



南京大學
NANJING UNIVERSITY



第3章 圈和遍历

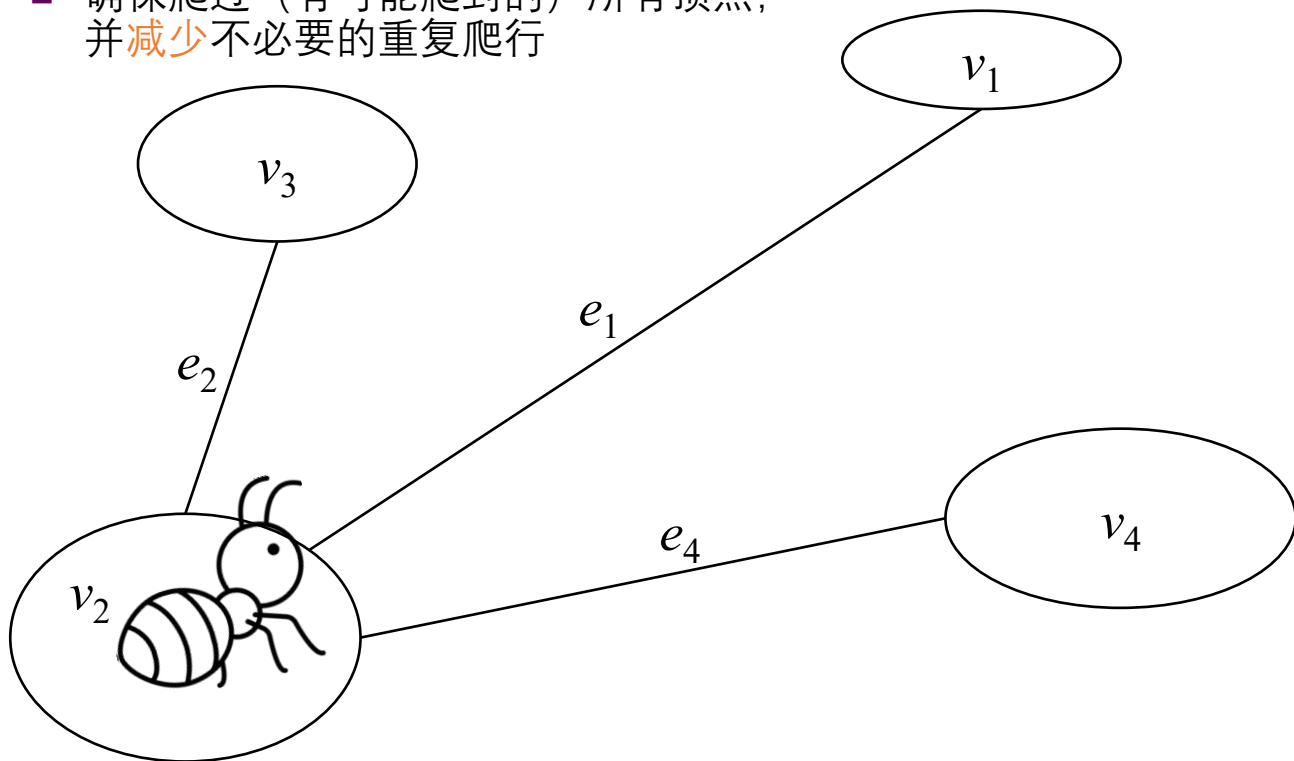
程龚

为什么专门讨论“圈”

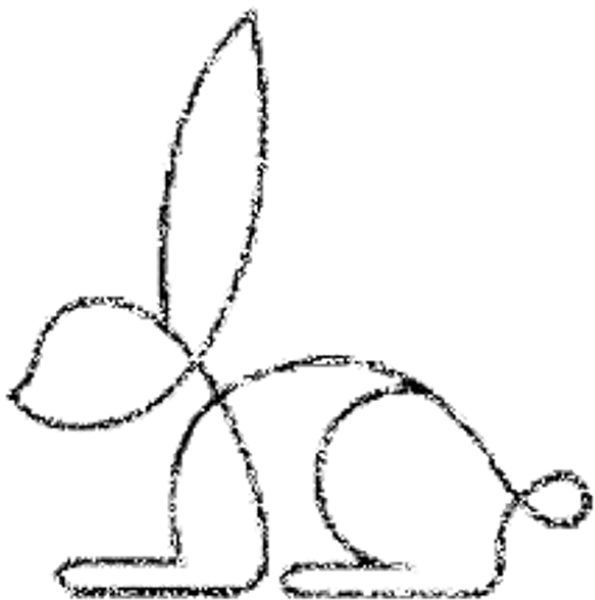
- 圈是图结构复杂性的主要表现之一
- 一些图论中的难问题在不含圈的图上较容易解决

上节课讨论的“遍历”

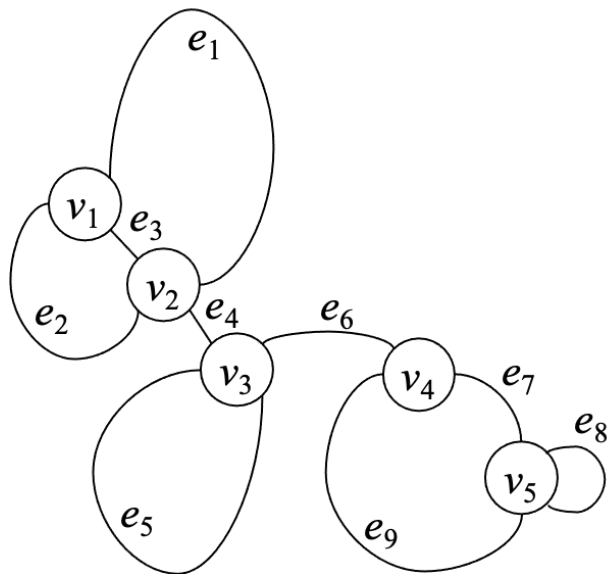
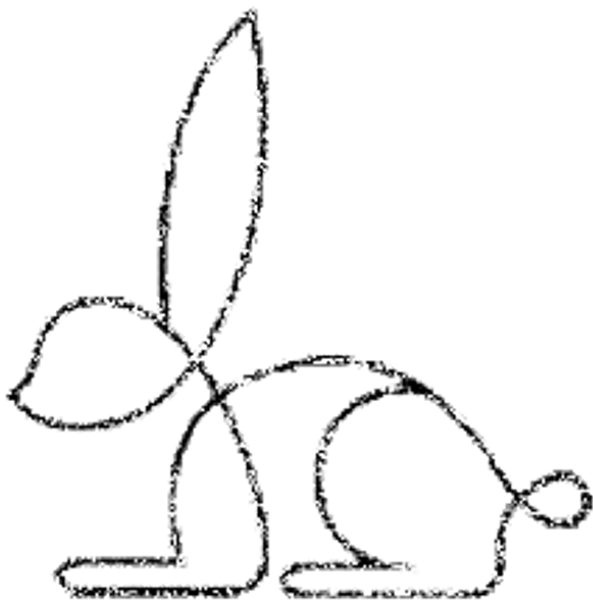
- 确保爬过（有可能爬到的）所有顶点，并减少不必要的重复爬行



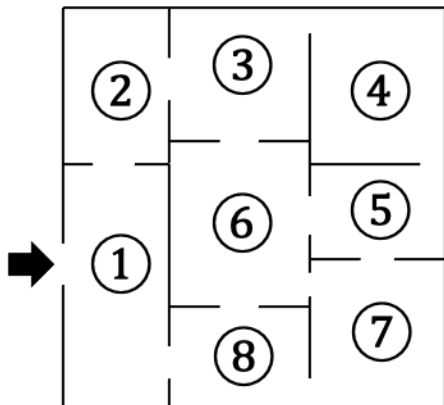
本次课讨论更严格的“遍历”



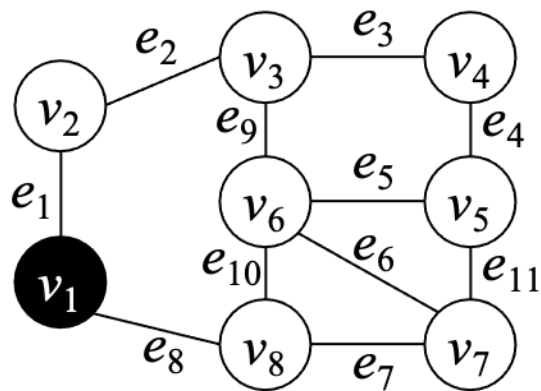
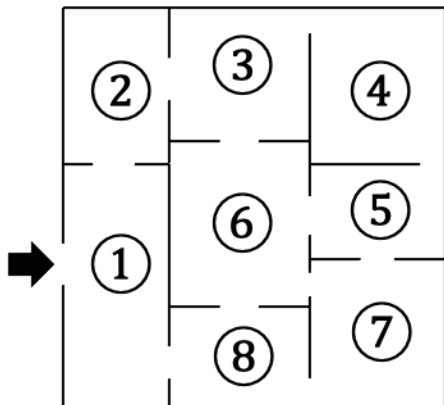
本次课讨论更严格的“遍历”



本次课讨论更严格的“遍历”



本次课讨论更严格的“遍历”



本次课的主要内容

3.1 圈和树

3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

本次课的主要内容

3.1 圈和树

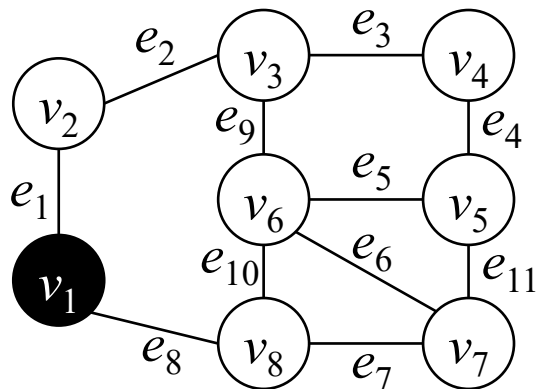
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

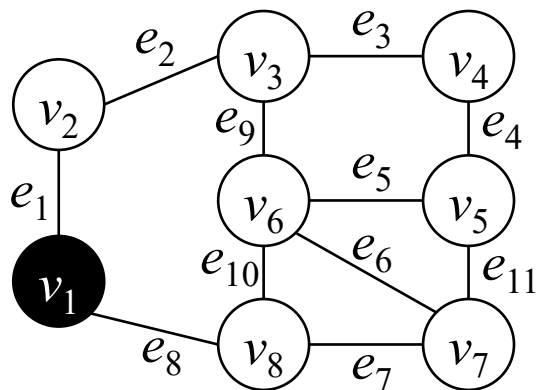
圈和树

- **闭路线**：起点和终点相同的非平凡路线



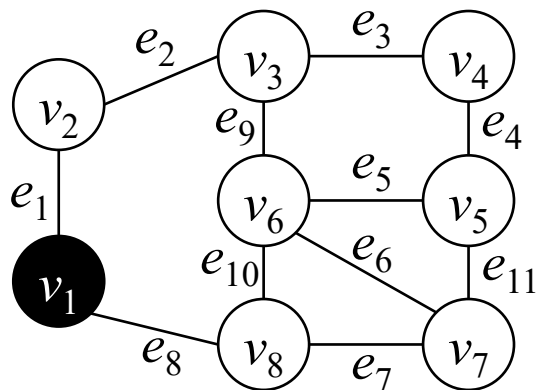
圈和树

- 闭路线：起点和终点相同的非平凡路线
- **闭迹**（回路）：边不重复出现的闭路线



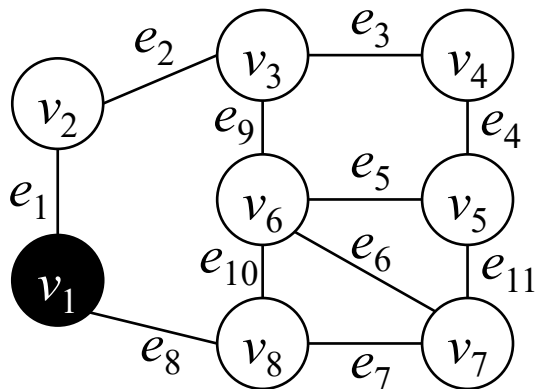
圈和树

- 闭路线：起点和终点相同的非平凡路线
- 闭迹（回路）：边不重复出现的闭路线
- **圈**：顶点不重复出现（除起点和终点相同）的闭迹



圈和树

- 闭路线：起点和终点相同的非平凡路线
 - 闭迹（回路）：边不重复出现的闭路线
 - 圈：顶点不重复出现（除起点和终点相同）的闭迹
-
- 若图中存在闭路线，一定存在闭迹吗？
 - 若图中存在闭迹，一定存在圈吗？

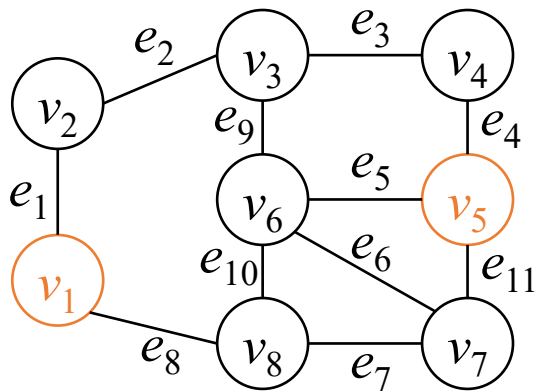


圈和树

- 闭路线：起点和终点相同的非平凡路线
- 闭迹（回路）：边不重复出现的闭路线
- 圈：顶点不重复出现（除起点和终点相同）的闭迹

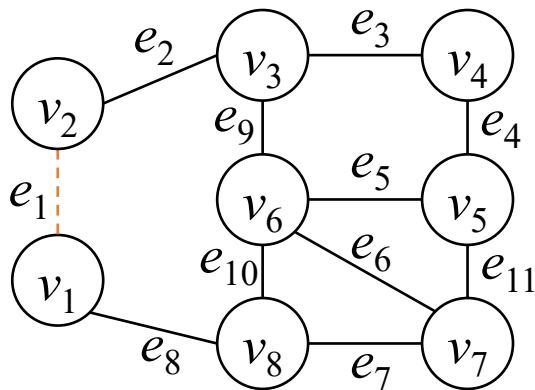
- 若图中存在闭路线，一定存在闭迹吗？
- 若图中存在闭迹，一定存在圈吗？

- 含顶点 u 和 v 的圈一定由两条 $u-v$ 路组成吗？
- 两条 $u-v$ 路一定能组成圈吗？



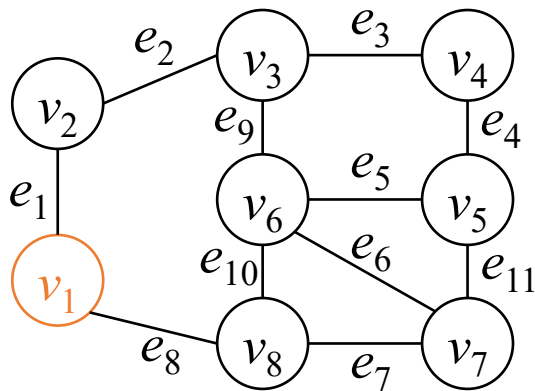
圈和树

- 对于图 $G = \langle V, E \rangle$ 和边 $e \in E$, e 是 G 的割边当且仅当 e 不在任何圈中。



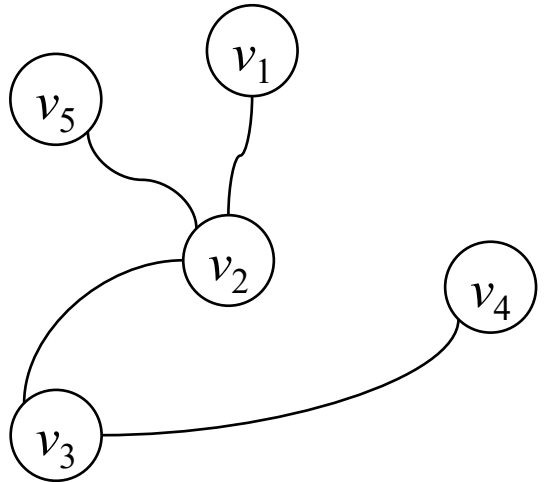
圈和树

- 对于图 $G = \langle V, E \rangle$ 和边 $e \in E$, e 是 G 的割边当且仅当 e 不在任何圈中。
- 对于图 $G = \langle V, E \rangle$ 和顶点 $v \in V$, v 是 G 的割点当且仅当 v 不在任何圈中。这个结论成立吗？



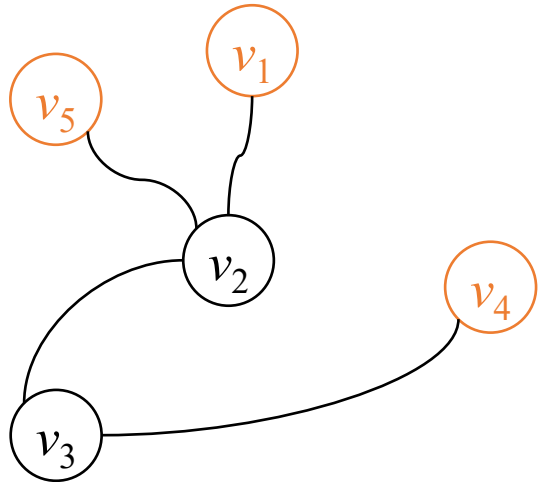
圈和树

- 树：不含圈的连通图



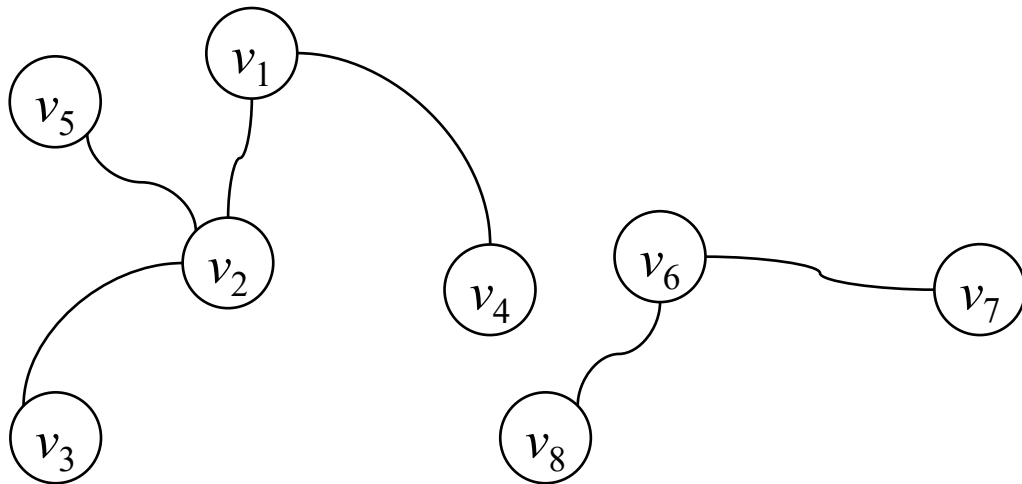
圈和树

- 树：不含圈的连通图
- **叶顶点**：树中度为1的顶点



圈和树

- 树：不含圈的连通图
- 叶顶点：树中度为1的顶点
- **森林**：不含圈的图



圈和树

■ 树的等价定义：

- 图 G 连通且不含圈。
- 图 G 中任意两个顶点间有且只有一条路。
- 图 G 不含圈且 $\varepsilon(G) = v(G) - 1$ 。
- 图 G 连通且 $\varepsilon(G) = v(G) - 1$ 。
- 图 G 极小连通，即： G 连通，但删除任意一条边均不连通。
- 图 G 极大无圈，即： G 不含圈，但增加任意一条边均形成圈。

圈和树

- 树的等价定义：
 - 图 G 连通且不含圈。
 - 图 G 中任意两个顶点间有且只有一条路。
 - 图 G 不含圈且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 连通且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 极小连通，即： G 连通，但删除任意一条边均不连通。
 - 图 G 极大无圈，即： G 不含圈，但增加任意一条边均形成圈。
- 树中的边有什么特征？你能就此给出树的另一种等价定义吗？

圈和树

- 树的等价定义：
 - 图 G 连通且不含圈。
 - 图 G 中任意两个顶点间有且只有一条路。
 - 图 G 不含圈且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 连通且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 极小连通，即： G 连通，但删除任意一条边均不连通。
 - 图 G 极大无圈，即： G 不含圈，但增加任意一条边均形成圈。
- 树中的边有什么特征？你能就此给出树的另一种等价定义吗？
 - 图 G 连通且每条边都是割边

圈和树

- 树的等价定义：
 - 图 G 连通且不含圈。
 - 图 G 中任意两个顶点间有且只有一条路。
 - 图 G 不含圈且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 连通且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 极小连通，即： G 连通，但删除任意一条边均不连通。
 - 图 G 极大无圈，即： G 不含圈，但增加任意一条边均形成圈。
- 树中的边有什么特征？你能就此给出树的另一种等价定义吗？
 - 图 G 连通且每条边都是割边
- 非平凡树的叶顶点数量的上界和下界分别是多少？

圈和树

- 树的等价定义：
 - 图 G 连通且不含圈。
 - 图 G 中任意两个顶点间有且只有一条路。
 - 图 G 不含圈且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 连通且 $\varepsilon(G) = v(G) - 1$ 。
 - 图 G 极小连通，即： G 连通，但删除任意一条边均不连通。
 - 图 G 极大无圈，即： G 不含圈，但增加任意一条边均形成圈。
- 树中的边有什么特征？你能就此给出树的另一种等价定义吗？
 - 图 G 连通且每条边都是割边
- 非平凡树的叶顶点数量的上界和下界分别是多少？
- **生成树**：连通图的生成子图，且是树

圈和树

- 如何判定一个图是否为树?

本次课的主要内容

3.1 圈和树

3.2 二分图

3.3 欧拉图

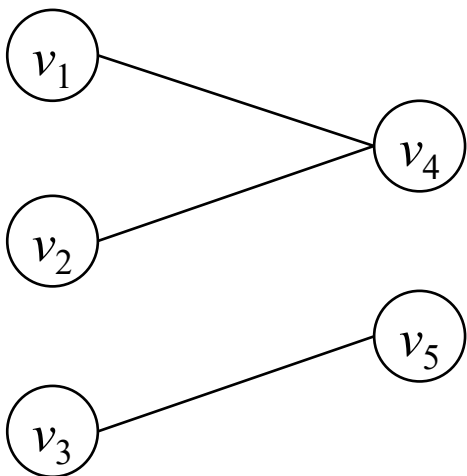
3.4 哈密尔顿图

二分图

- **奇圈**：长度为奇数的圈
- **偶圈**：长度为偶数的圈

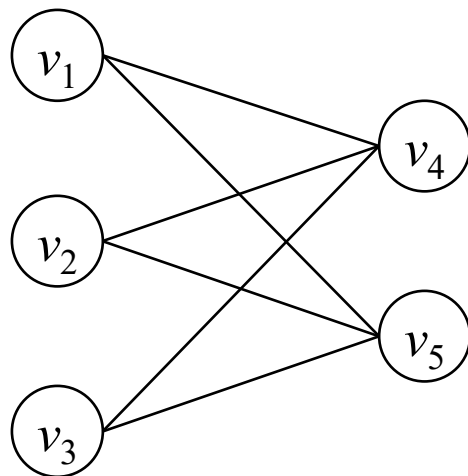
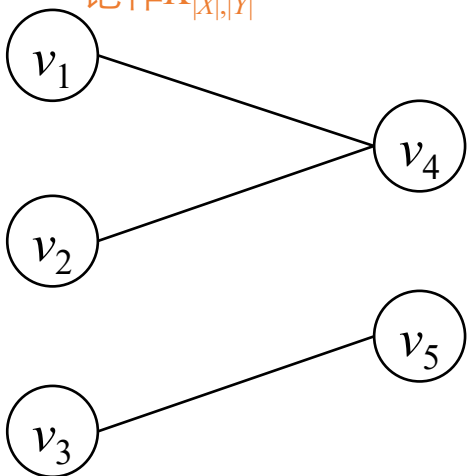
二分图

- 奇圈：长度为奇数的圈
- 偶圈：长度为偶数的圈
- **二分图**：对于图 $G = \langle V, E \rangle$ ， V 可划分为两个子集 X, Y ，使每条边 $e \in E$ 的两个端点分属于 X 和 Y



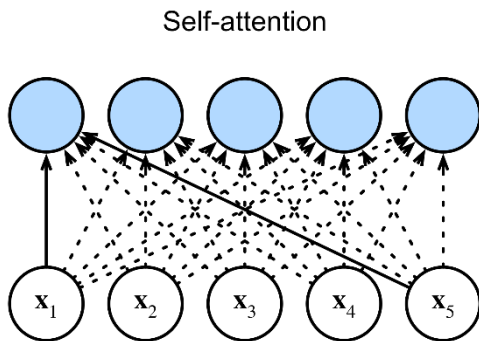
二分图

- 奇圈：长度为奇数的圈
- 偶圈：长度为偶数的圈
- 二分图：对于图 $G = \langle V, E \rangle$ ， V 可划分为两个子集 X, Y ，使每条边 $e \in E$ 的两个端点分属于 X 和 Y
- **完全二分图**：二分图，且 X 中每个顶点和 Y 中每个顶点都相邻，记作 $K_{|X|,|Y|}$



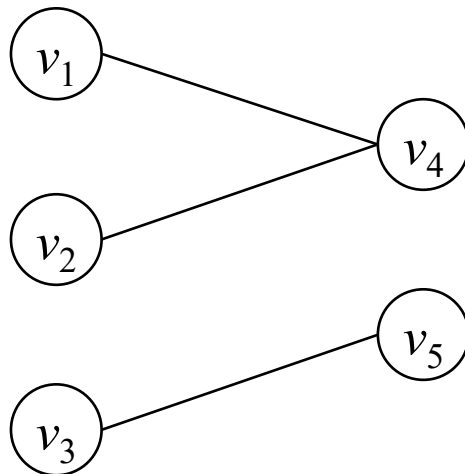
二分图

- 更严格地说，Transformer的self-attention本质是完全二分图
 - Query tokens
 - Key tokens



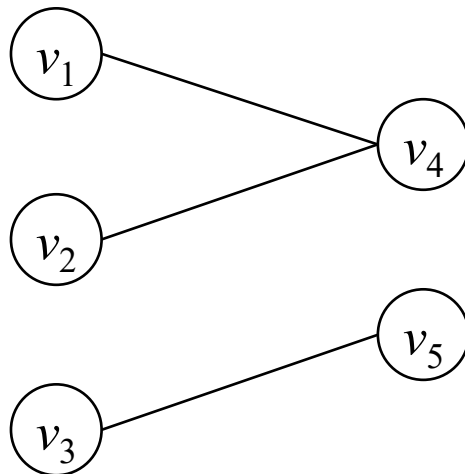
二分图

- 平凡图是二分图吗？



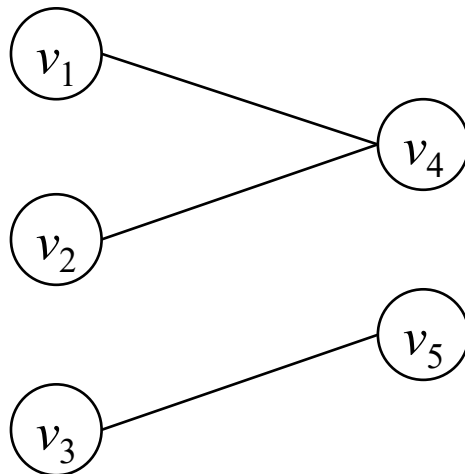
二分图

- 平凡图是二分图吗？
- 非平凡图 G 是二分图当且仅当 G 不含奇圈。



二分图

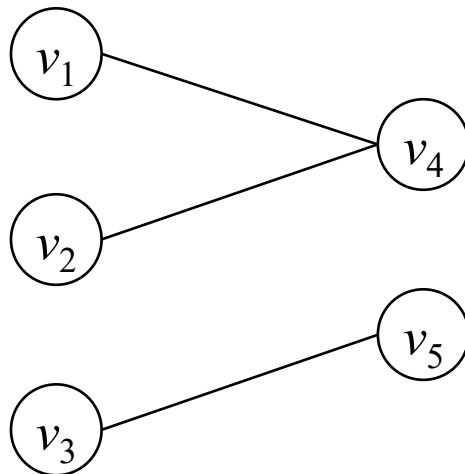
- 平凡图是二分图吗？
- 非平凡图 G 是二分图当且仅当 G 不含奇圈。
- 树是二分图吗？森林呢？



二分图

- 平凡图是二分图吗？
- 非平凡图 G 是二分图当且仅当 G 不含奇圈。
- 树是二分图吗？森林呢？
- 对于二分图 $G = \langle V, E \rangle$
 - 若 G 连通，则 V 的划分方式唯一吗？
 - 若 V 的划分方式唯一，则 G 一定连通吗？

随堂小测



二分图

- 如何判定一个图是否为二分图?

二分图

■ 扩展DFS算法

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false,

```
1  $u.visited \leftarrow true;$   
2 foreach  $(u, v) \in E$  do  
3   if  $v.visited = false$  then  
4     |  
5     |  $DFSbpt(G, v);$   
6     |  
7     |
```

二分图

- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart
初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSBpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```

二分图

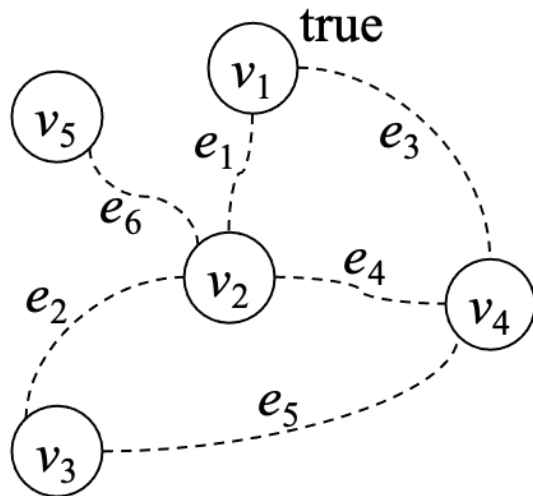
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart 初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSBpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```



二分图

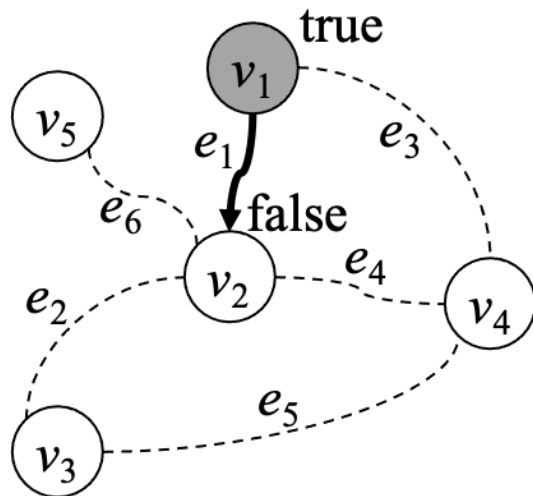
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 `visited` 初值为 `false`, 出发点的 `samePart` 初值为 `true`、其它顶点的 `samePart` 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSbpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```



二分图

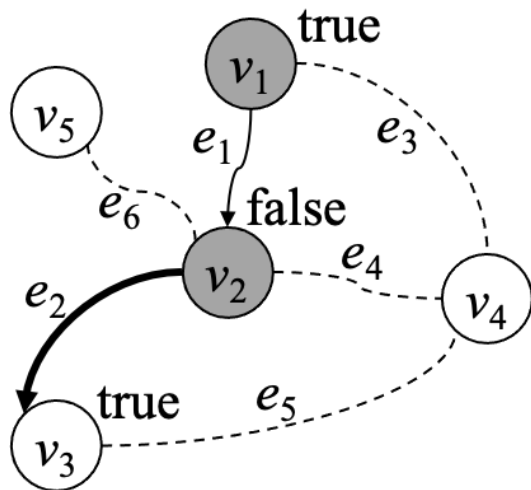
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart 初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$   
2 foreach  $(u, v) \in E$  do  
3   if  $v.visited = false$  then  
4      $v.samePart \leftarrow \neg u.samePart;$   
5      $DFSBpt(G, v);$   
6   else if  $u.samePart = v.samePart$  then  
7     判定 (“ $G$  非二分图”)
```



二分图

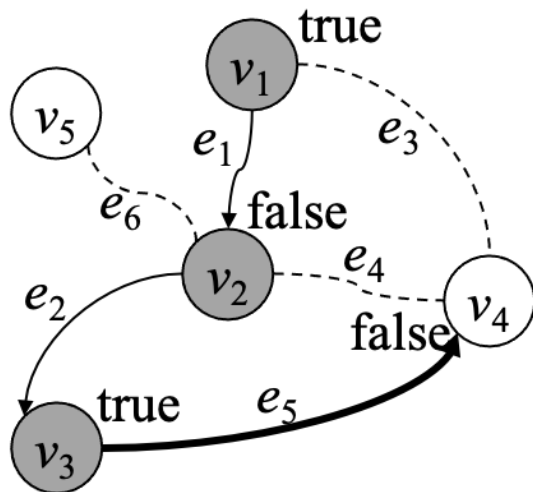
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart 初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$   
2 foreach  $(u, v) \in E$  do  
3   if  $v.visited = false$  then  
4      $v.samePart \leftarrow \neg u.samePart;$   
5      $DFSBpt(G, v);$   
6   else if  $u.samePart = v.samePart$  then  
7     判定 (“ $G$  非二分图”)
```



二分图

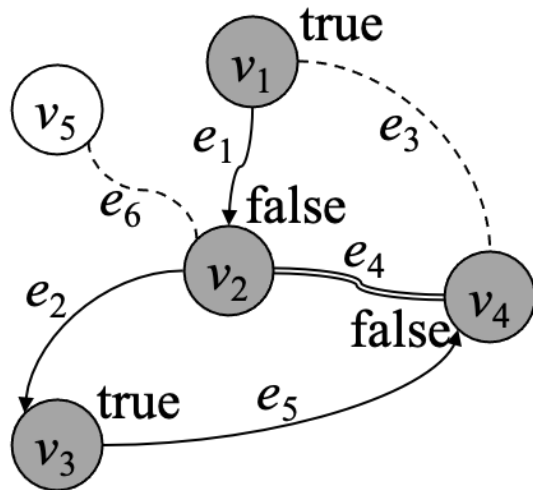
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 `visited` 初值为 `false`, 出发点的 `samePart` 初值为 `true`、其它顶点的 `samePart` 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSbpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```



二分图

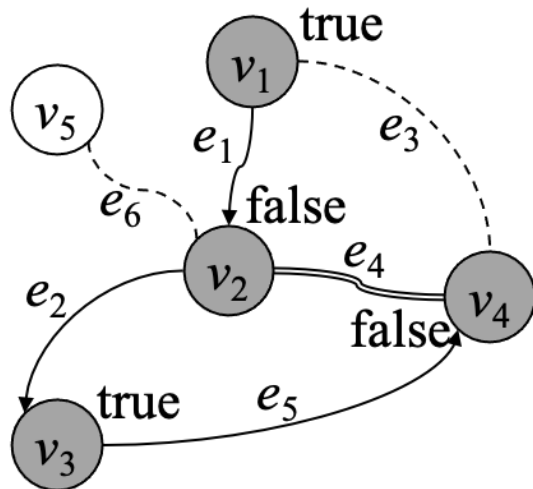
- 扩展DFS算法：尝试将每个顶点分到X或Y中
- 在DFSbpt算法中，树边和后向边的作用分别是什么？

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart 初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$   
2 foreach  $(u, v) \in E$  do  
3   if  $v.visited = false$  then  
4      $v.samePart \leftarrow \neg u.samePart;$   
5      $DFSbpt(G, v);$   
6   else if  $u.samePart = v.samePart$  then  
7     判定 (“ $G$  非二分图”)
```



二分图

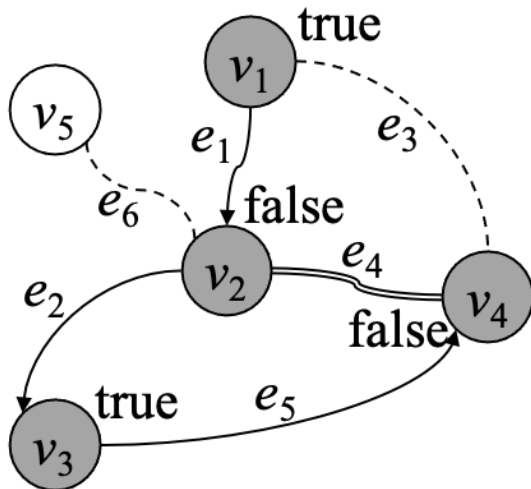
- 扩展DFS算法：尝试将每个顶点分到X或Y中
- 在DFSBpt算法中，树边和后向边的作用分别是什么？
 - 树边：二分顶点
 - 后向边：判定奇圈

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 `visited` 初值为 `false`, 出发点的 `samePart` 初值为 `true`、其它顶点的 `samePart` 初值未定义

```
1  $u.visited \leftarrow true$ ;  
2 foreach  $(u, v) \in E$  do  
3   if  $v.visited = false$  then  
4      $v.samePart \leftarrow \neg u.samePart$ ;  
5      $DFSBpt(G, v)$ ;  
6   else if  $u.samePart = v.samePart$  then  
7     判定 (“ $G$  非二分图”)
```



二分图

- 扩展DFS算法：尝试将每个顶点分到X或Y中
- 在DFSBpt算法中，树边和后向边的作用分别是什么？
 - 树边：二分顶点
 - 后向边：判定奇圈

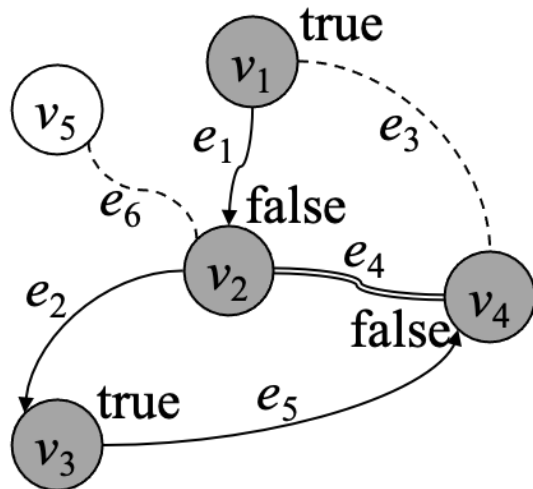
算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 `visited` 初值为 `false`, 出发点的 `samePart` 初值为 `true`、其它顶点的 `samePart` 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSBpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```

- 为什么判定结果一定正确？



二分图

- 时间复杂度: $O(n + m)$

算法 3.1: DFSBpt

输入: 非平凡连通图 $G = \langle V, E \rangle$, 顶点 u

初值: V 中所有顶点的 visited 初值为 false, 出发点的 samePart 初值为 true、其它顶点的 samePart 初值未定义

```
1  $u.visited \leftarrow true;$ 
2 foreach  $(u, v) \in E$  do
3   if  $v.visited = false$  then
4      $v.samePart \leftarrow \neg u.samePart;$ 
5      $DFSBpt(G, v);$ 
6   else if  $u.samePart = v.samePart$  then
7     判定 (“ $G$  非二分图”)
```

本次课的主要内容

3.1 圈和树

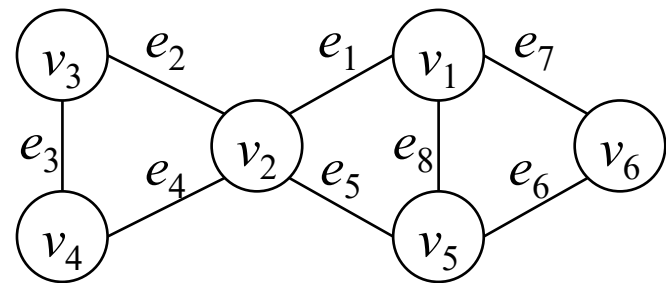
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

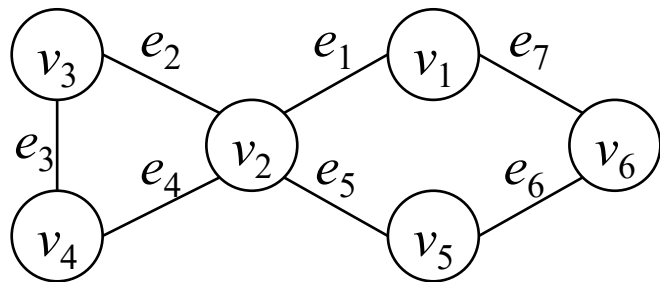
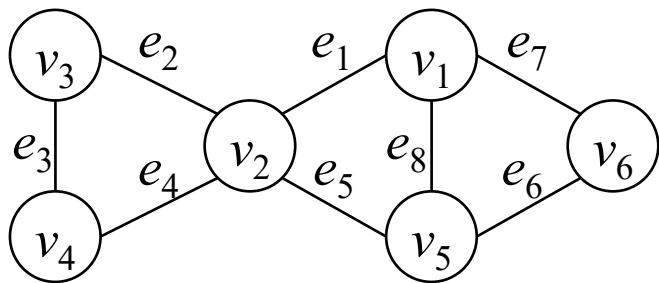
欧拉图

- **欧拉迹**：经过图的每条边恰一次的迹



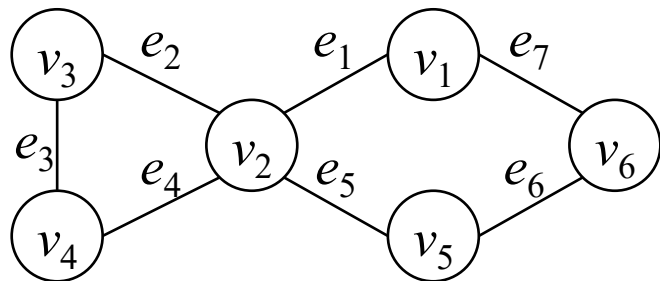
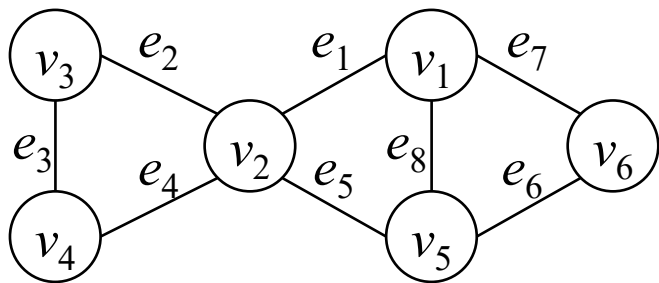
欧拉图

- 欧拉迹：经过图的每条边恰一次的迹
- **欧拉回路**：经过图的每条边恰一次的闭迹（回路）



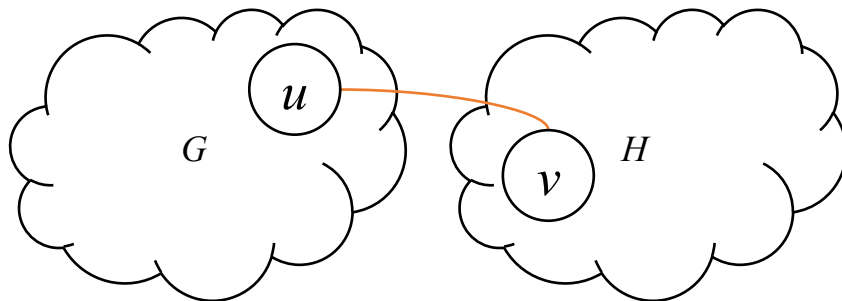
欧拉图

- 欧拉迹：经过图的每条边恰一次的迹
- 欧拉回路：经过图的每条边恰一次的闭迹（回路）
- **欧拉图**：含欧拉回路的图



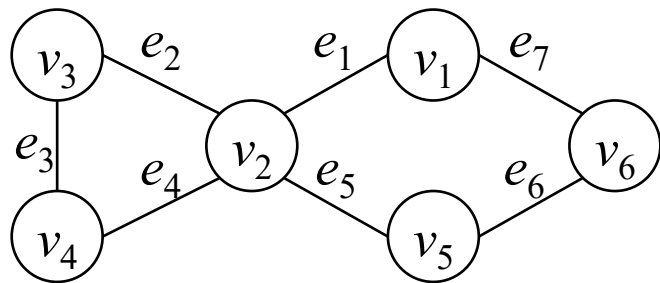
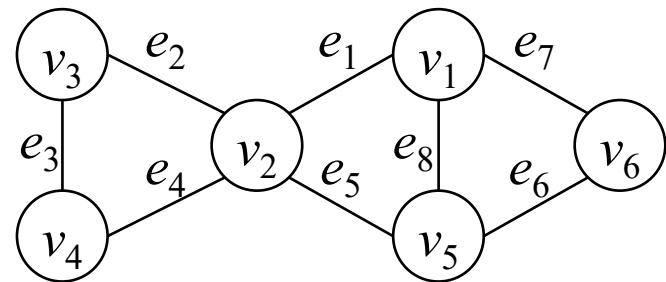
欧拉图

- 对于连通欧拉图 $G = \langle V_G, E_G \rangle$ 和 $H = \langle V_H, E_H \rangle$, 任取顶点 $u \in V_G$ 和 $v \in V_H$, 向 G 和 H 的不交并 $G + H$ 中增加边 (u, v) 形成的图
 - 含欧拉回路吗?
 - 含欧拉迹吗?



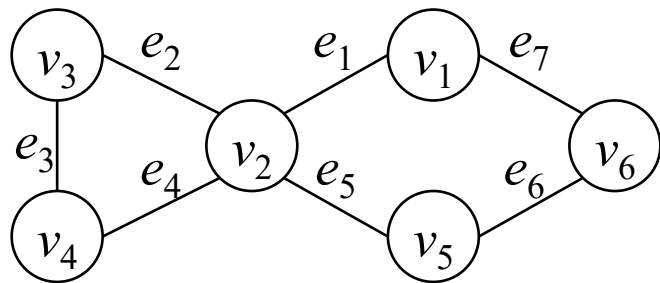
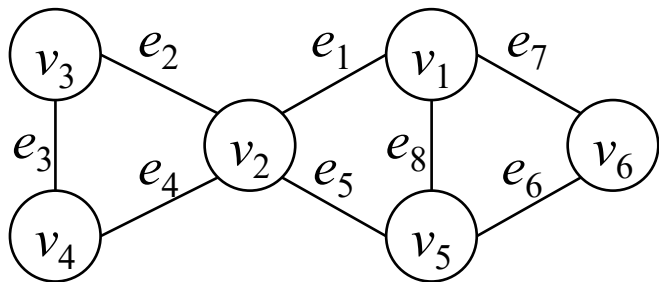
欧拉图

- 非空连通图 G 含欧拉回路当且仅当 G 没有顶点的度为奇数。



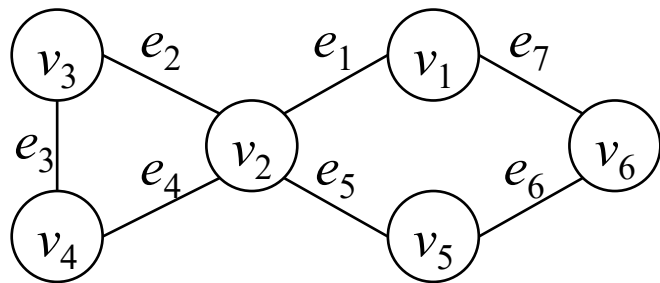
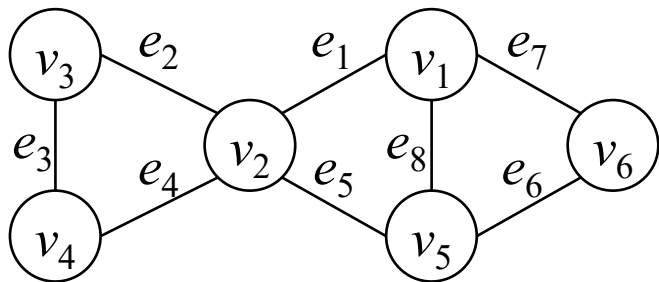
欧拉图

- 非空连通图 G 含欧拉回路当且仅当 G 没有顶点的度为奇数。
- 非空连通图 G 含欧拉迹当且仅当 G 有至多2个顶点的度为奇数。



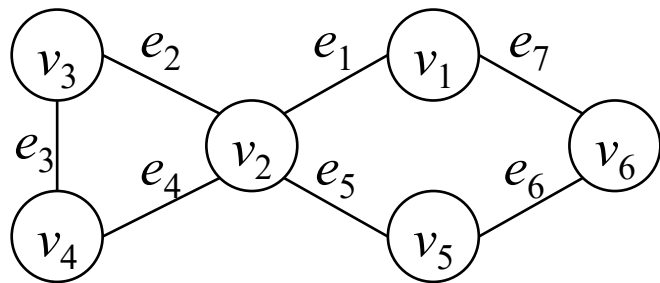
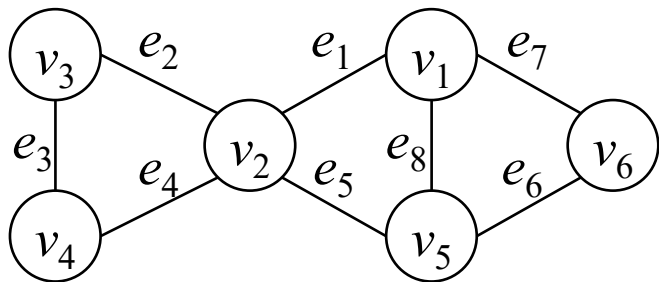
欧拉图

- 非空连通图 G 含欧拉回路当且仅当 G 没有顶点的度为奇数。
- 非空连通图 G 含欧拉迹当且仅当 G 有至多2个顶点的度为奇数。
- 含欧拉迹的图一定连通吗？
图 G 含欧拉回路和欧拉迹的充要条件分别是什么？



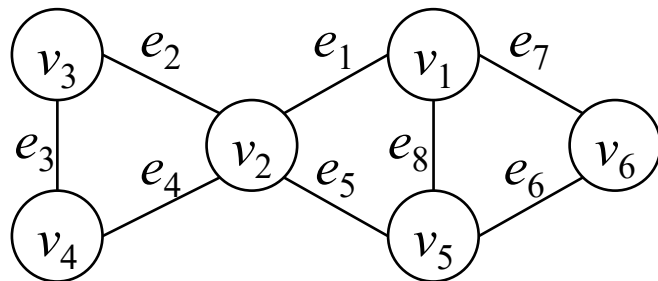
欧拉图

- 非空连通图 G 含欧拉回路当且仅当 G 没有顶点的度为奇数。
- 非空连通图 G 含欧拉迹当且仅当 G 有至多2个顶点的度为奇数。
- 含欧拉迹的图一定连通吗？
图 G 含欧拉回路和欧拉迹的充要条件分别是什么？
 - 含欧拉回路的额外条件：边集的边导出子图非空连通
 - 含欧拉迹的额外条件：是空图 或 边集的边导出子图非空连通



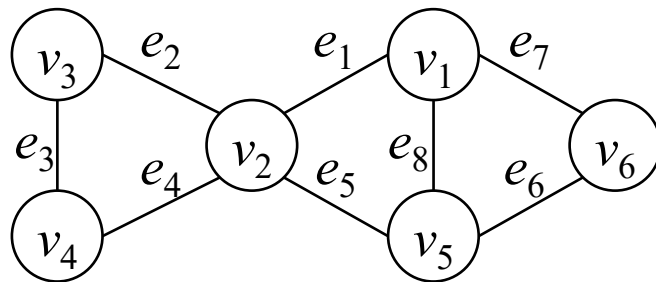
欧拉图

- 如何找出图中的一条欧拉迹?



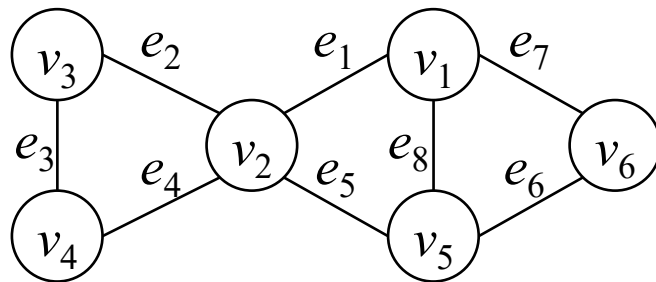
欧拉图

- 弗勒里算法
- 希尔霍尔策算法



欧拉图

- 弗勒里算法：尽可能避免选择割边
- 希尔霍尔策算法



欧拉图

- Fleury, 全名和真实身份尚存在争议

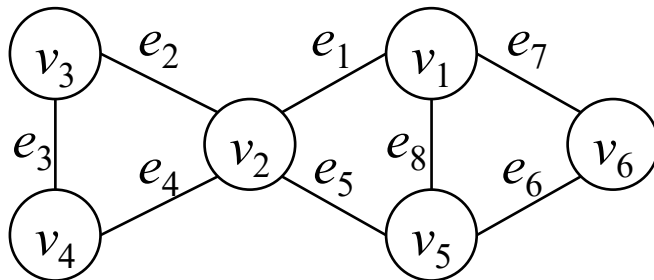
欧拉图

■ 出发点如何选择?

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6
7
8
9
10
11
12
13
14
```



欧拉图

■ 出发点如何选择？

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 **if** $\exists v \in V$, $d(v)$ 是奇数 **then**

2 | $u \leftarrow v$;

3 **else**

4 | $u \leftarrow V$ 中任意一个非孤立点;

5 输出 (u);

6

7

8

9

10

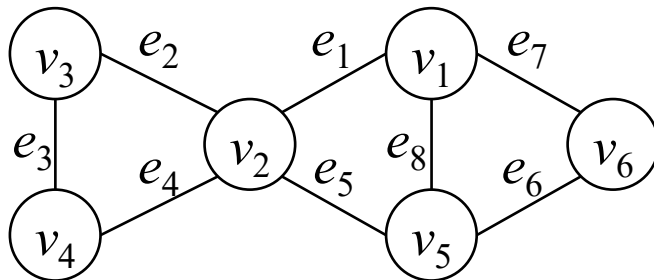
11

12

13

14

奇度顶点,
如果有的话



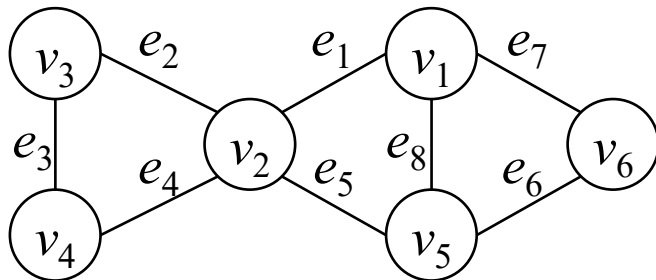
欧拉图

- 接下来，总是优先选择非割边

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14
```



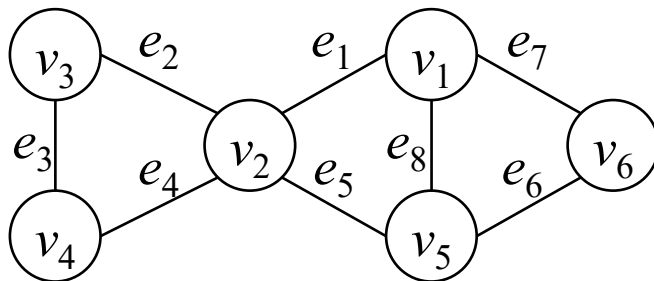
欧拉图

■ 删除经过的边

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```



欧拉图

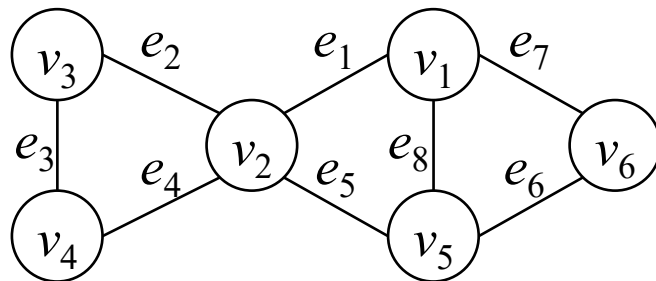
■ 删除经过的边

- 因此，每轮循环需要重新计算割边

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```

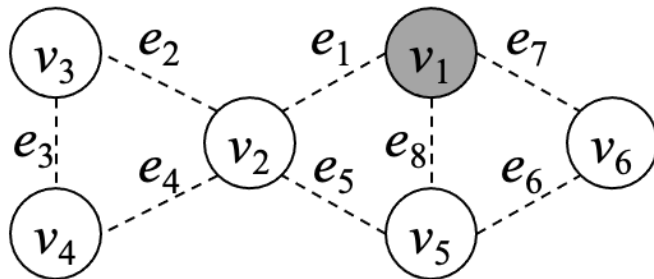


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```

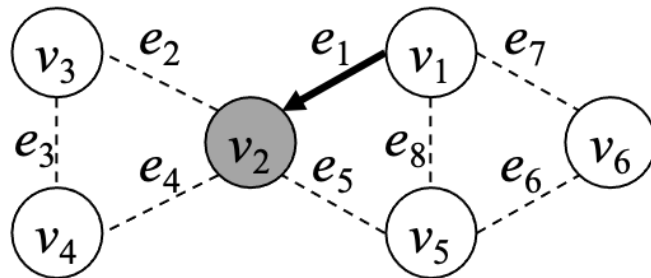


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```

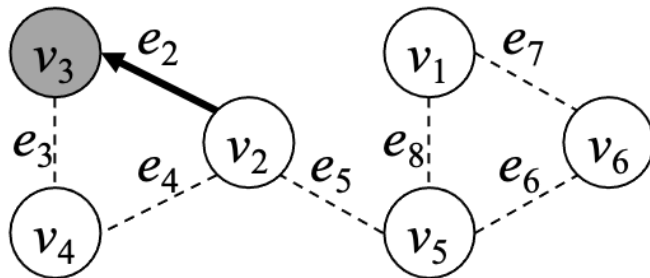


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```

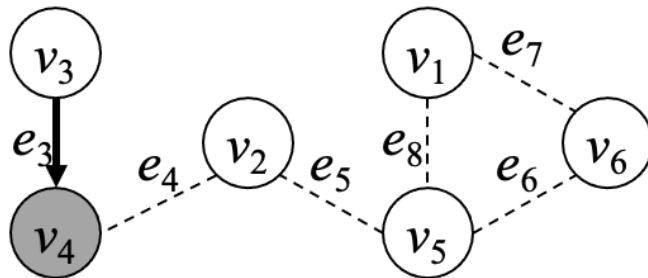


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```

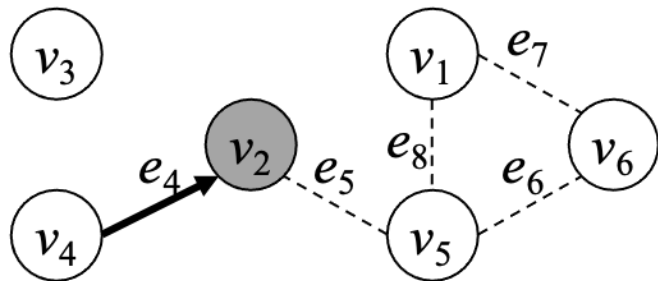


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```

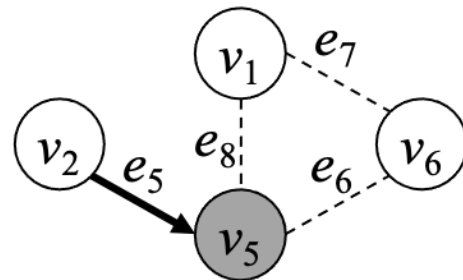


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```

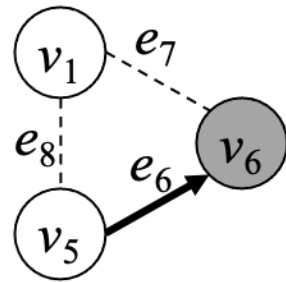


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```

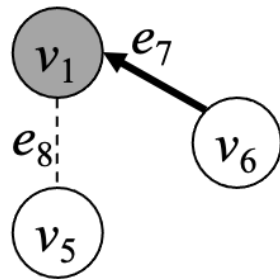


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```

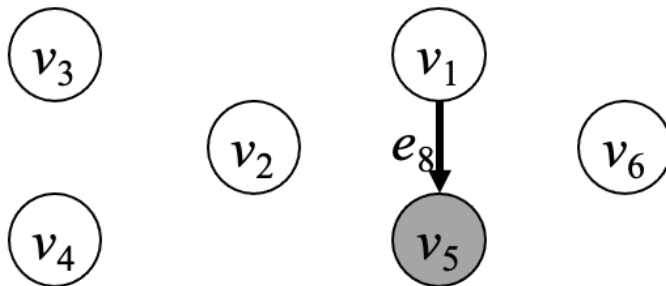


欧拉图

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then  
2   |  $u \leftarrow v$ ;  
3 else  
4   |  $u \leftarrow V$  中任意一个非孤立点;  
5 输出 ( $u$ );  
6 while  $u$  非孤立点 do  
7   | if  $u$  关联一条非割边  $e'$  then  
8     |  $e \leftarrow e'$ ;  
9   | else  
10    |  $e \leftarrow u$  关联的任意一条边;  
11    输出 ( $e$ );  
12     $u \leftarrow e$  的另一个端点;  
13    输出 ( $u$ );  
14     $G \leftarrow G - e$ ;
```



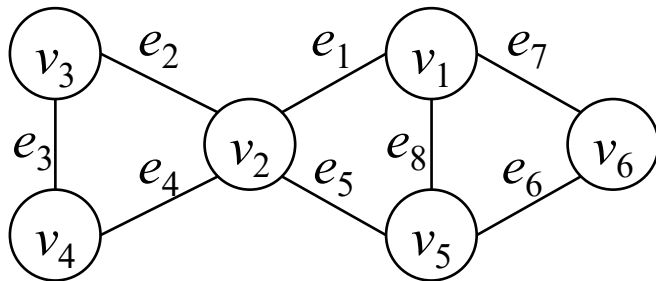
欧拉图

- 正确性：每一轮while循环条件判定前
 - 或者 G 为空图，则原图的欧拉迹已找到
 - 或者 G 含以 u 为起点的欧拉迹，则该欧拉迹和已输出的序列可拼接得到原图的欧拉迹

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```



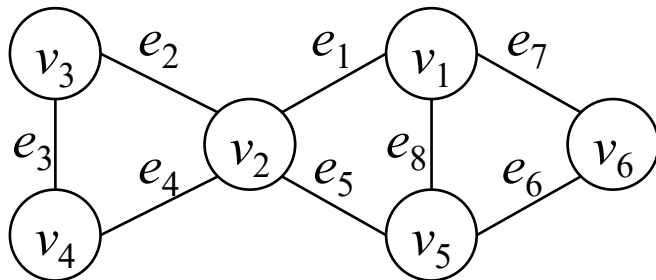
欧拉图

- 弗勒里算法每一轮while循环条件判定前, G 或者没有顶点的度为奇数, 或者恰有2个顶点 (包括 u) 的度为奇数。
- 弗勒里算法每一轮while循环条件判定前, G 或者为空图, 或者边集的边导出子图连通且含 u 。

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```



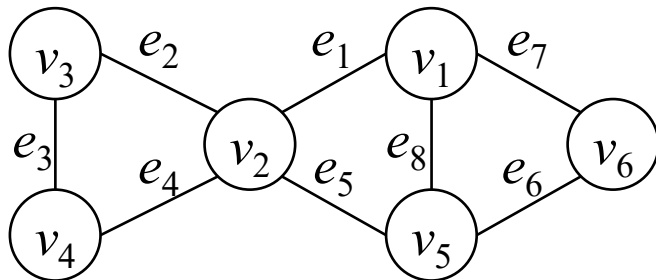
欧拉图

- 弗勒里算法每一轮while循环条件判定前， G 或者没有顶点的度为奇数，或者恰有2个顶点（包括 u ）的度为奇数。
- 弗勒里算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图连通且含 u 。

算法 3.2: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```



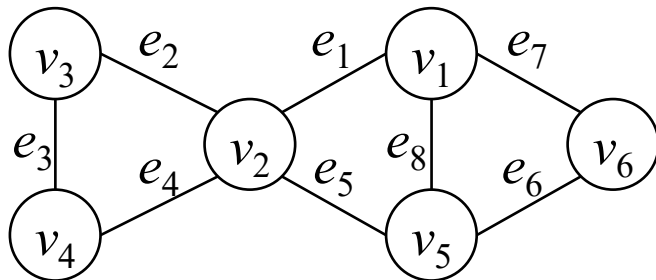
欧拉图

- 弗勒里算法每一轮while循环条件判定前， G 或者没有顶点的度为奇数，或者恰有2个顶点（包括 u ）的度为奇数。
- 弗勒里算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图连通且含 u 。

算法 3.2: 弗勒里算法

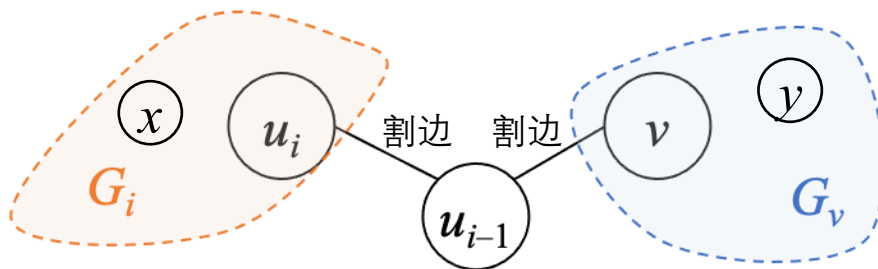
输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11    输出 ( $e$ );
12     $u \leftarrow e$  的另一个端点;
13    输出 ( $u$ );
14     $G \leftarrow G - e$ ;
```



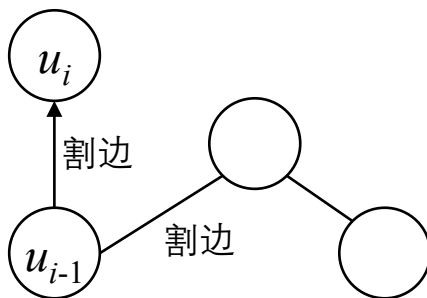
欧拉图

- 弗勒里算法每一轮while循环条件判定前， G 或者没有顶点的度为奇数，或者恰有2个顶点（包括 u ）的度为奇数。
- 弗勒里算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图**连通**且含 u 。
 - 假设第 i 轮首次不连通，则第 $i-1$ 轮与上一条结论矛盾



欧拉图

- 弗勒里算法每一轮while循环条件判定前， G 或者没有顶点的度为奇数，或者恰有2个顶点（包括 u ）的度为奇数。
- 弗勒里算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图连通且含 u 。
 - 假设第 i 轮首次不含 u ，会导致同样的矛盾



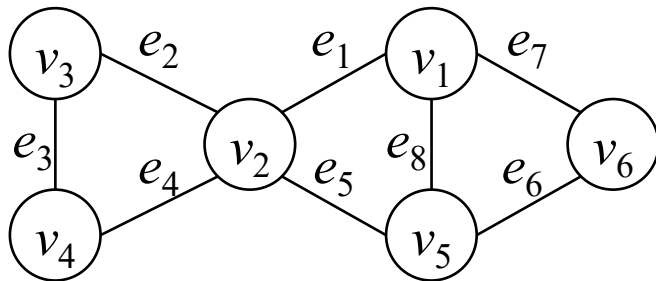
欧拉图

- 时间复杂度: $O(n + m(n + m))$
 - 每轮while循环: $O(n + m)$
 - 循环的轮数: $O(m)$

算法 3.2: 弗勒里算法

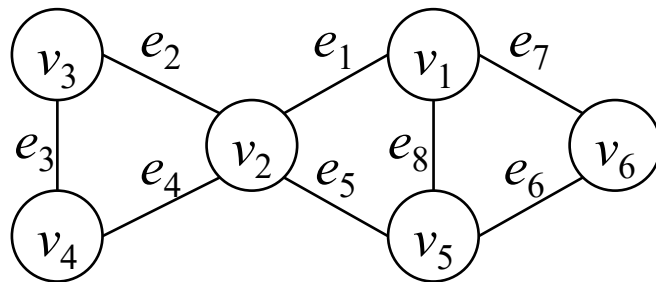
输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists v \in V, d(v)$ 是奇数 then
2   |  $u \leftarrow v$ ;
3 else
4   |  $u \leftarrow V$  中任意一个非孤立点;
5 输出 ( $u$ );
6 while  $u$  非孤立点 do
7   | if  $u$  关联一条非割边  $e'$  then
8     |  $e \leftarrow e'$ ;
9   | else
10    |  $e \leftarrow u$  关联的任意一条边;
11  输出 ( $e$ );
12   $u \leftarrow e$  的另一个端点;
13  输出 ( $u$ );
14   $G \leftarrow G - e$ ;
```



欧拉图

- 弗勒里算法
- 希尔霍尔策算法：闭迹拼接



欧拉图

- Carl Hierholzer, 1840年出生于德国

欧拉图

■ 第一条迹如何选择?

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

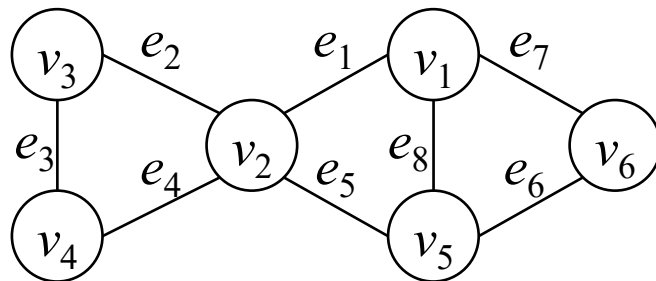
6

7

8

9

10



欧拉图

■ 第一条迹如何选择?

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 **if** $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 **then**

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 **else**

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

6

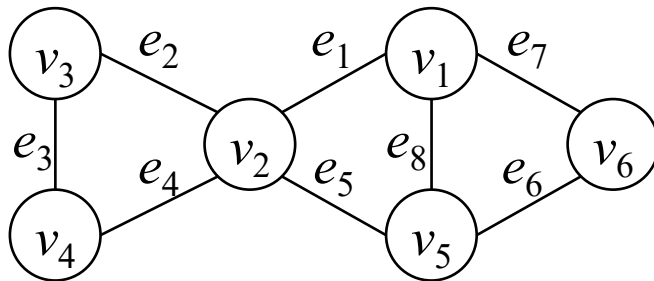
7

8

9

10

连接奇度顶点的迹,
如果有的话



欧拉图

- 接下来，能找到“有交点”的闭迹就拼接

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V, d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

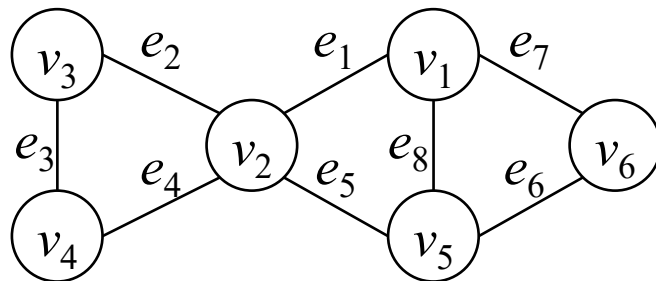
6 while $\exists w \in V, w$ 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)



欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

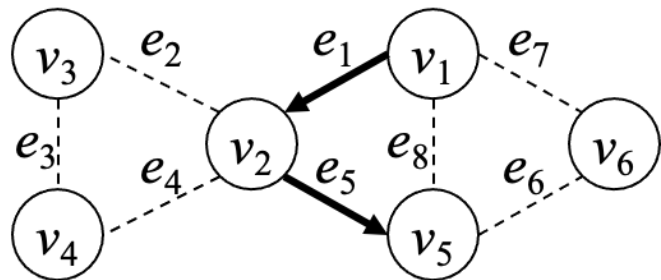
6 while $\exists w \in V$, w 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)



欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

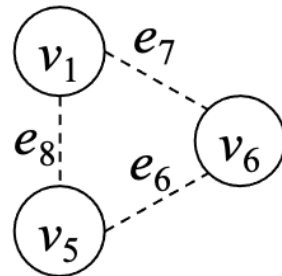
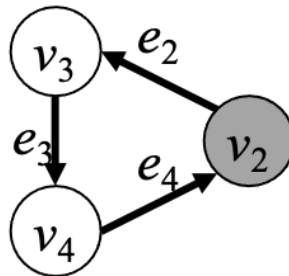
6 while $\exists w \in V$, w 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)



欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V, d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

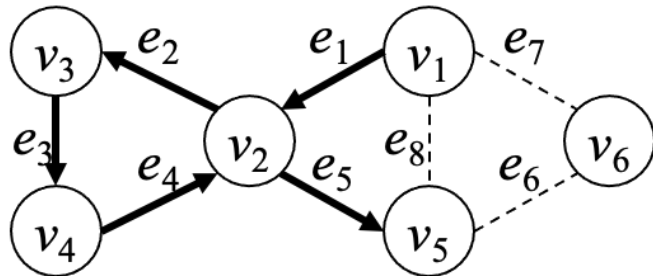
6 while $\exists w \in V, w$ 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)

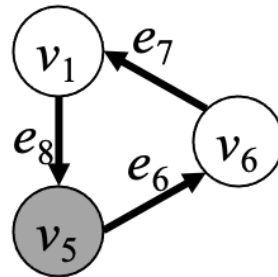


欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V$ ,  $d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u$ - $v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V$ ,  $w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w$ - $w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

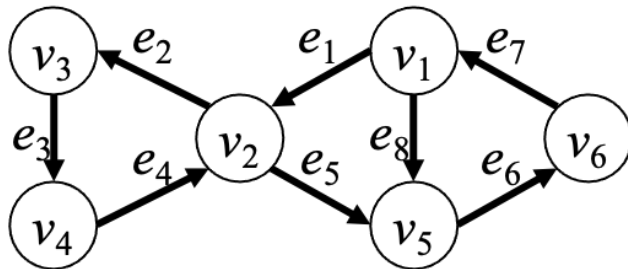
6 while $\exists w \in V$, w 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)

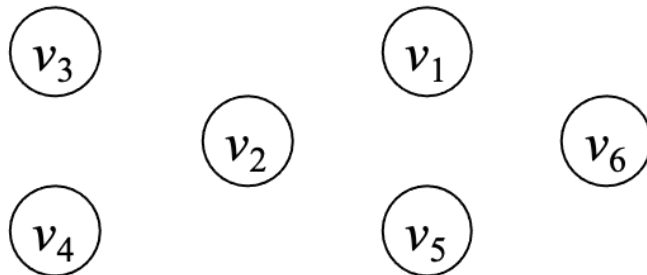


欧拉图

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V$ ,  $d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u$ - $v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V$ ,  $w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w$ - $w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



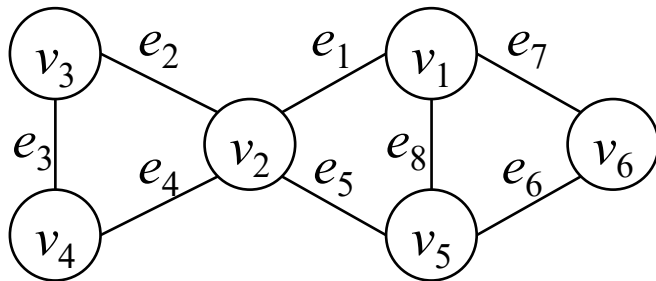
欧拉图

- 正确性：每一轮while循环条件判定前
 - 或者 G 为空图，则原图的欧拉迹已找到
 - 或者 G 的每个非平凡连通分支都含（该分支自身的）欧拉回路，且都和 T 有公共顶点，则这些欧拉回路和 T 可拼接得到原图的欧拉迹

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V, d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u-v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V, w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w-w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



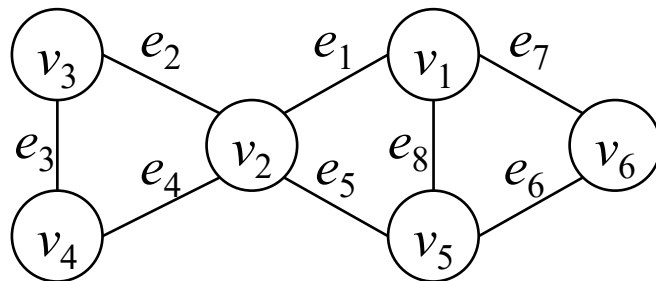
欧拉图

- 希尔霍尔策算法每一轮while循环条件判定前， G 没有顶点的度为奇数。
- 希尔霍尔策算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图的每个连通分支都和 T 有公共顶点。

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V, d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u$ - $v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V, w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w$ - $w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



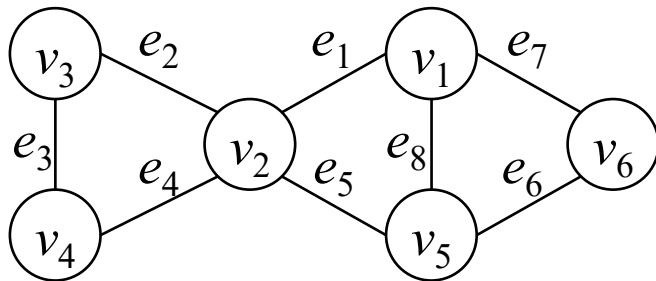
欧拉图

- 希尔霍尔策算法每一轮while循环条件判定前， G 没有顶点的度为奇数。
- 希尔霍尔策算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图的每个连通分支都和 T 有公共顶点。

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V, d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u$ - $v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V, w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w$ - $w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



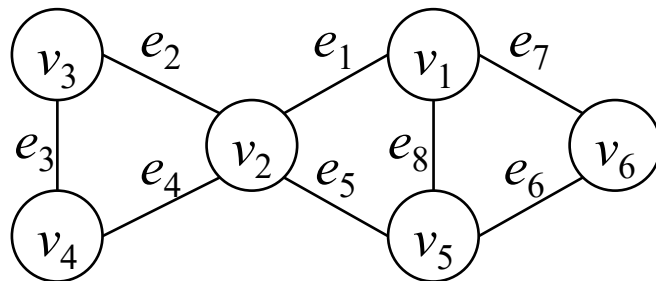
欧拉图

- 希尔霍尔策算法每一轮while循环条件判定前， G 没有顶点的度为奇数。
- 希尔霍尔策算法每一轮while循环条件判定前， G 或者为空图，或者边集的边导出子图的每个连通分支都和 T 有公共顶点。

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
1 if  $\exists u, v \in V, d(u)$ 是奇数 且  $d(v)$ 是奇数 then
2   |  $T \leftarrow G$  中任意一条  $u-v$  迹;
3 else
4   |  $T \leftarrow G$  中任意一条闭迹;
5  $G \leftarrow G - T$  经过的边的集合;
6 while  $\exists w \in V, w$  非孤立点且  $T$  经过  $w$  do
7   |  $T' \leftarrow G$  中任意一条  $w-w$  闭迹;
8   |  $T \leftarrow T$  和  $T'$  拼接;
9   |  $G \leftarrow G - T'$  经过的边的集合;
10 输出 ( $T$ )
```



欧拉图

- 在希尔霍尔策算法中，需要的迹一定存在吗？如何找出这样的迹？

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条 闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

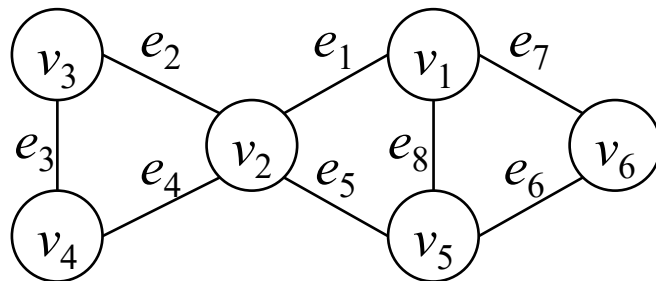
6 while $\exists w \in V$, w 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)



欧拉图

- 在希尔霍尔策算法中，需要的迹一定存在吗？如何找出这样的迹？
 - 根据顶点的度的奇偶性，一定能走，且一定能走到 v 或走回 u 或 w

算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V$, $d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 u - v 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

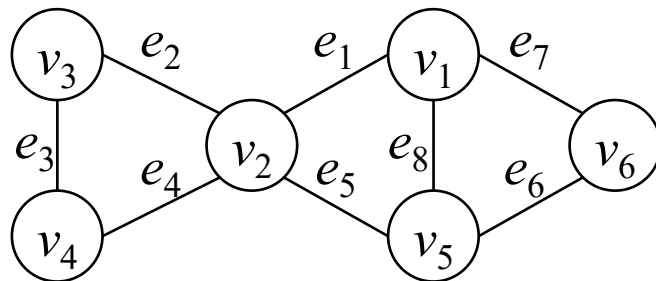
6 while $\exists w \in V$, w 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 w - w 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

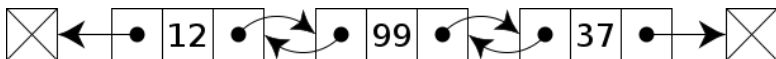
10 输出 (T)



欧拉图

■ 时间复杂度: $O(n + m)$

- 假设采用双向链表等数据结构来存储迹、维护所有满足条件的 w 、维护每个 w 关联的 T 尚未经过的边



算法 3.3: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

1 if $\exists u, v \in V, d(u)$ 是奇数 且 $d(v)$ 是奇数 then

2 | $T \leftarrow G$ 中任意一条 $u-v$ 迹;

3 else

4 | $T \leftarrow G$ 中任意一条闭迹;

5 $G \leftarrow G - T$ 经过的边的集合;

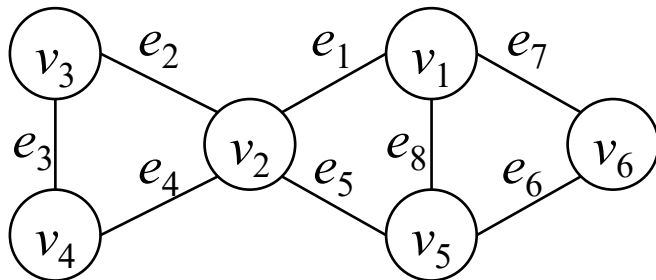
6 while $\exists w \in V, w$ 非孤立点且 T 经过 w do

7 | $T' \leftarrow G$ 中任意一条 $w-w$ 闭迹;

8 | $T \leftarrow T$ 和 T' 拼接;

9 | $G \leftarrow G - T'$ 经过的边的集合;

10 输出 (T)



本次课的主要内容

3.1 圈和树

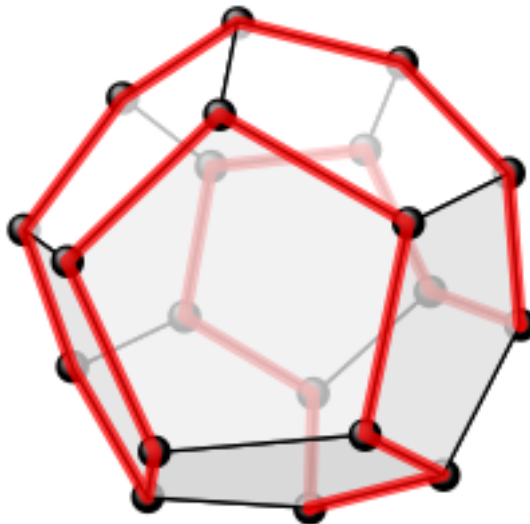
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

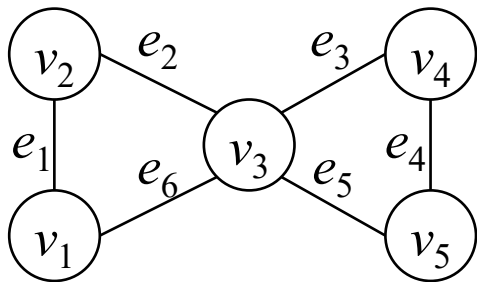
哈密尔顿图

- William Rowan Hamilton, 1805年出生于爱尔兰



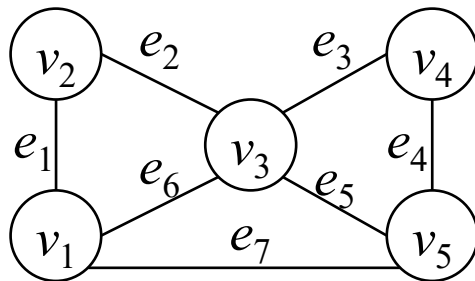
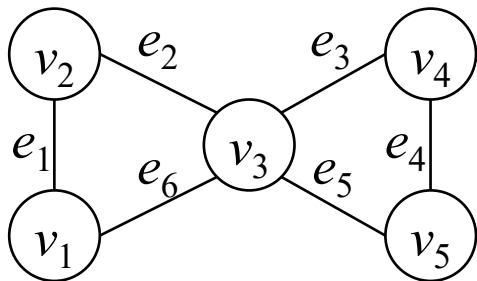
哈密尔顿图

- 哈密尔顿路：经过图的所有顶点的路



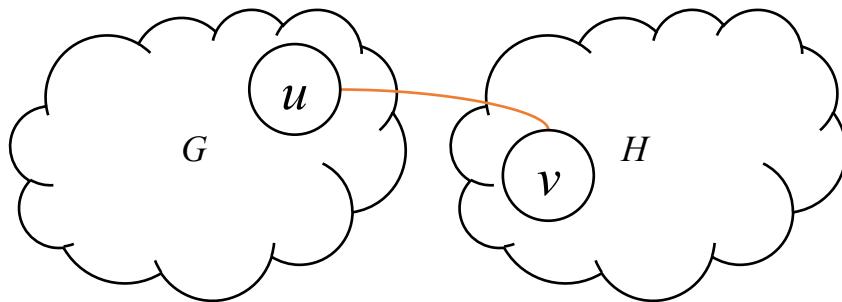
哈密尔顿图

- 哈密尔顿路：经过图的所有顶点的路
- 哈密尔顿圈：经过图的所有顶点的圈
- 哈密尔顿图：含哈密尔顿圈的图



哈密尔顿图

- 对于哈密尔顿图 $G = \langle V_G, E_G \rangle$ 和 $H = \langle V_H, E_H \rangle$, 任取顶点 $u \in V_G$ 和 $v \in V_H$, 向 G 和 H 的不交并 $G + H$ 中增加边 (u, v) 形成的图
 - 是哈密尔顿图吗?
 - 含哈密尔顿路吗?



哈密尔顿图

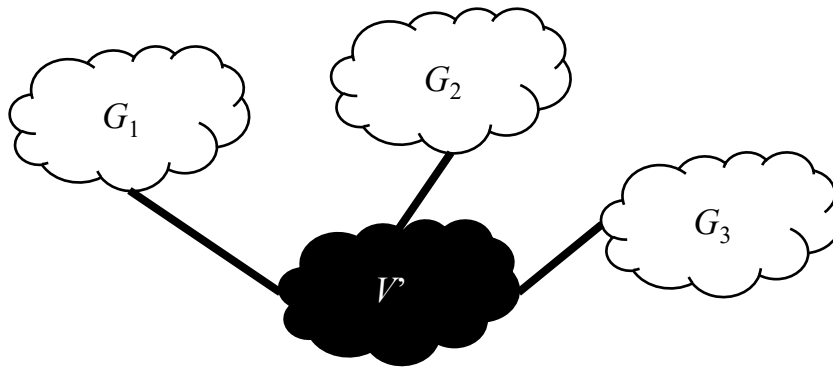
- 哈密尔顿图一定连通吗？有割点和割边吗？

哈密尔顿图

- 哈密尔顿图一定连通吗？有割点和割边吗？
- 欧拉图是哈密尔顿图吗？哈密尔顿图是欧拉图吗？

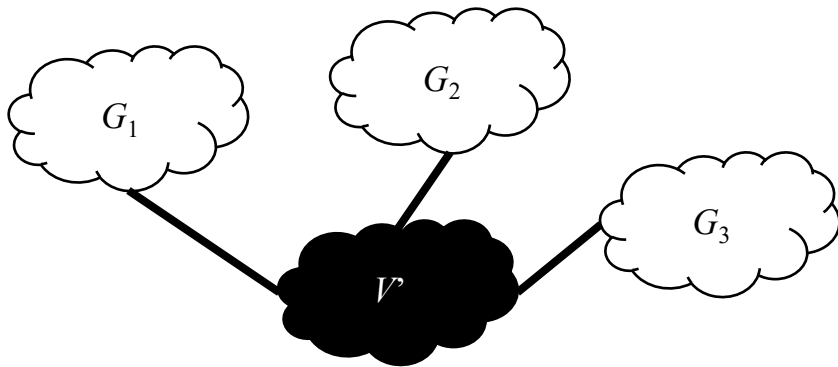
哈密尔顿图

- 哈密尔顿图一定连通吗？有割点和割边吗？
- 欧拉图是哈密尔顿图吗？哈密尔顿图是欧拉图吗？
- 若图 $G = \langle V, E \rangle$ 是哈密尔顿图，则对于任意一个顶点子集 $\emptyset \subset V' \subset V$ ， $G - V'$ 含至多 $|V'|$ 个连通分支。



哈密尔顿图

- 哈密尔顿图一定连通吗？有割点和割边吗？
- 欧拉图是哈密尔顿图吗？哈密尔顿图是欧拉图吗？
- 若图 $G = \langle V, E \rangle$ 是哈密尔顿图，则对于任意一个顶点子集 $\emptyset \subset V' \subset V$ ， $G - V'$ 含至多 $|V'|$ 个连通分支。
 - 若图 G 连通，则上述必要条件是充分条件吗？



哈密尔顿图

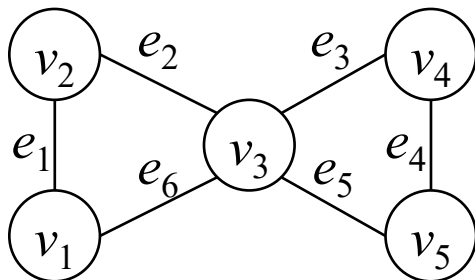
- 哈密尔顿图一定连通吗？有割点和割边吗？
- 欧拉图是哈密尔顿图吗？哈密尔顿图是欧拉图吗？
- 若图 $G = \langle V, E \rangle$ 是哈密尔顿图，则对于任意一个顶点子集 $\emptyset \subset V' \subset V$ ， $G - V'$ 含至多 $|V'|$ 个连通分支。
 - 若图 G 连通，则上述必要条件是充分条件吗？
- 对于阶为 n 的图 $G = \langle V, E \rangle$ ，若 $n \geq 3$ 且任意两个不相邻顶点 $u, v \in V$ 都满足 $d(u) + d(v) \geq n$ ，则 G 为哈密尔顿图。
 - 该充分条件是必要条件吗？

哈密尔顿图

- 如何判定一个图是否含哈密尔顿路?
- 如何判定一个图是否含哈密尔顿圈?

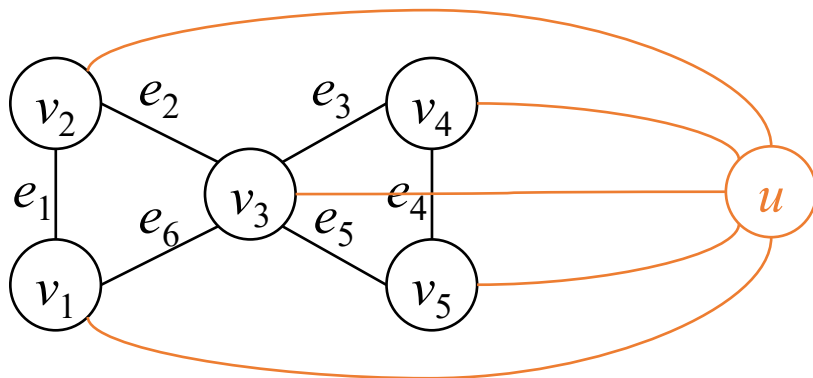
哈密尔顿图

- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿路当且仅当 H 含哈密尔顿圈？
- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿圈当且仅当 H 含哈密尔顿路？



哈密尔顿图

- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿路当且仅当 H 含哈密尔顿圈？
- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿圈当且仅当 H 含哈密尔顿路？



哈密尔顿图

- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿路当且仅当 H 含哈密尔顿圈？
- 对于图 G ，如何构造图 H ，使 G 含哈密尔顿圈当且仅当 H 含哈密尔顿路？
- 哈密尔顿路和哈密尔顿圈的存在性的判定问题的复杂度属于NPC，可以归约为旅行商问题（TSP），通常采用解决旅行商问题的算法来判定。

书面作业

- 练习3.4、3.5
- 练习3.12、3.13
- 练习3.24