于等于表示

如果式子没有带等号（如$x_i - x_j < a_k$），那么修改常量

$x_i - x_j \le a_k - 1$（整数）$x_i - x_j \le a_k - eps$（实数）

例6.

给定n个正整数变量以及m组两个变量间的关系，关系为$\geq$、$\leq$、$>$、$<$和$=$中的一个。判断是否能满足所有关系，并求n个变量之和的最小值。

$n, m \leq 100{,}000$

来源：SCOI2011 糖果

例6.解法

求的是最小值，所以用最长路

小于等于的式子乘-1转换

如果有环呢?

如果环上都是带等号的边，那么环上所有变量的值必然相同，就直接缩成一个点；如果环上存在一条不带等号的边，那么就无解

用Tarjan进行缩点，缩完点之后就成了DAG（有向无环图）

可以按拓扑序DP求最短路

复杂度就成了线性

当然这个数据范围用堆优化Dijkstra也行，SPFA说不定也能过

例7.

有$0 \sim n$共$n+1$个数，每个数可以出现也可以不出现。有m个区间，用$[l,r]:k$描述，表示范围在l到r之间的数出现了至少k个。求最少出现了多少个数。如果不存在一组解则输出-1。

$n \le 50{,}000$，m反正不会太大。

来源：ZJU1508 Interval

例7.解法

这题的难点在于建模

我们对每个数设一个变量x_i。如果它出现了，那么$x_i = 1$，否则$x_i = 0$

对于一个区间$[l,r]$:k，有$x_l + x_{l+1} + \cdots + x_r \geq k$，但不符合形式

用$S[]$表示$x[]$的前缀和，显然有$0 \leq S_i - S_{i-1} \leq 1$

区间$[l,r]$:k的式子就变成了$S_r - S_{l-1} \geq k$

初值为$S_{-1} = 0$（因为题目里下标是从0开始的）

答案就是S_n的最小值，求最长路即可

例8

一家24小时营业的超市需要招聘出纳员。超市每个小时都需要不同数量的出纳员。用R_i表示一天中i点到$i+1$点这一小时内需要的出纳员数量，特别地R_{23}表示23点到次日0点需要的出纳员数量。每天的$R[]$都是相同的。可以有多于R_i的出纳员工作，但是绝对不能少于R_i人。

有 n 人应聘，每个人愿意从一个特定的整点开始连续工作8小时，求最少要招多少人。

$n \leq 1{,}000$。

来源： POJ1275/ZJU1420 Cashier Employment

例8.解

直观的想法是设变量$x_0 \sim x_{23}$表示每个时刻招多少人

设时刻i应聘的人数为t_i，那么有$0 \le x_i \le t_i$

到第i个时刻仍然在工作的人为$x_i + x_{i-1} \dots + x_{i-7}$，故有$\sum_{j=0}^{7} x_{i-j} \ge R_i$

同样定义$S[]$为$x[]$的前缀和，那么式子变为：

$$\begin{cases} S[i] - S[i-8] \ge R_i,\ 8 \le i \le 23 \\ S[23] + S[i] - S[i+16] \ge R_i,\ \ 0 \le i < 8 \end{cases}$$

当然还有$0 \le S[i] - S[i-1] \le t_i$

例8.解

可最后一个不等式中有3个变量，而且似乎都消不掉

但是这其中$S[23]$虽然是变量，但却在每一个式子中都出现了

可以把$S[23]$（也就是最后的答案）当已知量

枚举答案，判断是否可行

当然也可以二分答案

例9.

给定一个$N \times M$的矩阵$X[][]$以及两个数L和U（$L \le U$），问是否存在两个向量$A[]$和$B[]$满足$\forall i \in [1, N], j \in [1, M]$，有$L \le X[i][j] \times \frac{A[i]}{B[j]} \le U$。

$N, M \le 400$。

来源：HDU3666 The Matrix Problem

例9.解

对$A[]$和$B[]$的每个元素设点

原式两边乘$B[j]$得$L \times B[j] \leq X[i][j] \times A[i] \leq U \times B[j]$

但是差分约束系统中约束的变量不能带有系数

或许我们可以把乘法消掉。比如变成加法?

取对数!

原式变成$\lg L + \lg B[j] \leq \lg X[i][j] + \lg A[i]$

我们的变量就成了$\lg A[]$和$\lg B[]$

差分约束

差分约束系统的概念其实十分简单，解法也只是最短路算法而已

重点在于建模，以及对于模型性质的深入分析

拿不准的时候可以猜性质然后暴力对拍检验

NOIP知识点串讲 图论（一）

关键路径

关键路径

在规划一个工程的时候，常有若干工序需要考虑

每道工序有各自的预期完成时间

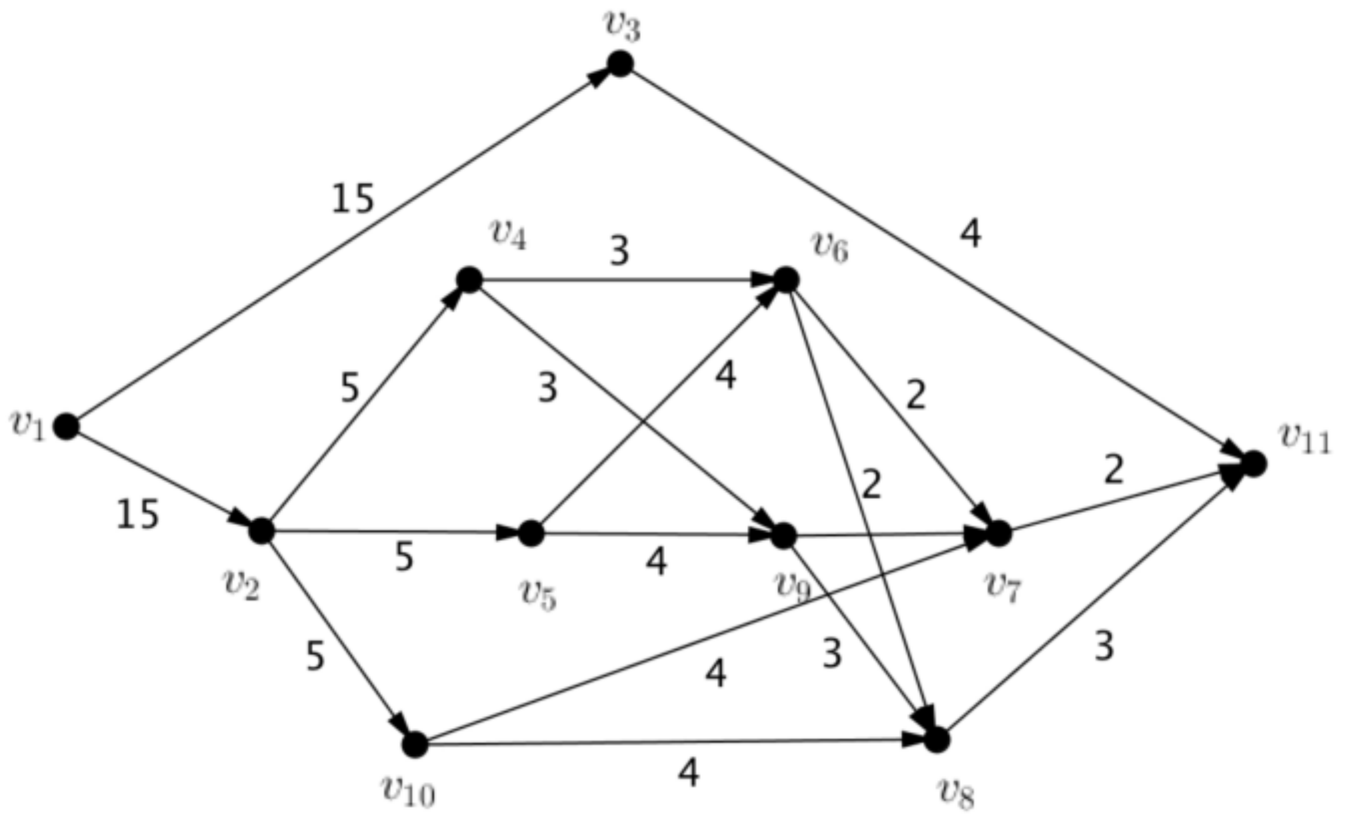
工序之间也会有依赖关系

想知道完成工程的最少时间

每项工序的最早开始时间、最晚开始时间

PT图

序号	名称	所需时间（天）	先序工序
1	基础设施	15	
2	下部砌砖	5	1
3	电线安装	4	1
4	圈梁支模	3	2
5	水暖管道	4	2
6	大梁安装	2	4,5
7	楼板吊装	2	6,9,10
8	楼板浇模	3	6,9,10
9	吊装楼梯	3	4,5
10	上部砌砖	4	2



用点表示工序

边表示依赖关系

边权表示工序的完成时间

PT图

一定是DAG

可以进行拓扑排序

完成工程的最少时间：最长路

每项工序的最早开始时间：到每个点的最长路

最晚开始时间：源点到终点的最长路减去到终点的最长路

复杂度：线性

参考资料

刘明华、朱佳豪、沈子翔，《图论》改编教材

胡泽聪，《差分约束》

NOIP知识点串讲 图论（二）

Colin

树

树的等价定义
DFS序
最近公共祖先
哈夫曼树
最小生成树
生成树计数
Prüfer序列
斯坦那树

NOIP知识点串讲 图论（二）

树的等价定义

树T的等价性质

T连通且无回路

T连通且每条边都是割边

T连通且有$n-1$条边

T有$n-1$条边且无回路

T的任意两结点间有唯一道路

T无回路，但在任意两结点间加上一条边后恰有一个回路

NOIP知识点串讲 图论（二）

DFS序

例1.

有一棵 N 个结点的树，以编号为1的结点为根，每个结点有一个权值。共有 M 个操作，可分为三种：

·把某个节点x的点权增加a。

·把某个以节点x为根的子树中所有点的权值都增加a。

·询问某个节点x到根的路径中所有点的点权和。

$N, M \leq 100000$

来源：HAOI2015 T2

DFS序

按DFS访问节点的顺序

入栈时加入序列 \ 入栈和出栈时各加入一次

一棵子树对应DFS序中连续一段

u和v的最近公共祖先为DFS序中u和v对应区间中深度最小的节点

对子树进行操作时，通常需要用到DFS序

例1.解

DFS序

入栈时为正，出栈时为负

子树操作，对序列中连续一段操作

一个点到根路径上的权值和就是序列中[1,入栈位置]的区间和

用线段树维护即可

NOIP知识点串讲 **图论（二）**

最小生成树

最小生成树性质

设生成树T的最大边权为$f(T)$，最小生成树为T'。则$f(T') = min(f(T))$

设V'是结点集V的真子集，则连接V'和V的权值最小的边一定在最小生成树中

KRUSCAL算法

将边按权值升序排序

每次按顺序尝试加入一条新边

若加入新边后未产生回路，则加入这条新边

直到形成一棵生成树

复杂度$O(mlogm)$

PRIM算法

初始时$V' = \{v_0\}$（v_0为任意一点）

每次选取一条满足$u \in V - V', v \in V'$且权值最小的边(u, v)

将(u, v)加入生成树，将u加入V'

直到$V = V'$

使用堆优化时，复杂度为$O(E\log(V))$

使用斐波拉契堆优化时，复杂度为$O(E + V\log(V))$

例4.

动物园里有n个景点，第i个景点有a_i头动物，有m条道路连接这n个景点。每次熊孩子们从景点u走到景点v，都会选择一条路径上$\min(a_i)$最大的路径，并设这个值为$f(u,v)$。求所有$f(u,v)$的平均值，即$\frac{\sum_{u\neq v} f(u,v)}{n(n-1)}$。

$2 \leq n \leq 10^5, 0 \leq m \leq 10^5$

来源： CF437D The Child and Zoo

例4.解

对于每条边边权设为两个点中较小的a_i

一定走最大生成树上的路径

考虑每条边的贡献

按照边权降序加入边，用并查集维护连通信息

一条树边的贡献次数为加入时两端点连通块大小的乘积

K小生成树

通过一些性质可以得到一定在前k小生成树中的边和一定不在的边

从而可以通过缩点构造新图，新图的规模为k个点，$2k-2$条边

通过对新图求前k小生成树，我们就可以得到原图的前k小生成树

有兴趣的同学可参见2014年国家队论文中俞鼎力同学部分

NOIP知识点串讲 **图论（二）** | 最近公共祖先

例2.

A 国有n座城市，编号从1到n，城市之间有m条双向道路。每一条道路对车辆都有重量限制，简称限重。现在有q辆货车在运输货物，第i辆货车需要从城市x_i运输货物到城市y_i。司机们想知道每辆车在不超过车辆限重的情况下，最多能运多重的货物。

$0 < n < 10000$, $0 < m < 50000$，$0 < q < 30000$

来源：NOIP2013 货车运输

例2.解

一定走最大生成树上的边

询问树上点对间路径的最小边权

求lca（最近公共祖先）

暴力求LCA

让两个点暴力往上跳

在线算法

复杂度$O(qh)$，h为树的深度

随机情况下为$O(qlogn)$

最坏情况下为$O(qn)$

RMQ求LCA

等价于询问dfs序中，u,v间深度最小的点

转化为求区间最值问题

在线算法

使用ST表，可做到$O(nlogn+q)$

倍增求LCA

需先预处理出每个点向上跳2^i步的祖先，即Up数组

让较深的点跳到两点高度相同的位置

再让两点一起往上跳

在线算法

复杂度$O(nlogn + qlogn)$

Algorithm 2 LCA(u,v)

```
if Dep[u] < Dep[v] then
    SWAP(u, v)
end if
t ← Dep[u] − Dep[v]
i ← 0
while t > 0 do
    if t%2 = 1 then
        u ← Up[u][i]
    end if
    t ← t/2
    i ← i + 1
end while
t ← MAX_HEIGHT
while u ≠ v do
    while t > 0 ∧ Up[u][t] = Up[v][t] do
        t ← t − 1
    end while
    u ← Up[u][t]
    v ← Up[v][t]
end while
return u
```

TARJAN算法

用并差集维护祖先信息

设询问为(u,v)

在后访问到的点（不妨设为u）回答

- 若u,v为祖孙关系则v为祖先

  访问到u时，$parent[v]\ =v$

- 若u,v非祖先关系

  访问到u时，$parent[v]=lca(u,v)$

离线算法

复杂度为$O(n+q)$

Algorithm 1 Tarjan(u)

```
visit[u] ← true
for all (u, v) ∈ QUERY do
    if visit[u] = true then
        ans(u, v) ← Find(v)
    end if
end for
for all (u, v) ∈ TREE do
    if visit[u] = false then
        Tarjan(v)
        parent[v] ← u
    end if
end for
```

parent 为并查集
Find 为并查集的查找操作
QUERY 为询问结点对集合
TREE 为树边集合

用数据结求LCA

在线算法

使用树链剖分或LCT时复杂度为$O(qlogn)$

NOIP知识点串讲 图论（二）

哈夫曼树

例3.

一部《荷马史诗》中有n种不同的单词，从1到n进行编号。其中第i种单词出现的总次数为 w_i。Allison 想要用k进制串s_i来替换第i种单词，使得其任意的$i \neq j$，都有s_i不是s_j的前缀。

现在 Allison 想要知道，如何选择 s_i，才能使替换以后得到的新的《荷马史诗》长度最小。在确保总长度最小的情况下，Allison 还想知道最长的s_i的最短为多少?

$2 \leq n \leq 100000$，$2 \leq k \leq 9$

来源：Noi2015 荷马史诗

哈夫曼树

先考虑$k = 2$的情况

以树的形式表示编码

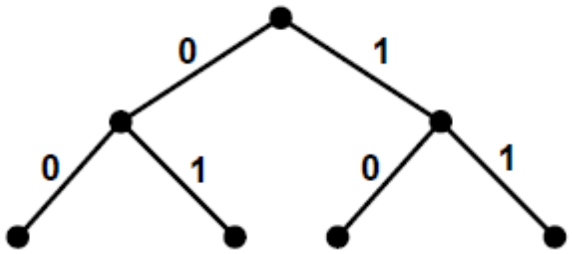
叶子节点代表一种单词

根到叶子的路径表示一种编码

可以保证不会有不同单词存在前缀关系

用出现次数表示每个叶子的权值

替换后文本最短等价于树的带权路径总长（WPL）最小



构造方法

初始时每个节点各自为一棵子树，权值为其出现频率

每次选取权值最小的两棵子树，将其合并为一棵新子树，新子树的权值为两棵子树权值和

重复这个过程，直到只剩下一棵子树

正确性

权值最小的叶子节点（设为u）的深度一定最深

否则通过替换，可以得到一棵更优的树

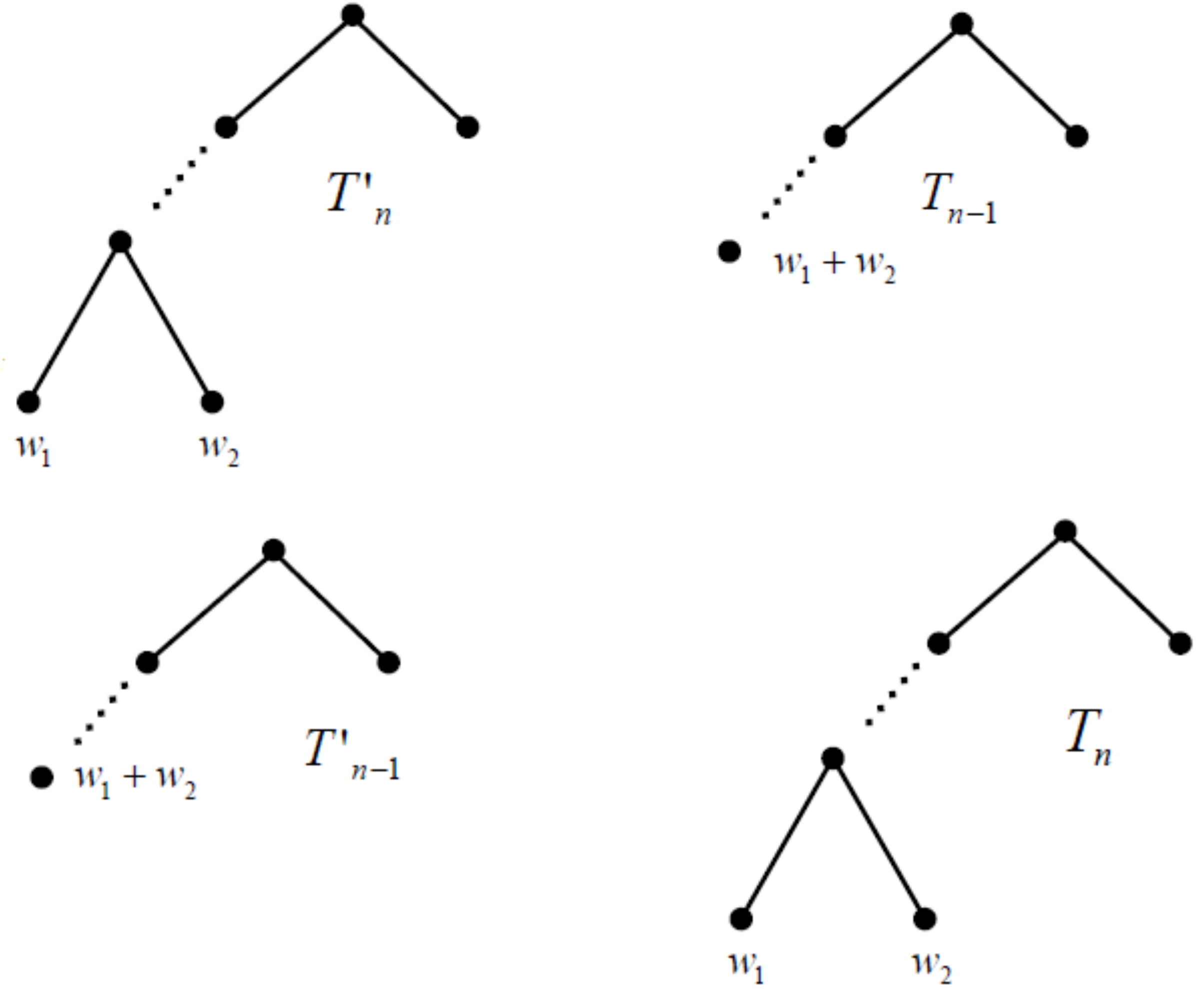
u一定有兄弟节点

否则将u移到父亲位置，可以得到一棵更优的树

u的兄弟节点一定为权值次大的节点

否则通过替换，可以得到一棵更优的树

哈夫曼树正确性



$$WPL(T'_n)=WPL(T_{n-1})+(w_1+w_2)$$

$\wedge\ \vee\qquad\wedge\ \vee$

$$WPL(T_n)=WPL(T'_{n-1})+(w_1+w_2)$$

哈夫曼树正确性

当只有两个结点时，得到的树一定是最优的

当结点数超过2时，由归纳法可证得到的树仍然是最优的

例3.解

对于$k \neq 2$的情况构造方法与正确性证明均类似

NOIP知识点串讲

图论（二）

生成树计数

例5.

已知n个点和m条边，求生成树的数目对10^9+7取模的结果。

$n \leq 100$

生成树计数

关联矩阵

基本关联矩阵

矩阵A的行列式$\det(A)$

行列式计算：高斯消元不改变行列式的绝对值

上三角矩阵的对角线元素乘积即为行列式

设图G的基本关联矩阵为B_k，则图G的生成树个数为$\det\left(B_k B_k^T\right)$

矩阵树定理

余子式

代数余子式

若连通图G的邻接矩阵为A，将A的对角线元素(i,i)依次替换为节点v_i的度$d(v_i)$，记所得矩阵为M，则M的代数余子式即为G的生成树的数目

NOIP知识点串讲 图论（二）

Prüfer序列

例6.

给定n个标号节点，求在满足节点i的度数为d_i的情况下，完全图的生成树数目。

$1 \le n \le 150$，数据保证满足条件的树不超过10^{17}个

来源：HNOI2004 树的计数

思路

- 分类讨论？容斥？
- 构造双射
- n个标号节点的树到长度为$n-2$，值为1到n的序列的双射

TREE TO SEQUENCE

1）将树中标号最小的叶子删除

2）并将其相邻节点的标号加入序列

3）重复前两步，直到树中只剩下两个节点

TREE TO SEQUENCE

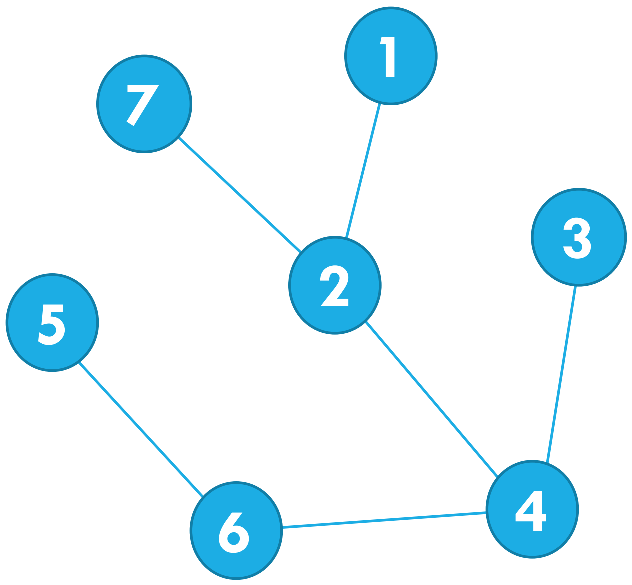
1）将树中标号最小的叶子删除

2）并将其相邻节点的标号加入序列

3）重复前两步，直到树中只剩下两个节点

- Sequence： 2 4 6 4 2

- 序列长度为$n-2$
- 节点i在序列中出现d_i-1次



SEQUENCE TO TREE

1）根据序列确定每个点的度数d_i

2）找到标号最小的d_i等于1的节点u

3）将u与序列中第一个节点v连边

4）将d_u和d_v减一，并将v从序列中删除

5）重复第2到4步，直到序列为空

6）将此时d_i等于1的两个点连边

SEQUENCE TO TREE

1） 根据序列确定每个点的度数d_i

2） 找到标号最小的d_i等于1的节点u

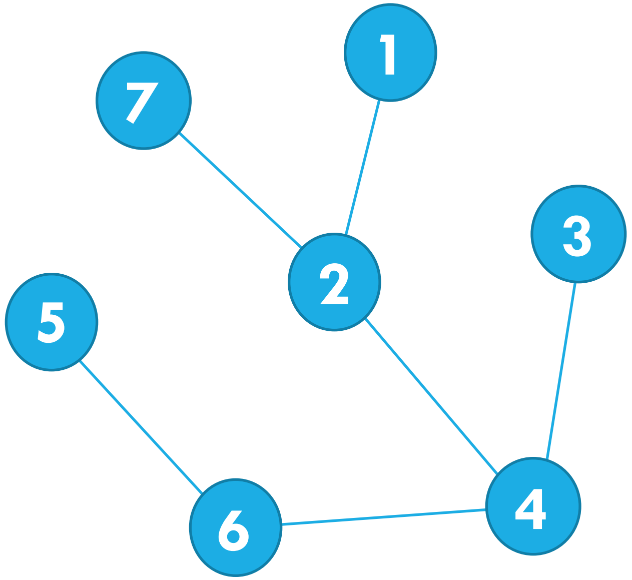
3） 将u与序列中第一个节点v连边

4） 将d_u和d_v减一，并将v从序列中删除

5） 重复第2到4步，直到序列为空

6） 将此时d_i等于1的两个点连边

- 任何时刻每个连通支都只有一个d_i非0的点，故不会连出回路



Sequence：2 4 6 4 2

d_i: 1 3 1 3 1 2 1

d_i: 0 2 1 3 1 2 1

d_i: 0 2 0 2 1 2 1

d_i: 0 2 0 2 0 1 1

d_i: 0 2 0 1 0 0 1

d_i: 0 1 0 0 0 0 1

正确性

双射：单射 & 满射

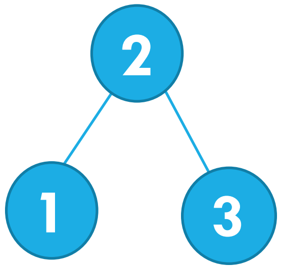
单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳

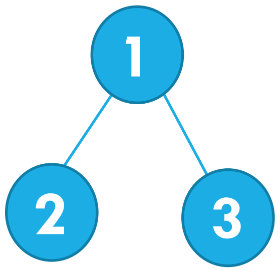
单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳

3个节点的树：构成一条链，两棵树不同当且仅当中间节点不同



2



1

单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳

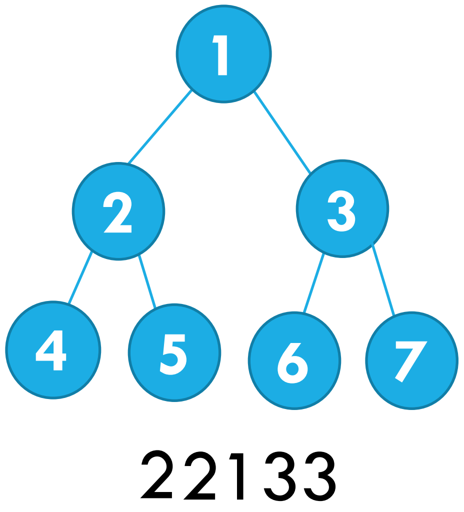
3个节点的树：构成一条链，两棵树不同当且仅当中间节点不同

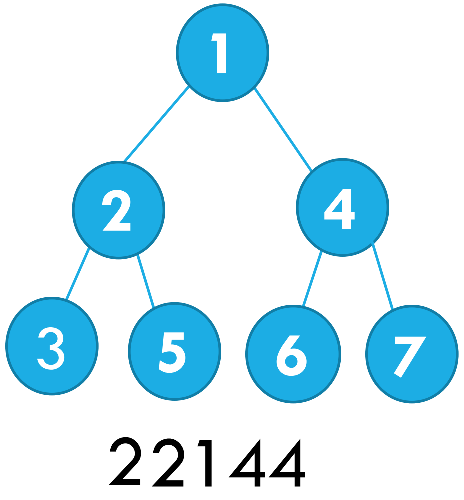
n个节点的树:

设T_1中标号最小的叶子为u，T_2中标号最小的叶子为v

单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳





3个节点的树：构成一条链，两棵树不同当且仅当中间节点不同

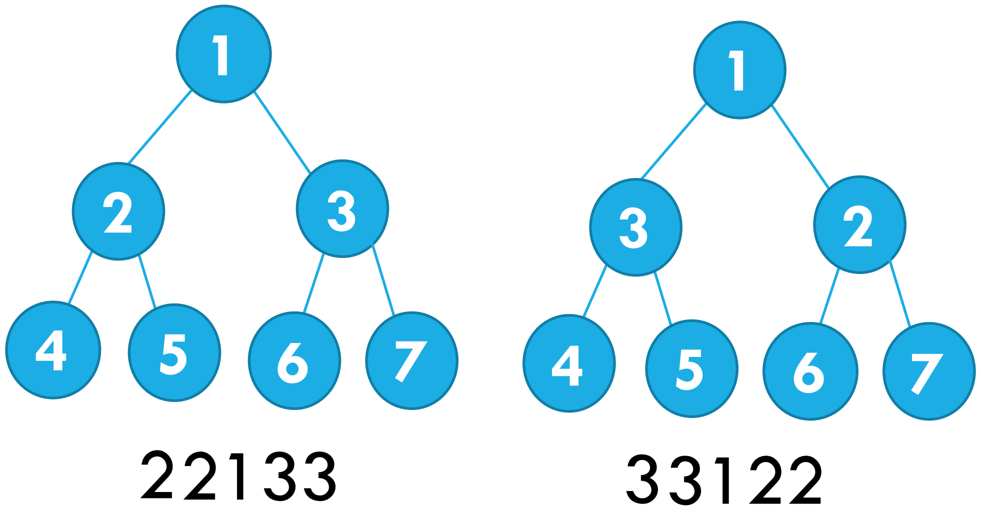
n个节点的树：

设T_1中标号最小的叶子为u，T_2中标号最小的叶子为v

- 若u和v不同：则T_1、T_2分别除去u、v后的部分必然不同，此时我们得到两棵规模更小的树，由归纳假设知序列的后$n-3$位必然不同

单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳



3个节点的树：构成一条链，两棵树不同当且仅当中间节点不同

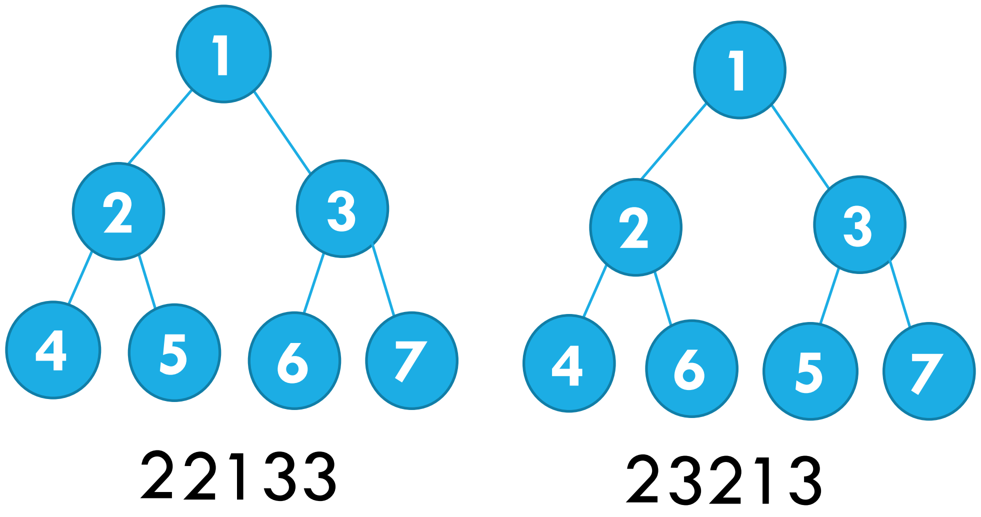
n个节点的树：

设T_1中标号最小的叶子为u，T_2中标号最小的叶子为v

- 若u和v不同：则T_1、T_2分别除去u、v后的部分必然不同，此时我们得到两棵规模更小的树，由归纳假设知序列的后$n-3$位必然不同
- 若u和v相同，但u和v在树中相邻的节点不同：则序列的第一位必然不同

单射

- 只需证任意两个不同的树所映射的序列不同
- 对树的节点数进行归纳



3个节点的树：构成一条链，两棵树不同当且仅当中间节点不同

n个节点的树：

设T_1中标号最小的叶子为u，T_2中标号最小的叶子为v

- 若u和v不同：则T_1、T_2分别除去u、v后的部分必然不同，此时我们得到两棵规模更小的树，由归纳假设知序列的后$n-3$位必然不同
- 若u和v相同，但u和v在树中相邻的节点不同：则序列的第一位必然不同
- 若u和v相同，且u和v在树中相邻的节点也相同：则T_1、T_2分别除去u、v后的部分必然不同，由归纳假设知序列的后$n-3$位必然不同

满射

- 要证每个序列都有一棵树映射过来
- 依据“sequence to tree”的构造，我们知任意一个长度为$n-2$的序列S可以得到一棵n个节点的树T
- 通过归纳法不难证明T所映射的序列就是S
- 故由n个标号节点构成的树到长度为$n-2$，值为$1-n$的序列的映射为双射

转化

- 给定n个标号节点，在满足节点i的度数为d_i的情况下，完全图的生成树数目
- 长度为$n-2$，其中i出现了d_i-1次的序列的数目
- $\frac{(n-2)!}{(d_1-1)!\times\cdots(d_n-1)!}$

PRÜFER序列

- Heinz Prüfer在1918年证明Cayley's formula时提出

应用：

- 1.给定n个标号节点，满足节点i的度数为d_i的树的数目为：$\frac{(n-2)!}{(d_1-1)!\times\cdots(d_n-1)!}$
- 2.Cayley's formula：n个标号节点的完全图生成树个数为n^{n-2}
- 3.通过在枚举Prüfer 序列时加以限制，我们可求得完全二分图的生成树个数
- 4.通过生成Prüfer序列来随机生成n个标号节点的树

NOIP知识点串讲 图论（二）

斯坦纳树

例7.

从未来过绍兴的小 D 有幸参加了 Winter Camp 2008，他被这座历史名城的秀丽风景所吸引，强烈要求游览绍兴及其周边的所有景点。

主办者将绍兴划分为 N 行 M 列（$N \times M$）个方块，如下图（8×8）：

				沈园			
				八字桥			
			周恩来故居				东湖
					大禹陵		
	兰亭						
鉴湖							

景点含于方块内，且一个方块至多有一个景点。无景点的方块视为路。

为了保证安全与便利，主办方依据路况和治安状况，在非景点的一些方块内安排不同数量的志愿者；在景点内聘请导游（导游不是志愿者）。在选择旅游方案时，保证任意两个景点之间，存在一条路径，在这条路径所经过的每一个方块都有志愿者或者该方块为景点。既能满足选手们游览的需要，又能够让志愿者的总数最少。

例7.

例如，在上面的例子中，在每个没有景点的方块中填入一个数字，表示控制该方块最少需要的志愿者数目：

1	4	1	3	4	2	4	1
4	3	1	2	沈园	1	2	3
3	2	1	3	八字桥	3	1	2
2	6	5	周恩来故居	2	4	1	东湖
5	1	2	1	3	4	2	5
5	1	3	1	5	大禹陵	1	4
5	兰亭	6	1	4	5	3	4
鉴湖	2	2	2	3	4	1	1

图中用深色标出的方块区域就是一种可行的志愿者安排方案，一共需要 20 名志愿者。由图可见，两个相邻的景点是直接（有景点内的路）连通的（如沈园和八字桥）。

现在，希望你能够帮助主办方找到一种最好的安排方案。

N,M,K≤10，其中K为景点的数目

斯坦纳树

斯坦纳树（Minimal Steiner Tree）问题是组合优化问题

将指定点集合中的所有点连通，且边权总和最小的生成树

与最小生成树相似

最小生成树是在给定的点集和边中寻求最短网络使所有点连通

而最小斯坦纳树允许在给定点外增加额外的点

斯坦纳树

考虑用状压DP求解

dp[i][state]表示以i为根，指定集合中的点的连通状态为state的生成树的最小总权值

一种转移：

dp[i][state∪i]=min{ dp[i][state∪i], dp[j][state]+e[i][j] }

只能得到链?

补充另一种转移：

dp[i][state]=min{ dp[i][subset1]+dp[i][subset2] }

State = subset1 ∪ subset2

算法流程

从小到大枚举state（算到state时，其子集已算完）

对于每个state：

dp[i][state]=min{ dp[i][subset1]+dp[i][subset2] }

需要枚举子集

for(sub=(state-1)&state; sub; sub=(sub-1)&state)

dp[i][state∪i]=min{ dp[i][state∪i], dp[j][state]+e[i][j] }

将当前state的dp[i][state]全部加入spfa的队列，去更新其他点

正确性

两种转移可以覆盖所有情况

从小到大枚举state：

算到state时，其子集都已算完

算完state时，其用加边操作的转移都已完成

复杂度

两部分：枚举子集和spfa

枚举子集：

共2^k个状态

子集数？

sum{C(i,k)*2^i} (1<=i<=k) = 3^k (二项式展开)

O（n*3^k）

spfa：

每一个state进行一轮spfa

O(2^k*c*E)

O(n*3^k + 2^k*c*E)

例8.

花花收到了一块海星巧克力，共有 $r \times c$ 小块。每一小块都有自己的形状，他们有的是海星，有的是贝壳，还有的是海螺……其中还有一些因为挤压，已经分辨不出来是什么形状了。

花花给每一块巧克力都标上了一个美味程度，这个值越大表示这一块越美味。

正当花花用舌头舔了舔嘴巴，准备开始享受美味时，源源屁颠屁颠地跑过来了。

看到源源恳求的目光，花花决定从中选出一些和源源一起分享。首先花花希望选出的这些巧克力是连通的（这里的连通指有公共边），然后这些巧克力要包含至少k种形状，而那些被挤压过的巧克力，是不能被选中的。

但自私的花花又想把美味的巧克力尽量多地给自己，所以他希望选出的巧克力的美味值之和尽可能的小。你能帮帮他么?

$r, c \leq 25$，$k \leq 5, -1 \leq a_{i,j} \leq r \times c, 0 \leq b_{i,j} \leq 10^5, T \leq 20$。

例8.

对于aij较小的数据，可以直接套用现成的斯坦树算法求解。

对于aij较大的数据，我们考虑加入随机化算法，变到上面一个问题。

由于k比较小，我们可以将所有的形状随机分成k组

将每个组的形状认为是同一种

从而变成只有k种形状

套用斯坦那树算法进行求解

正确性

求出的解是满足合法性限制

而只有当原问题最优解的那k种形状全都分到不同组，才得到最优解

最优解发生的概率?

设原来共出现了t种形状

期望意义下，这k类中每一类都包含原来的$\frac{t}{k}$种形状

进行一次这样的随机，取到最优解的概率近似是:

$$\frac{(\frac{t}{k})^k}{C(t,k)} = \frac{k!}{k^k} \geq 0.0384$$

随机200次就基本能对

例8.

“人生就像一盒巧克力，你永远不知道吃到的下一块是什么味道。”

明明收到了一大块巧克力，里面有若干小块，排成 n 行 m 列。每一小块都有自己特别的图案 $c_{i,j}$，它们有的是海星，有的是贝壳，有的是海螺……其中还有一些因为挤压，已经分辨不出是什么图案了。明明给每一小块巧克力标上了一个美味值 $a_{i,j}$ ($0 \le a_{i,j} \le 10^6$)，这个值越大，表示这一小块巧克力越美味。

正当明明咽了咽口水，准备享用美味时，舟舟神奇地出现了。看到舟舟恳求的目光，明明决定从中选出一些小块与舟舟一同分享。

舟舟希望这些被选出的巧克力是连通的（两块巧克力连通当且仅当他们有公共边），而且这些巧克力要包含至少 k ($1 \le k \le 5$) 种。而那些被挤压过的巧克力则是不能被选中的。

明明想满足舟舟的愿望，但他又有点“抠”，想将美味尽可能多地留给自己。所以明明希望选出的巧克力块数能够尽可能地少。如果在选出的块数最少的前提下，美味值的中位数（我们定义 n 个数的中位数为第 $\left\lfloor \frac{n+1}{2} \right\rfloor$ 小的数）能够达到最小就更好了。

你能帮帮明明吗？

$1 \le 颜色数 \le n \times m \le 233$

例8.

只要求块数最少?

点权都是1，变成上一题

随机k染色+斯坦那树

只要求中位数最小?

二分中位数

权值比中位数大的设为1，小的设为-1，求最小的斯坦那树

随机k染色+斯坦纳树

例8.

个数最少的前提下，中位数最小?

二分中位数

取一个较大的数M

若权值比中位数大，设为M+1，权值比中位数小，设为M-1

由于+1和-1不会影响到M的个数

故可以得到个数最少前提下的中位数最小。

参考资料

张小平，《图论》课件

刘明华，《Prüfer序列》

THANKS FOR YOUR LISTENING