

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

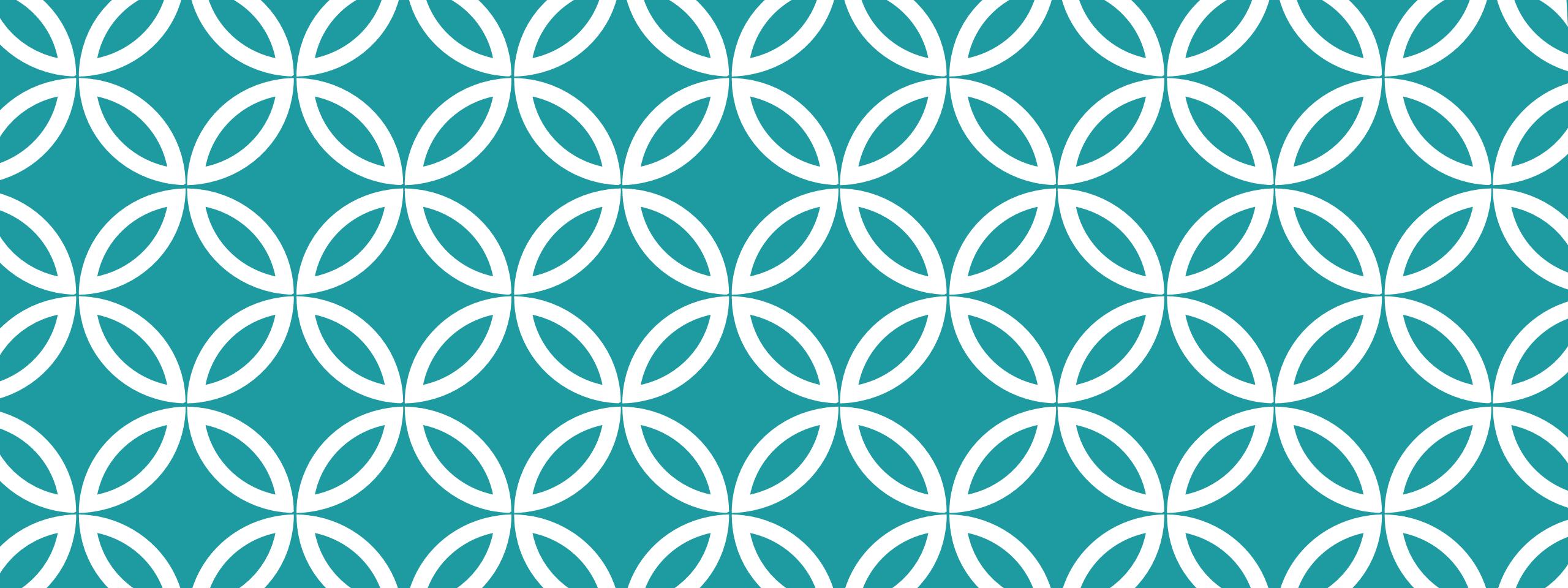
最近公共祖先

哈夫曼树

最小生成树

生成树计数

prufer序列



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）

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

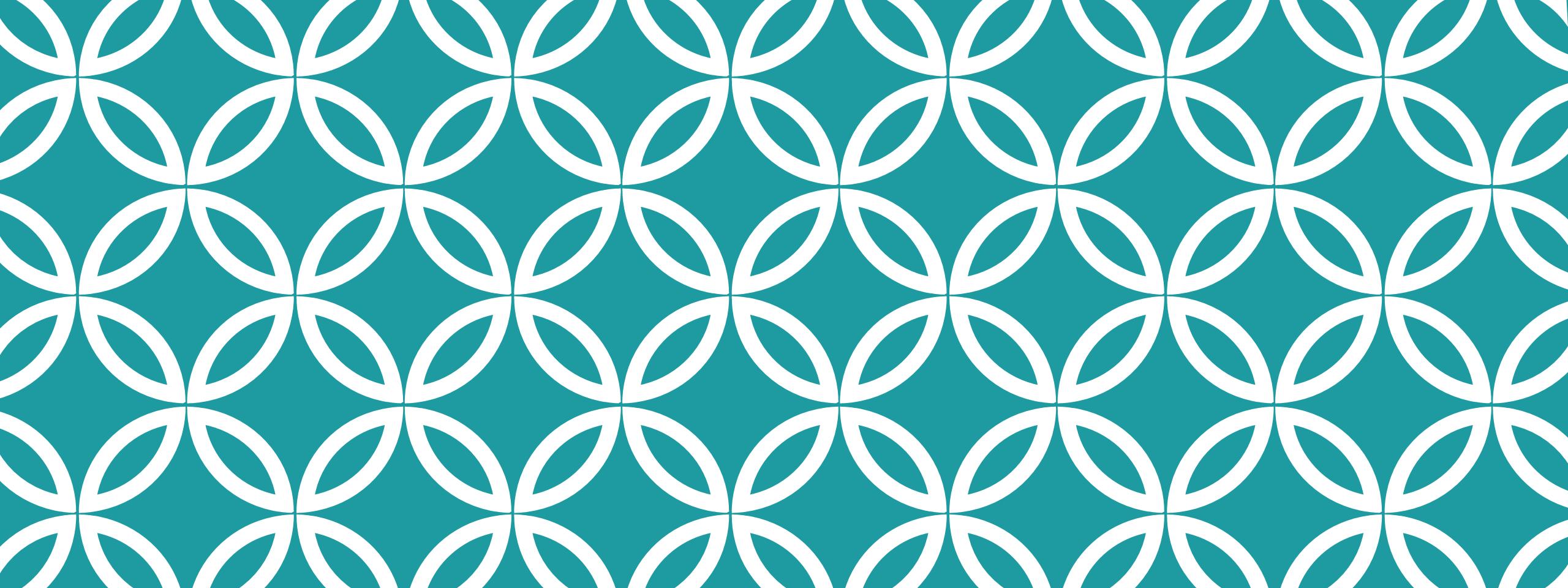
二分图具备很多特殊的性质

故我们常要对原图进行二分图染色

# DAG（有向无环图）

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

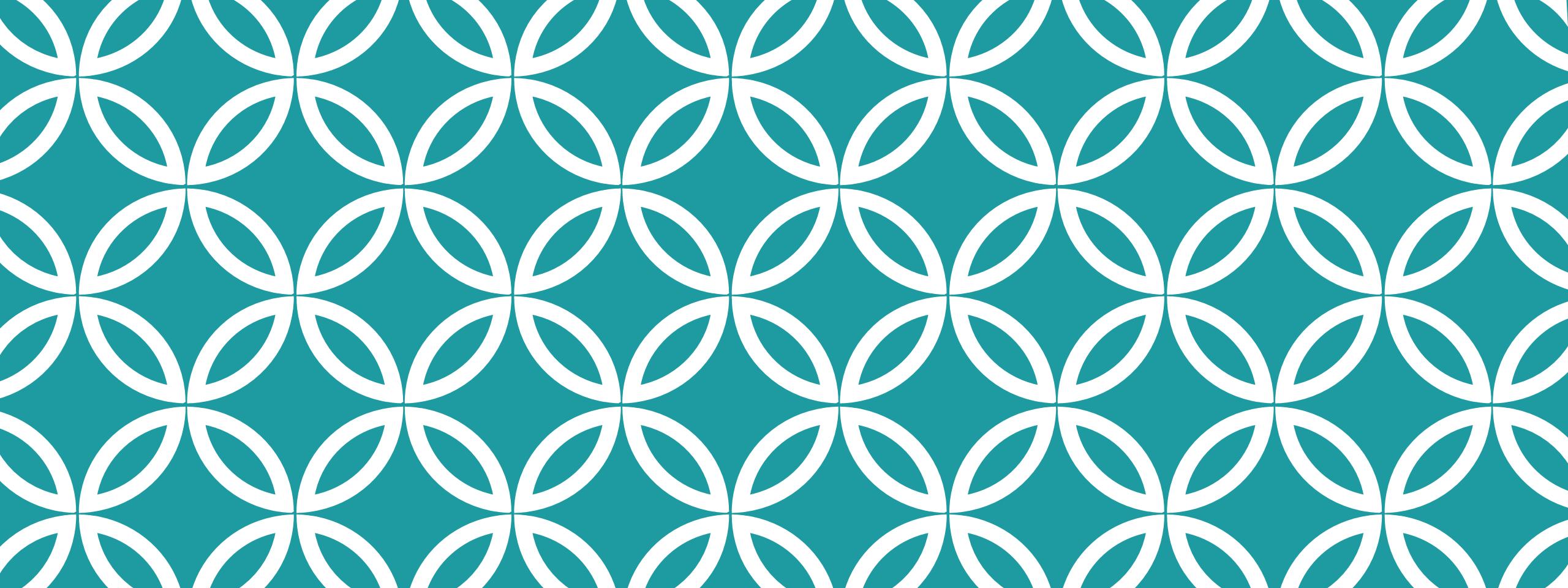
邻接矩阵

权矩阵

关联矩阵

边列表

邻接表



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。

$$n \leq 100, l \leq 10^9$$

## 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  $(v_i, v_{k_1}, v_{k_2}, \dots, v_{k_l}, v_j)$  存在当且仅当  $a_{ik_1}a_{k_1k_2}a_{k_2k_3}\dots a_{k_lj} = 1$ 。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 $O(n^3 \log l)$

## 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  $P$

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

复杂度为  $O(n^4)$

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

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

复杂度仍为  $O(n^4)$

# WARSHALL 算法

---

## Algorithm 1 Warshall 算法

---

```
1: 矩阵  $P \leftarrow A$ 
2: for all  $i = 1 \rightarrow n$  do
3:   for all  $j = 1 \rightarrow n$  do
4:     for all  $k = 1 \rightarrow n$  do
5:        $p_{jk} \leftarrow p_{jk} \vee (p_{ji} \wedge p_{ik})$ 
6:     end for
7:   end for
8: end for
```

---

证明：

对  $i$  进行归纳：

当  $i = 1$  时，我们注意到

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

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  $\{v_j, v_k, v_1, v_2, \dots, v_{t-1}\}$  之间存在  $v_j$  到达  $v_k$  的道路。

对于  $i = t$  的情形，由于

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

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  $\{v_j, v_1, v_2, \dots, v_t, v_k\}$  中存在不经过  $v_t$  的从  $v_j$  到  $v_k$  的道路，后者等价于  $\{v_j, v_1, v_2, \dots, v_t, v_k\}$  中存在经过  $v_t$  的从  $v_j$  到  $v_k$  的道路。故  $p_{jk}^{(t)} = 1$  当且仅当结点集合  $\{v_j, v_k, v_1, v_2, \dots, v_t\}$  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

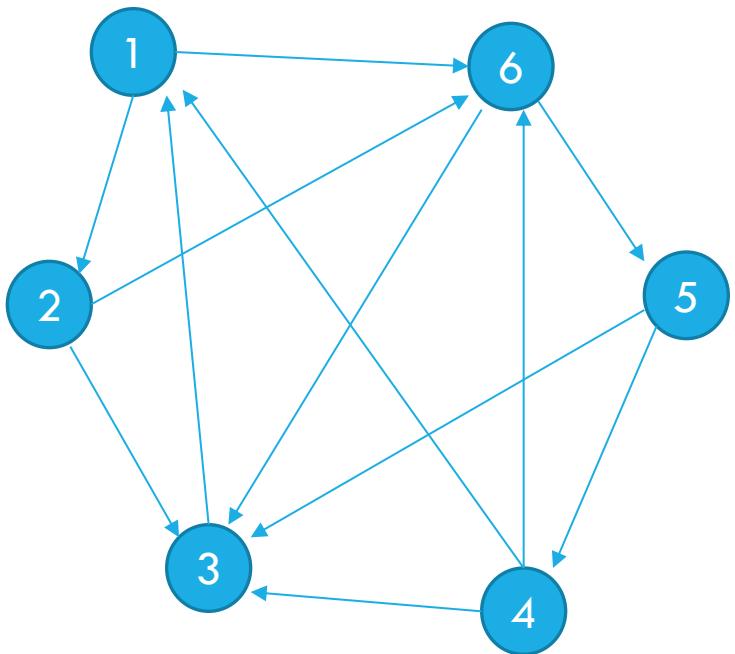
深度优先搜索算法

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



---

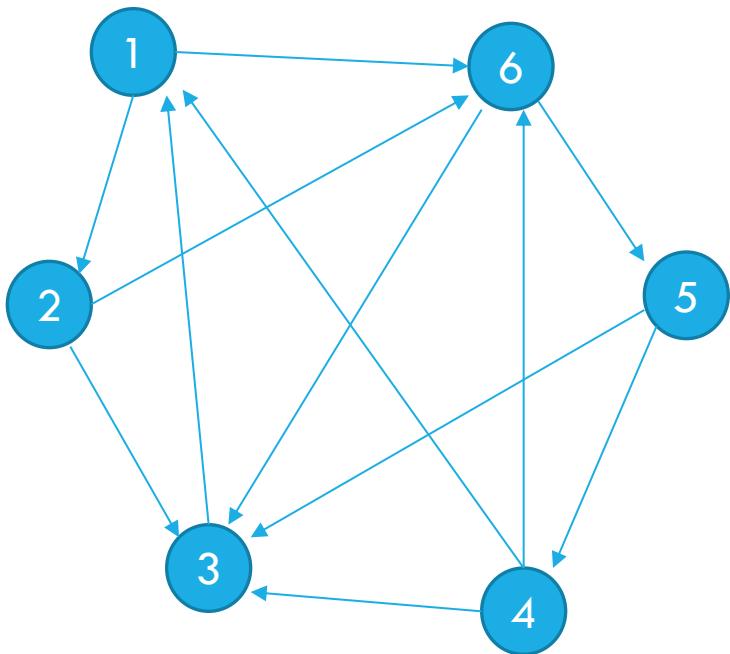
## Algorithm 2 BFS

---

```
1: for all  $u \in V(G)$  do
2:    $vis[u] \leftarrow false$ 
3: end for
4: 任选一个节点  $v_0$  加入队列尾端
5:  $vis[v_0] \leftarrow true$ 
6: while 队列不为空 do
7:   弹出队列首端元素  $u$ 
8:   for all  $(u, v) \in E(G)$  do
9:     if  $vis[v] = false$  then
10:       将  $v$  加入队列尾端
11:        $vis[v] \leftarrow true$ 
12:     end if
13:   end for
14: end while
```

---

# DFS



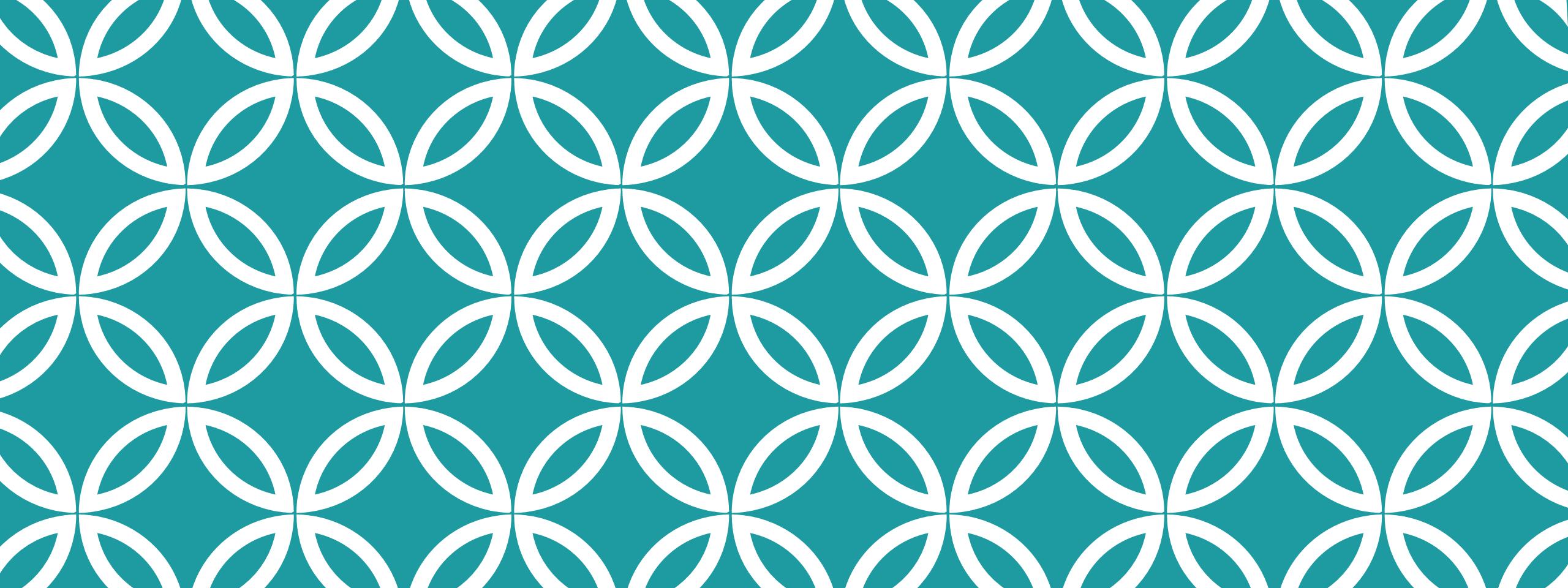
---

**Algorithm 3** DFS

---

```
1: for all  $u \in V(G)$  do
2:    $vis[u] \leftarrow false$ 
3: end for
4: 任选一个节点  $v_0$  加入队列首端
5:  $vis[v_0] \leftarrow true$ 
6: while 队列不为空 do
7:   弹出队列首端元素  $u$ 
8:   for all  $(u, v) \in E(G)$  do
9:     if  $vis[v] = false$  then
10:       将  $v$  加入队列首端
11:        $vis[v] \leftarrow true$ 
12:     end if
13:   end for
14: end while
```

---



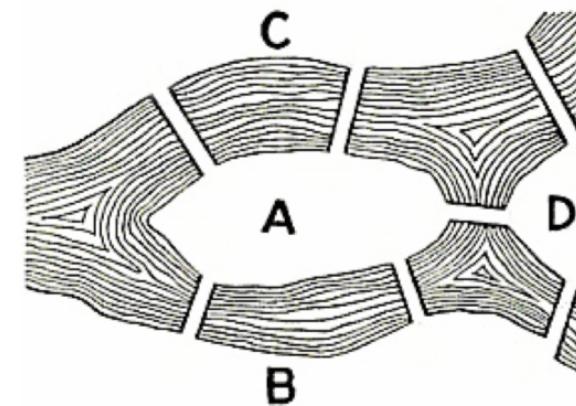
NOIP知识点串讲 **图论（一）** | 欧拉回路

# 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。

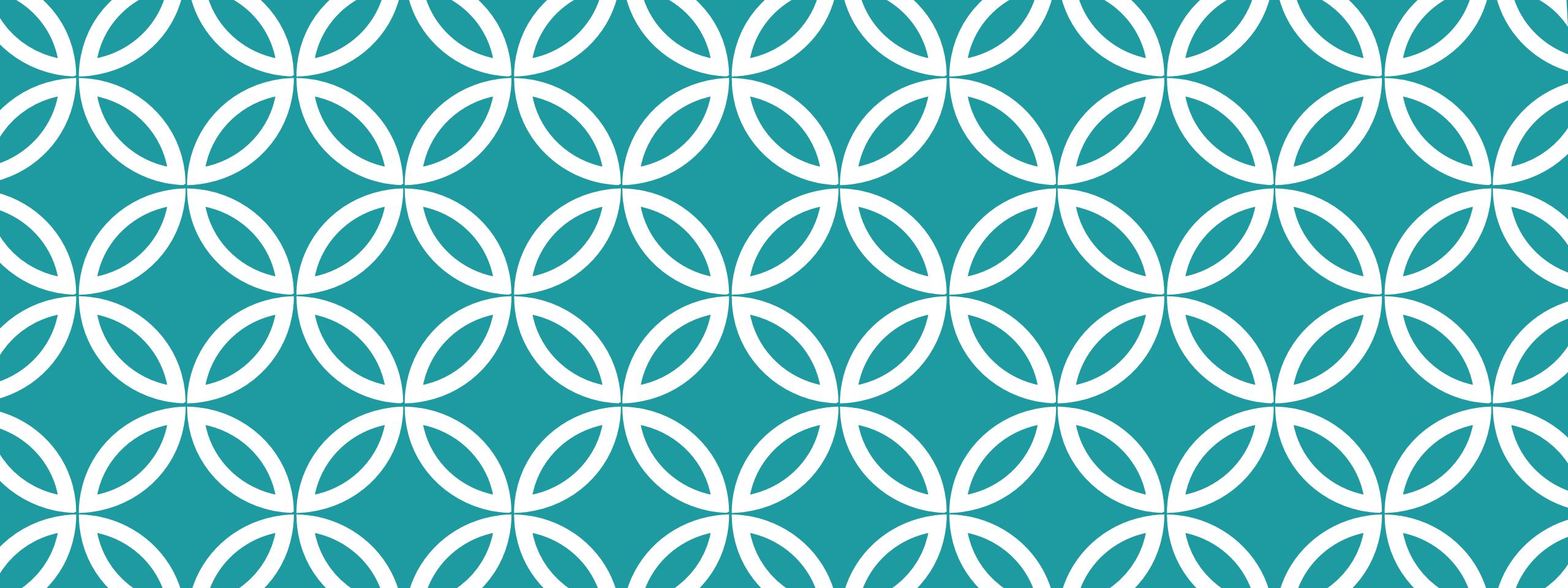
如果不是，那么由于原图是连通的， $C_1$  和原图的其它部分（即  $G - C_1$ ）必然有公共顶点。由于在  $G - C_1$  中每个节点度数任然是偶数，故从这个公共顶点出发，在原图的剩余部分中将得到另一回路。两个回路相连得到一个更长的回路，经过若干步后我们一定能将原图所有边包含进来，此时我们得到原图的欧拉回路。  $\square$

## 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。

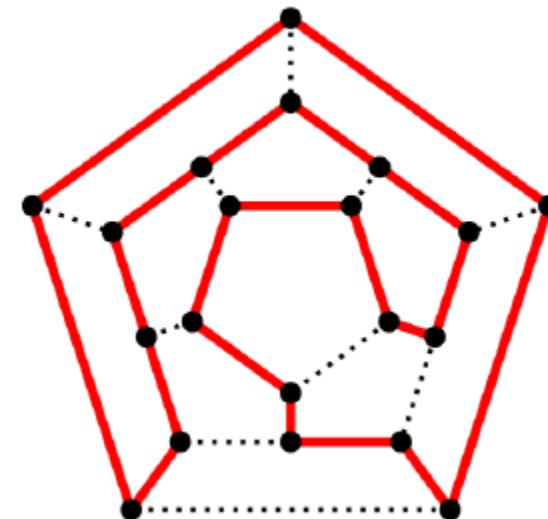


NOIP知识点串讲 **图论（一）**

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：

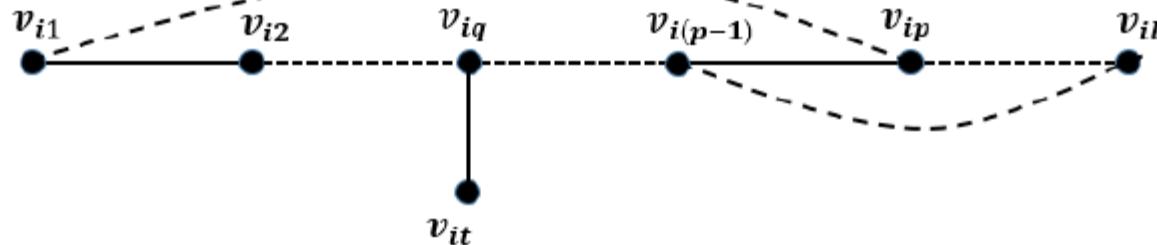
反证法证明  $G$  为连通图

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。

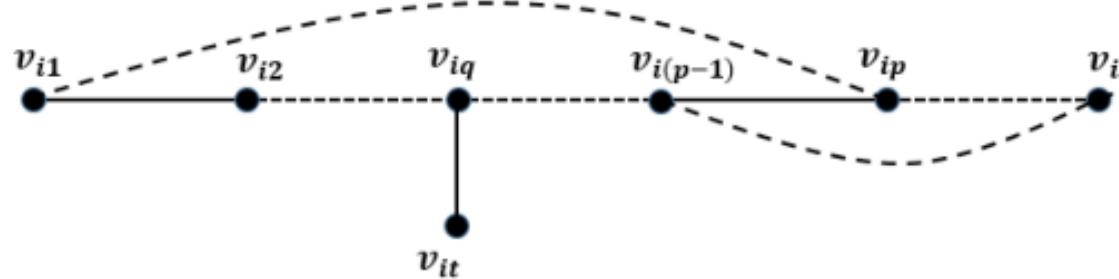
构造法证明  $G$  中存在  $H$  道路

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

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  $v_{i1}, v_{i2}, \dots, v_{il}$  的初级回路  $C$ 。假设  $l < n$  且  $G$  中不存在经过节点  $v_{i1}, v_{i2}, \dots, v_{il}$  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<sup>4</sup>。因此若边  $(v_{i1}, v_{ip}) \in E(G)$ , 则  $(v_{il}, v_{i(p-1)}) \notin E(G)$ <sup>5</sup>。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  $v_{i1}, v_{i2}, \dots, v_{il}$  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。□

## 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  $d(v_{i1}) + d(v_{in}) \leq n - 1$ 。

证明：设  $P = (v_{i1}, v_{i2}, \dots, v_{in})$  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  $d(v_{i1}) + d(v_{in}) \leq n - 1$ 。  $\square$

## 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  $C(G)$ 。

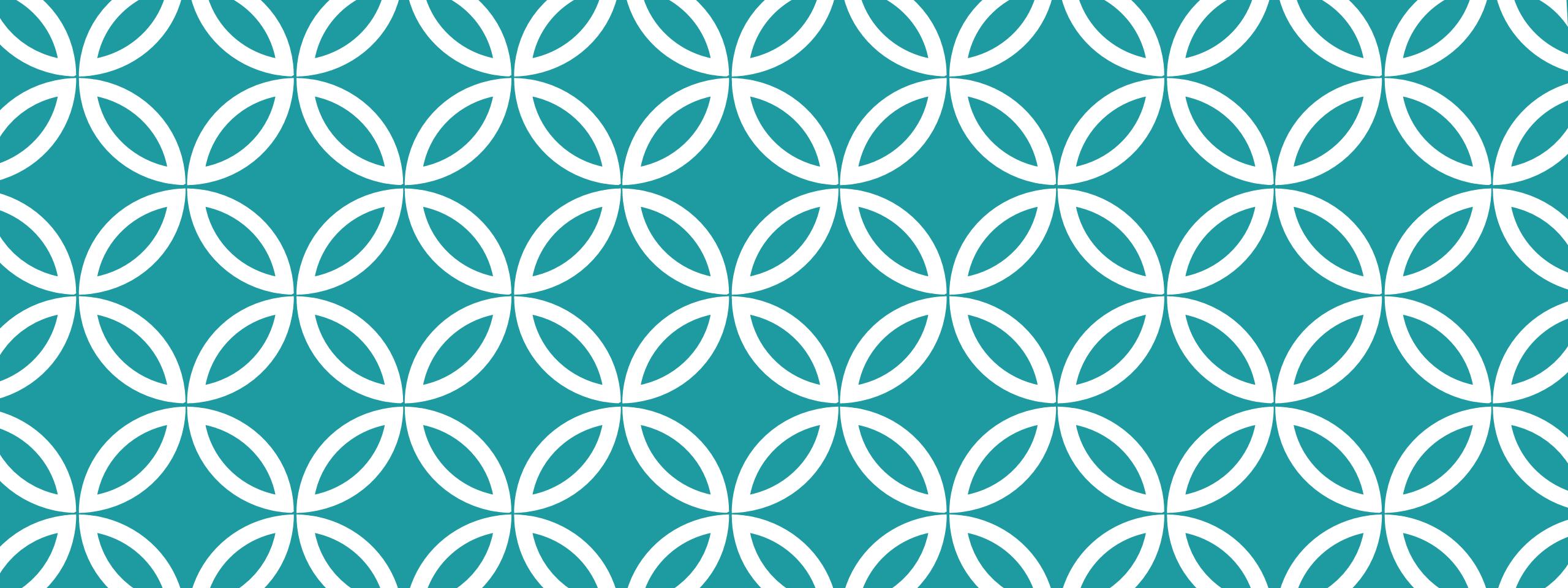
简单图  $G$  存在  $H$  回路的充要条件是其闭合图存在  $H$  回路。

## 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:  $d_0 \leftarrow +\infty$ 
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:    $d(s) \leftarrow$  栈中边的权值和
10:  if  $d(s) \geq d_0$  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:       $d_0 \leftarrow d(s)$ 
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

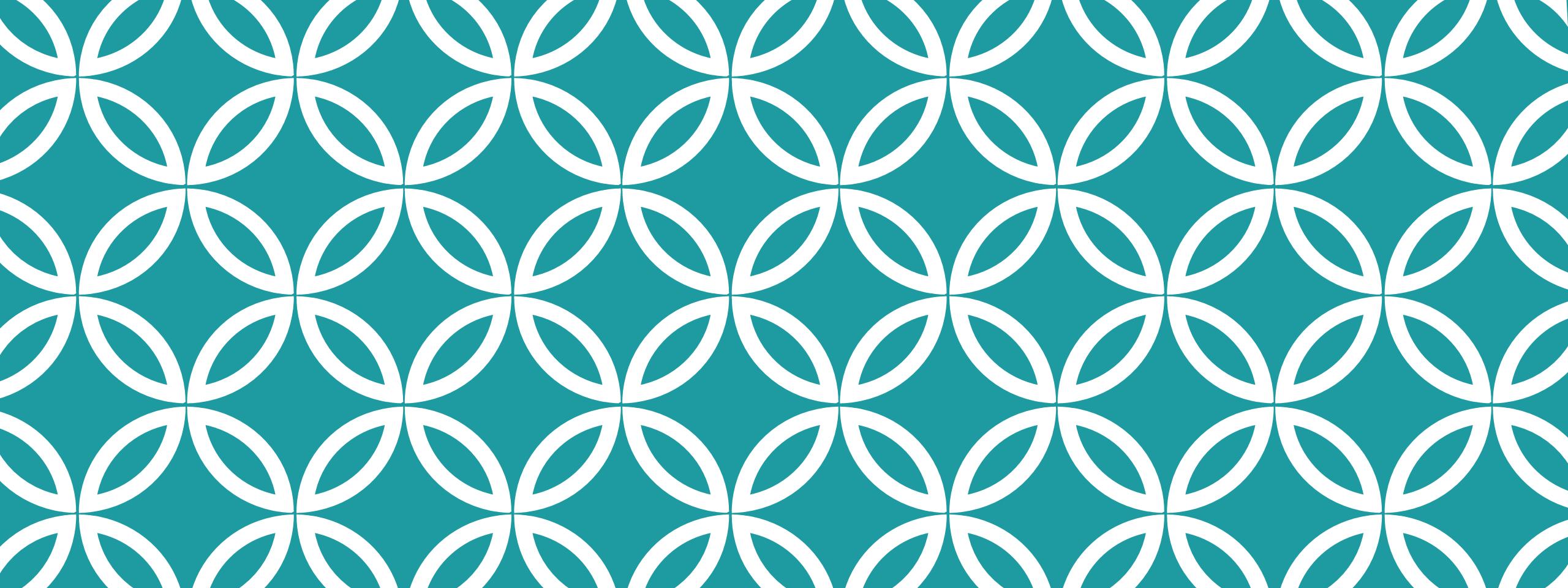
---

```
1:  $T \leftarrow \{(1, 1)\}$ 
2: while  $T$  不是  $H$  回路 do
3:    $u \leftarrow$  不在  $T$  中且距离  $T$  最近的点
4:    $v \leftarrow$  在  $T$  中且距离  $u$  最近的点
5:    $v_1, v_2 \leftarrow v$  在  $T$  中相邻的节点
6:   if  $w(u, v_1) - w(v, v_1) \leq w(u, v_2) - w(v, v_2)$  then
7:      $T \leftarrow T - \{(v_1, v)\} + \{(u, v), (u, v_1)\}$ 
8:   else
9:      $T \leftarrow T - \{(v_2, v)\} + \{(u, v), (u, v_2)\}$ 
10:  end if
11: 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。

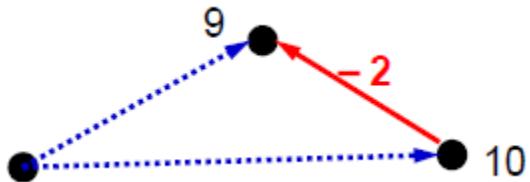
$$n \leq 10^5, m \leq 10^6$$

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

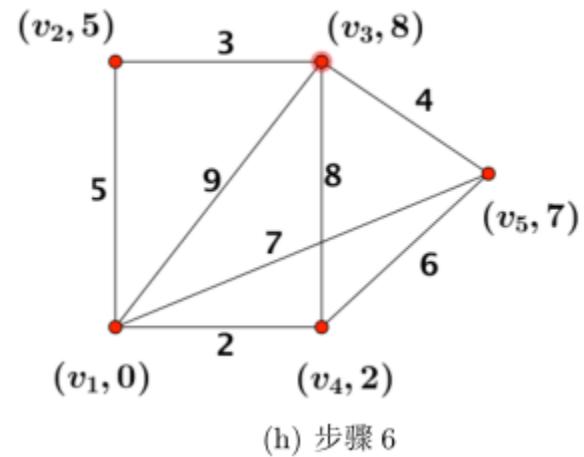
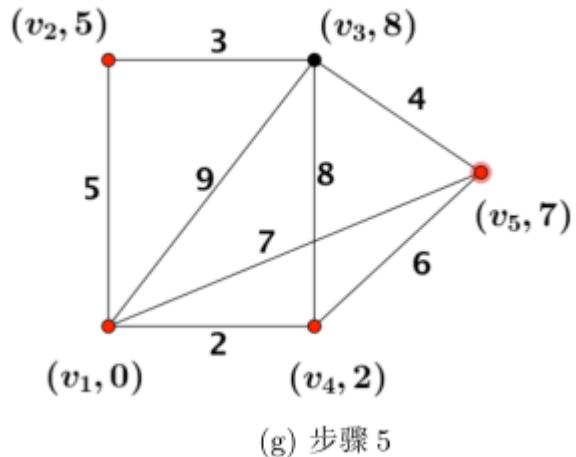
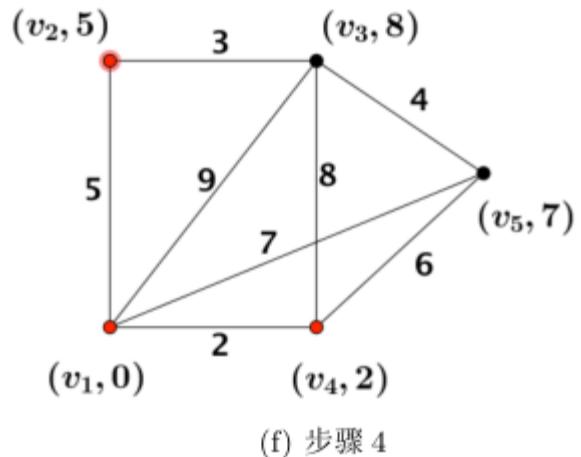
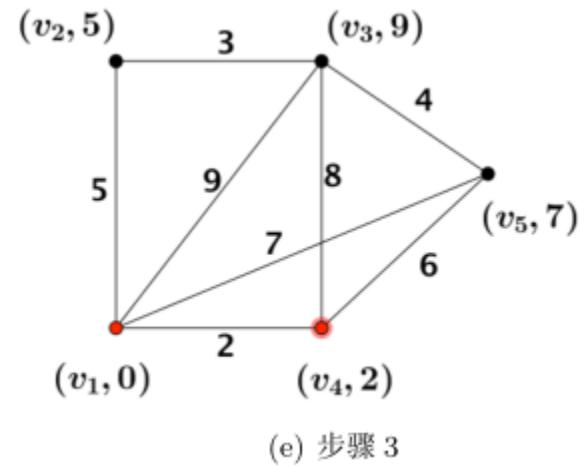
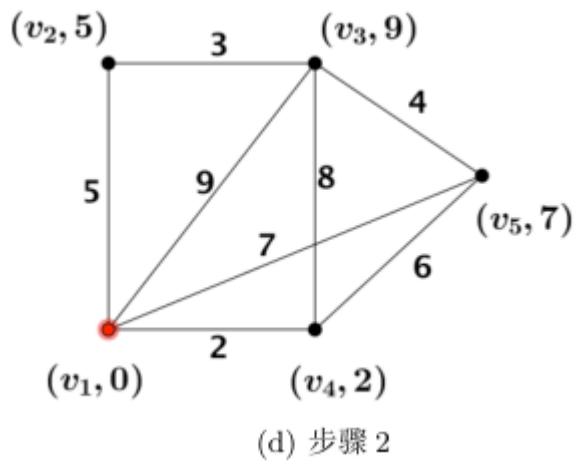
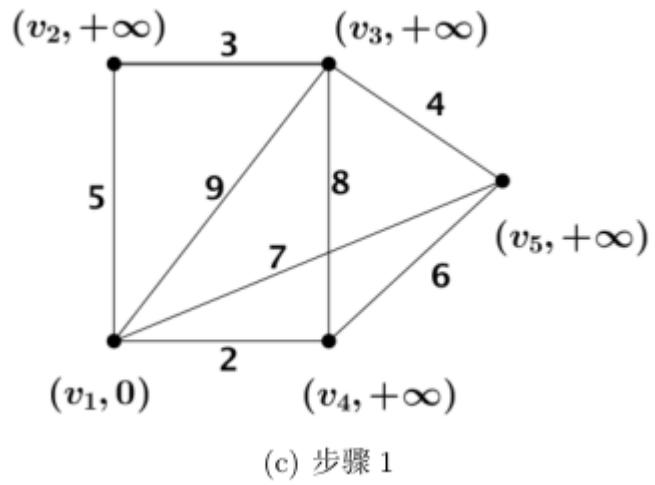
**Input:**  $V$ : 点集

**Input:**  $E$ : 边集

```
1:  $v_{now} \leftarrow v_s$ 
2:  $\pi(v_i) = +\infty, \pi(v_s) = 0$ 
3: for  $i = 1$  to  $|V|$  do
4:   标记  $v_{now}$ 
5:   for all  $v_i \in V$  and  $(v_{now}, v_i) \in E$  do
6:      $\pi(v_i) \leftarrow \min\{\pi(v_i), \pi(v_{now}) + w(v_{now}, v_i)\}$ 
7:   end for
8:    $v_{min} \leftarrow \pi$  值最小且未被标记的  $v_i$ 
9:    $v_{now} \leftarrow v_{min}$ 
10: end for
```

---

# DIJKSTRA 算法



# 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  $O(m \log n)$ .

# 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

1:  $\pi(v_i) = +\infty$ ,  $\pi(v_s) = 0$

2: **repeat**

3:     **for all**  $(v_i, v_j) \in E$  **do**

4:          $\pi(v_j) \leftarrow \min\{\pi(v_j), \pi(v_i) + w(v_i, v_j)\}$

5:     **end for**

6: **until** 没有  $\pi$  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

```
1:  $\pi(v_i) = +\infty, \pi(v_s) = 0$ 
2: 把  $v_s$  加入队列  $Q$ , 并标记  $v_s$ 
3: while  $Q$  is not empty do
4:    $v_i \leftarrow Q$  的队首元素
5:   弹出  $Q$  的队首元素
6:   for all  $(v_i, v_j) \in E$  do
7:     if  $\pi(v_i) + w(v_i, v_j) < \pi(v_j)$  then
8:        $\pi(v_j) \leftarrow \pi(v_i) + w(v_i, v_j)$ 
9:       if  $v_j$  未被标记 then
10:        把  $v_j$  加入队列  $Q$ , 并标记  $v_j$ 
11:       end if
12:     end if
13:   end for
14:   取消  $v_i$  标记
15: 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  $O(m \log n)$   
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 $a_1x_1 + a_2x_2 + \dots + a_Nx_N = B$ 存在非负整数解的条件。他想知道，给定 $N$ 、 $\{a_n\}$ 、以及 $B$ 的取值范围，有多少 $B$ 可以使等式存在非负整数解。

$$N \leq 12, \quad 0 \leq a_i \leq 5 \times 10^5, \quad 1 \leq B_{min} \leq B_{max} \leq 10^{12}$$

## 例 5. 解

设  $a_{min} = \min(a_i)$

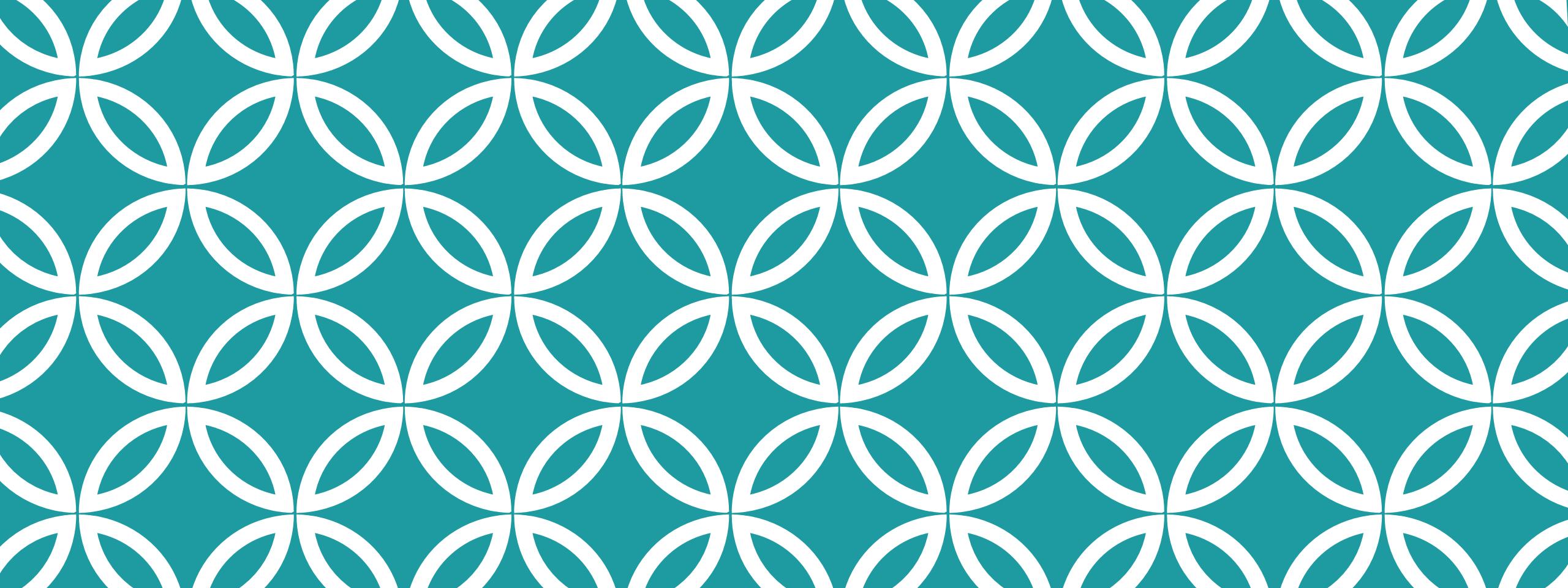
若  $b$  能被表示出，则  $b + a_{min}$  也能被表示出

只需要求出模  $a_{min}$  意义下最小能被表示出的数，就能算出答案

建  $a_{min}$  个点， $dis[i]$  分别表示模  $a_{min}$  意义下余  $i$  的最小能被表示出的数

对于每一个  $a_i$ ，从  $b$  向  $(x + a_i) \% a_{min}$  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 $i \rightarrow j:w$ , 最后一定满足 $dist[j] \leq dist[i] + w$

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 $x_i - x_j \geq a_k$ 对应边 $j \rightarrow i:a_k$

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 $x_i - x_S \geq 0$ 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

$x_i - x_j \leq a_k - 1$  (整数)  $x_i - x_j \leq a_k - \text{eps}$  (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

## 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  $-1$ 。

$n \leq 50,000$ ,  $m$  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 $x_l + x_{l+1} + \dots + x_r \geq k$ ，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 $0 \leq x_i \leq t_i$

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 $\sum_{j=0}^7 x_{i-j} \geq R_i$

同样定义 $S[]$ 为 $x[]$ 的前缀和, 那么式子变为:

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

当然还有 $0 \leq S[i] - S[i-1] \leq t_i$

## 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 $U$  ( $L \leq U$ ) , 问是否存在两个向量 $A[]$ 和 $B[]$ 满足 $\forall i \in [1, N], j \in [1, M]$ , 有 $L \leq X[i][j] \times \frac{A[i]}{B[j]} \leq U$ 。

$N, M \leq 400$ 。

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？

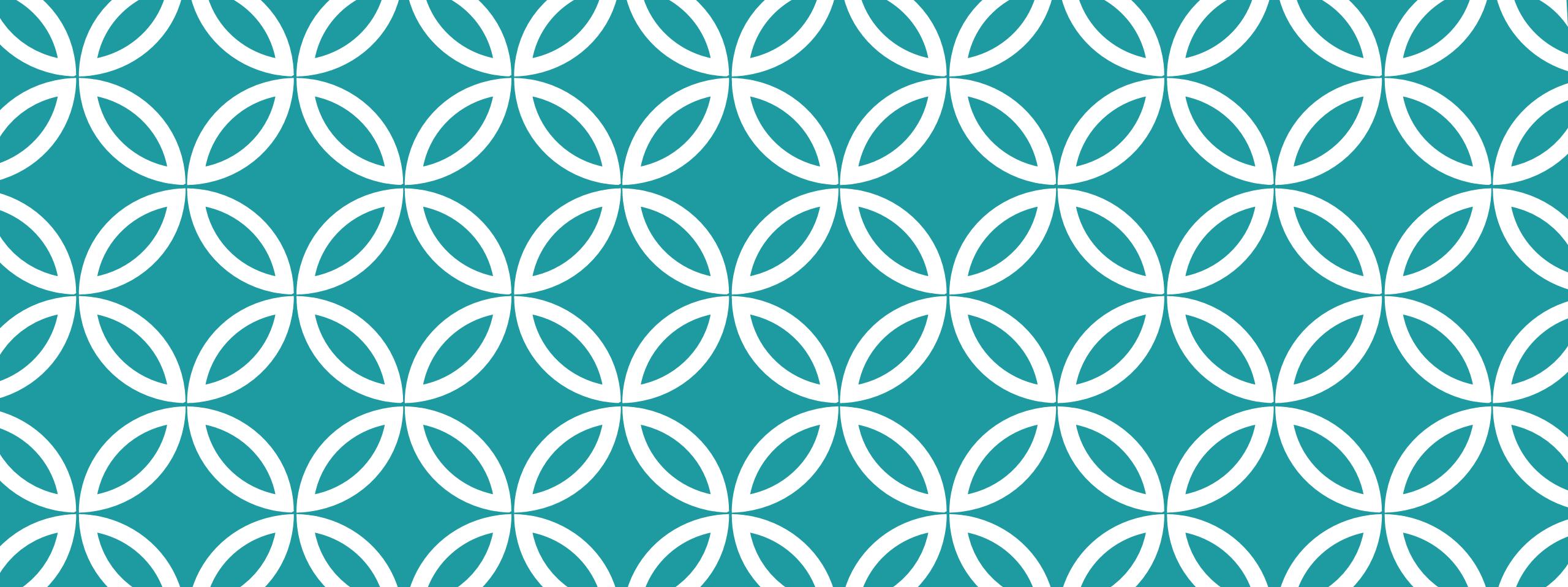
取对数！

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

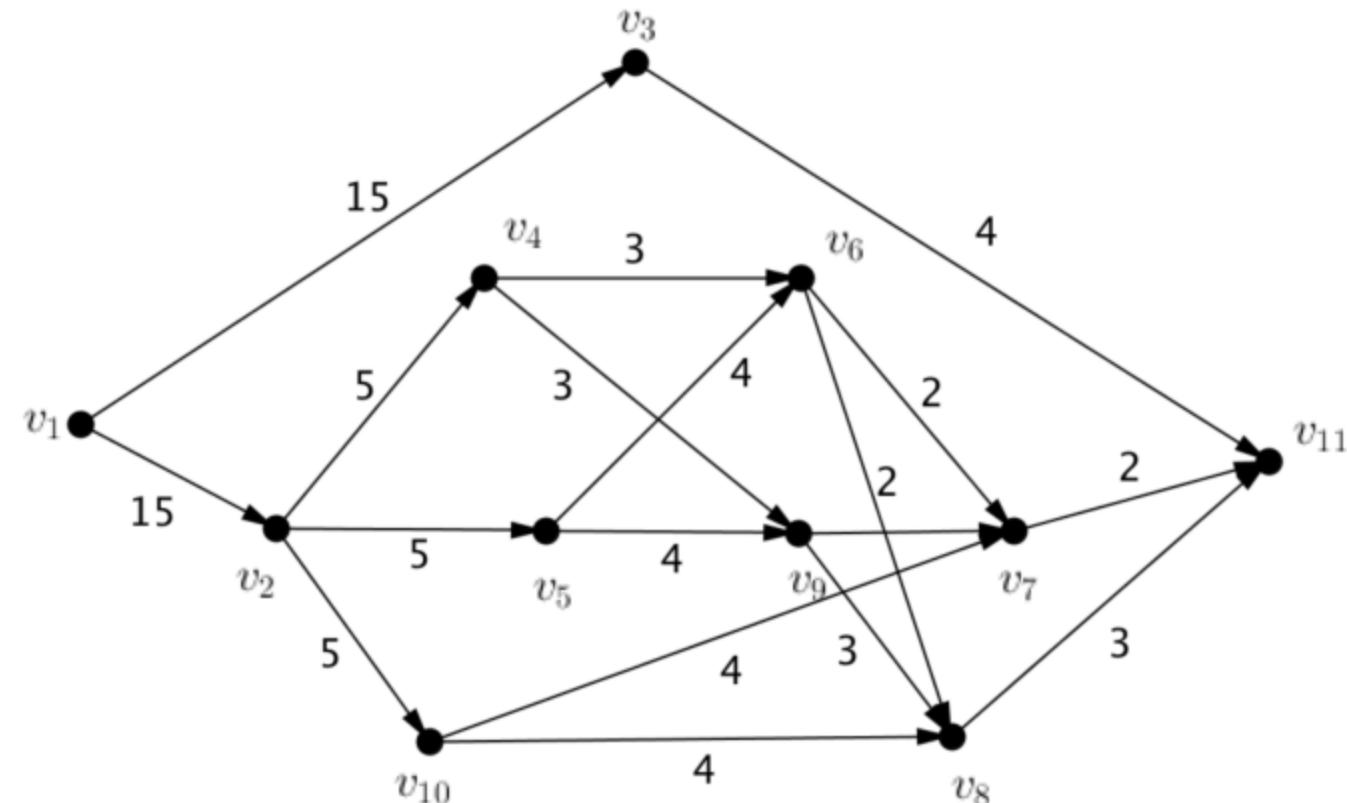
# PT图

| 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

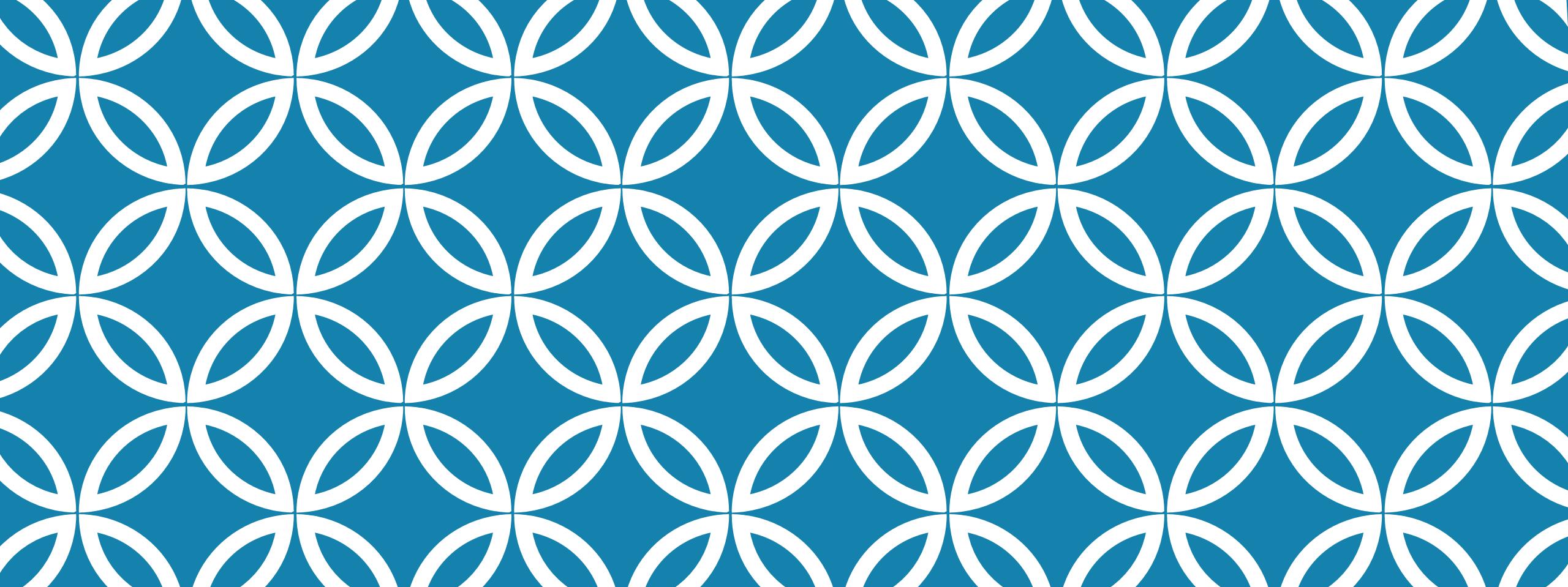
每项工序的最早开始时间：到每个点的最长路

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》



**THANKS FOR YOUR LISTENING**