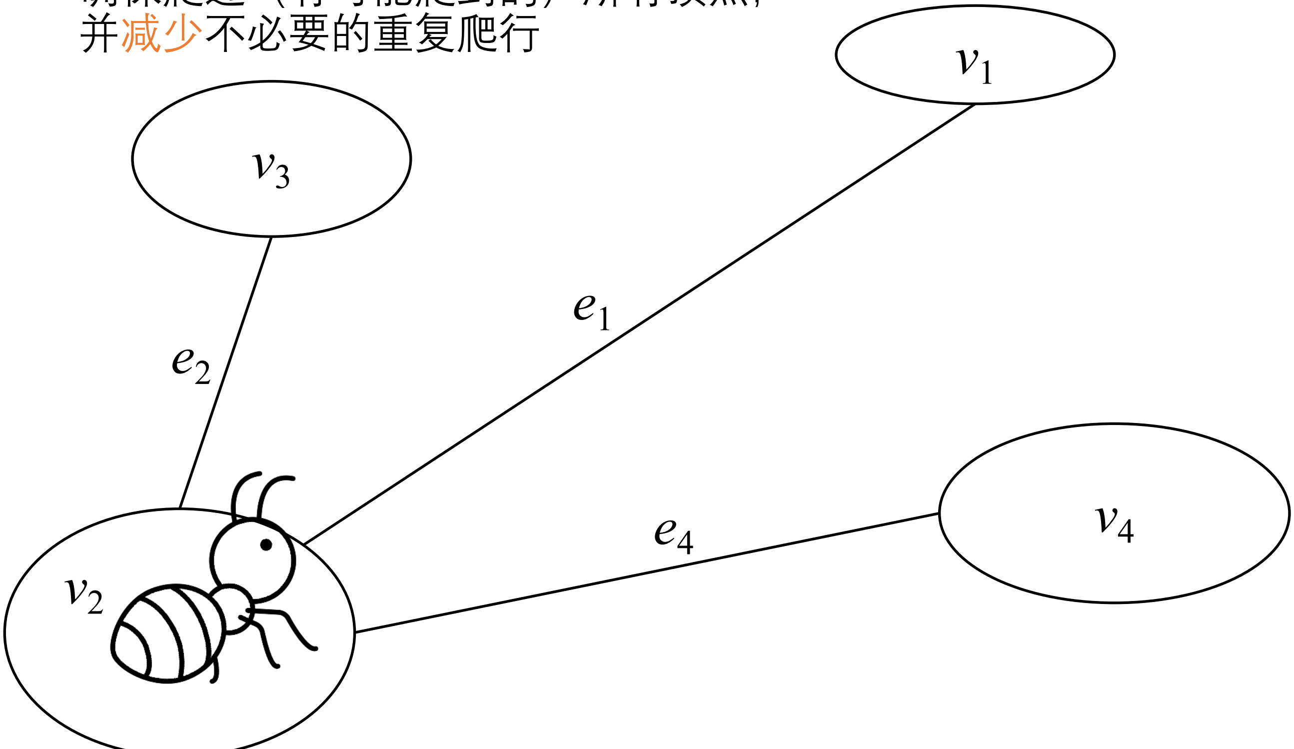
3 圈和遍历

程龚

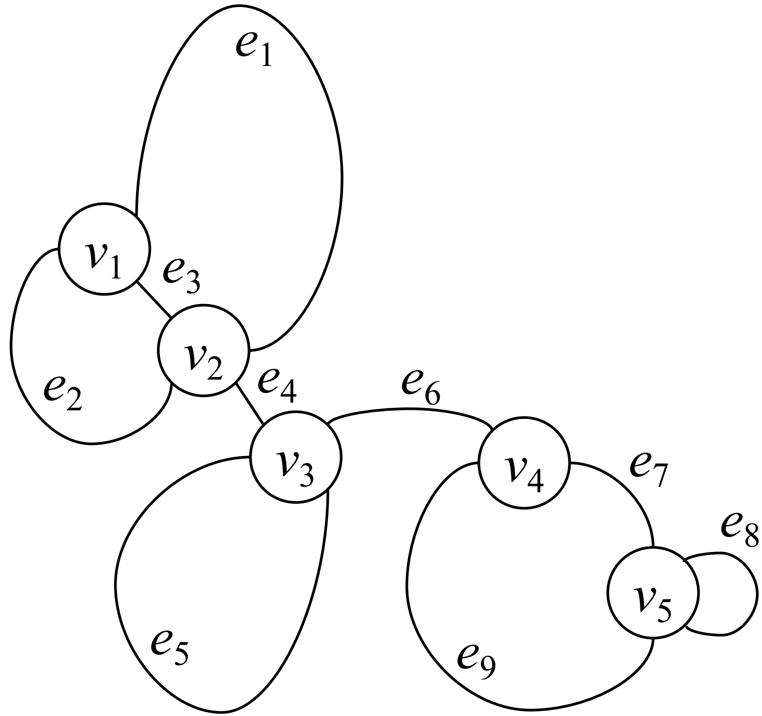
上次课讨论的“遍历”

- 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

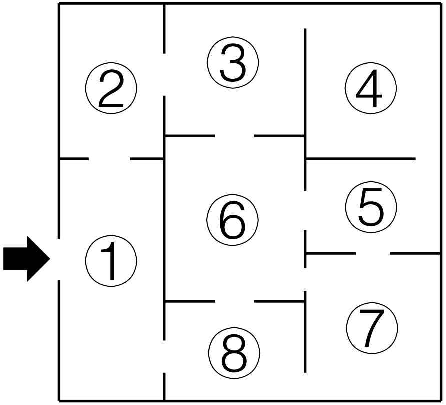


本次课讨论更严格的“遍历”

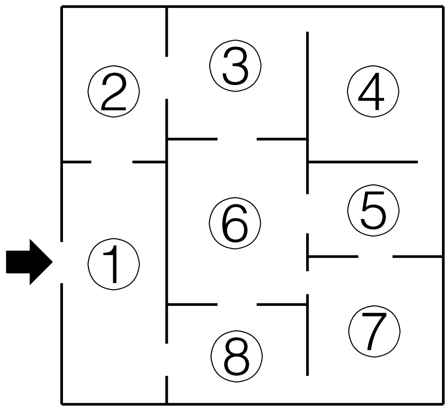
本次课讨论更严格的“遍历”

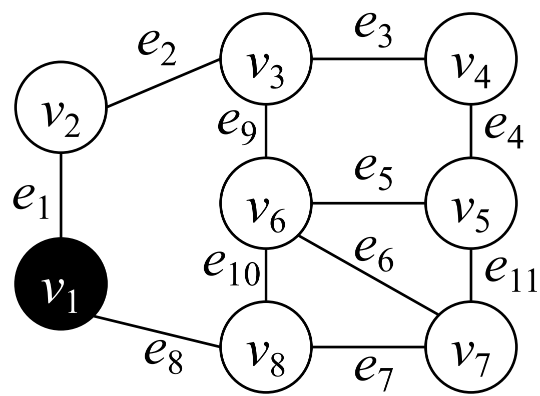


本次课讨论更严格的“遍历”



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

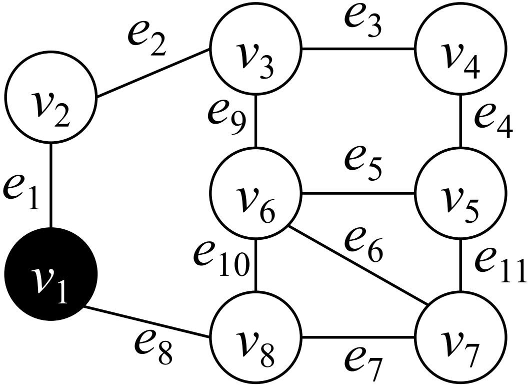
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

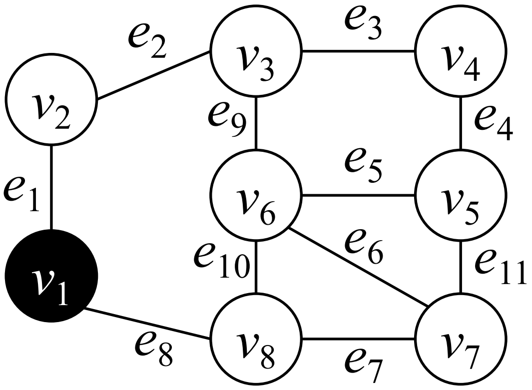
圈和树

- **闭路线**：起点和终点相同的非平凡路线



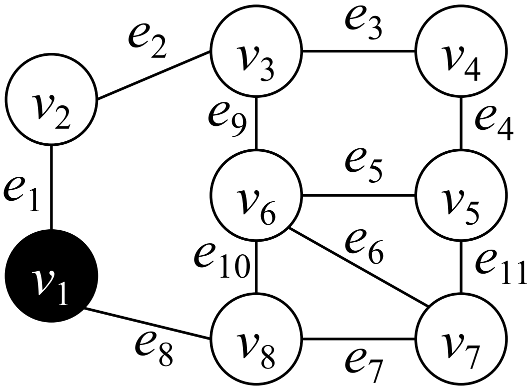
圈和树

- 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



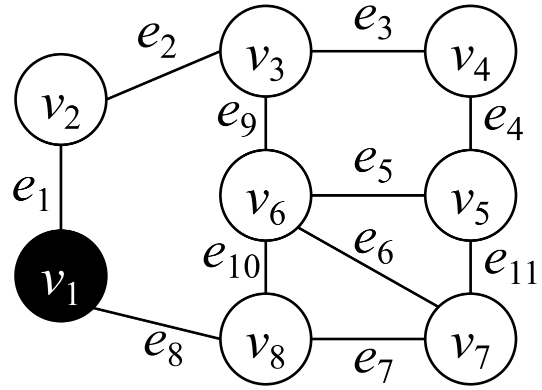
圈和树

- 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

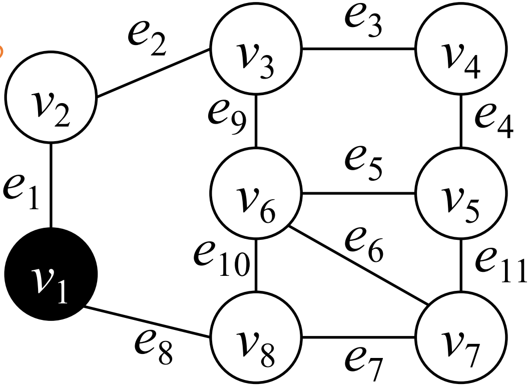
- 闭路线：起点和终点相同的非平凡路线
- 闭迹（回路）：边不重复出现的闭路线
- 圈：顶点不重复出现（除起点和终点相同）的闭迹
- **围长**：最短圈的长度
- **周长**：最长圈的长度



圈和树

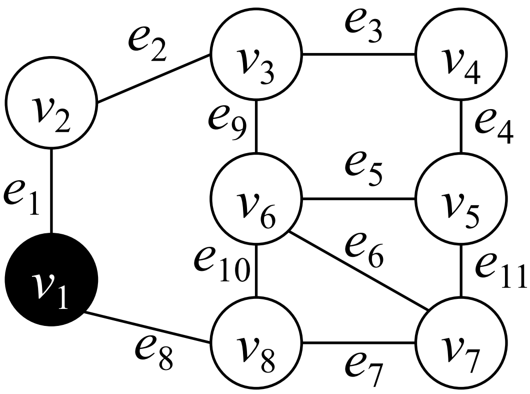
- 闭路线：起点和终点相同的非平凡路线
- 闭迹（回路）：边不重复出现的闭路线
- 圈：顶点不重复出现（除起点和终点相同）的闭迹
- 围长：最短圈的长度
- 周长：最长圈的长度

- 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？



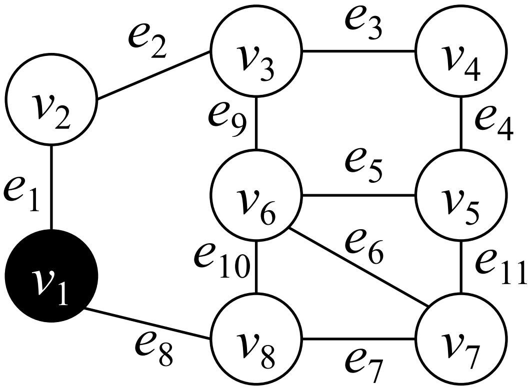
圈和树

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



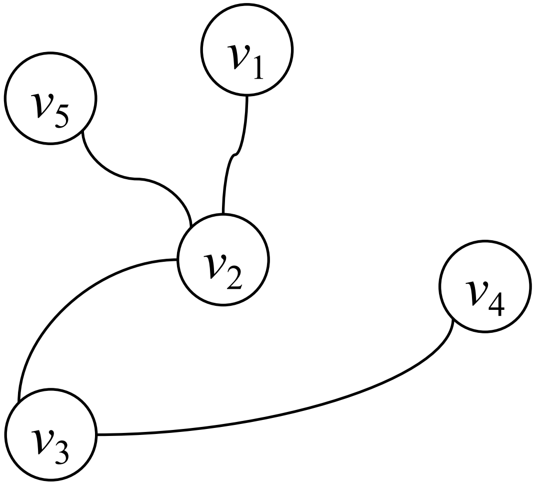
圈和树

- 对于图$G = <V, E>$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$G = <V, E>$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？



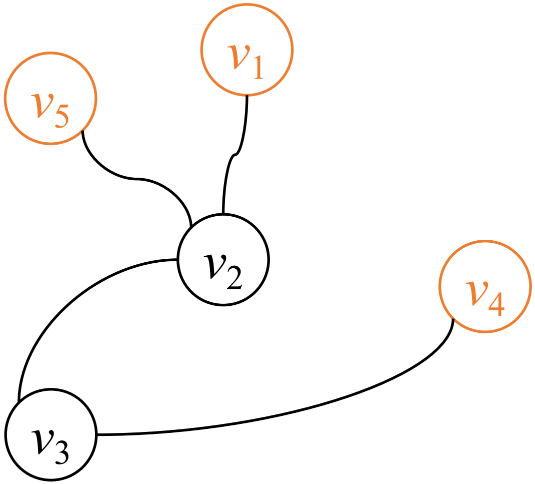
圈和树

- **树**：不含圈的连通图



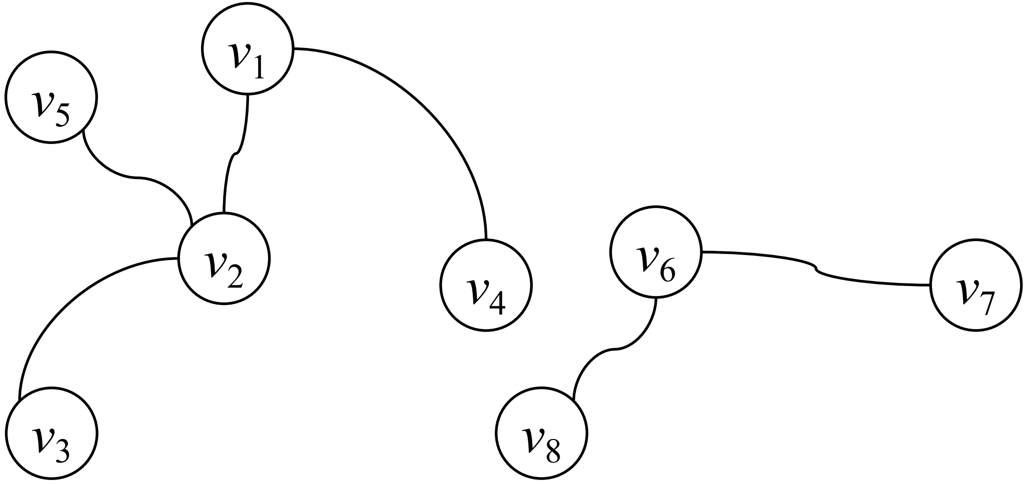
圈和树

- 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：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

- 完全图是树吗？

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$\varepsilon(G)=\nu(G)-1$。
  - 图G连通且$\varepsilon(G)=\nu(G)-1$。
  - 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$\varepsilon(G)=\nu(G)-1$。
  - 图G连通且$\varepsilon(G)=\nu(G)-1$。
  - 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。
- 树的每条边都是割边吗？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$\varepsilon(G)=\nu(G)-1$。
  - 图G连通且$\varepsilon(G)=\nu(G)-1$。
  - 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。
- 树的每条边都是割边吗？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$\varepsilon(G)= \nu(G)-1$。
  - 图G连通且$\varepsilon(G)= \nu(G)-1$。
  - 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。
- 树的每条边都是割边吗？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$\varepsilon(G)=\nu(G)-1$。
  - 图G连通且$\varepsilon(G)=\nu(G)-1$。
  - 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。
- 树的每条边都是割边吗？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

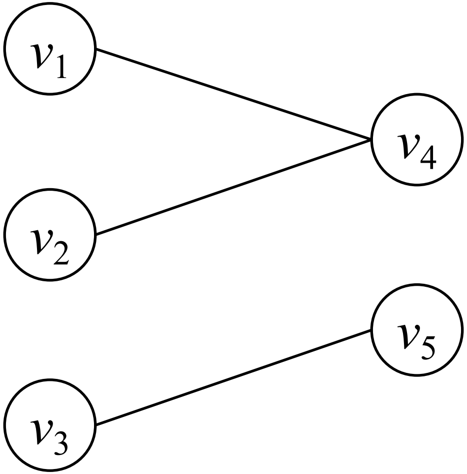
3.4 哈密尔顿图

二分图

- **奇圈**：长度为奇数的圈
- **偶圈**：长度为偶数的圈

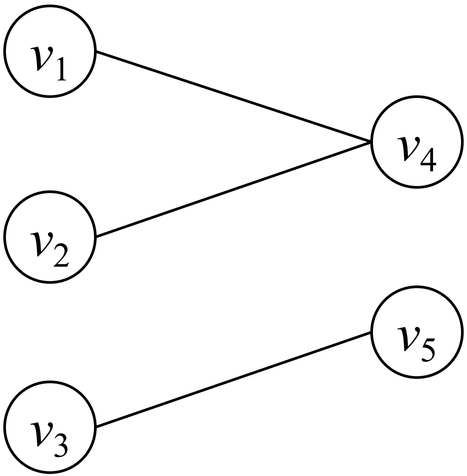
二分图

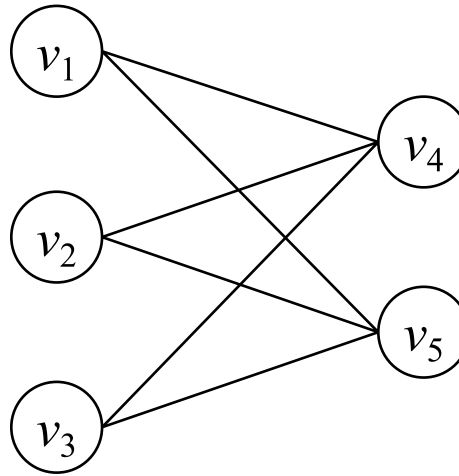
- 奇圈：长度为奇数的圈
- 偶圈：长度为偶数的圈
- **二分图**：对于图$G = <V, E>$，若V可划分为两个子集X, Y，使每条边$e \in E$的两个端点分属于X和Y



二分图

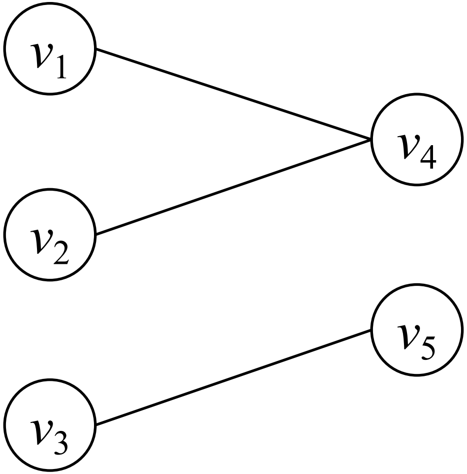
- 奇圈：长度为奇数的圈
- 偶圈：长度为偶数的圈
- 二分图：对于图$G = <V, E>$，若V可划分为两个子集X, Y，使每条边$e \in E$的两个端点分属于X和Y
- **完全二分图**：二分图，且X中每个顶点和Y中每个顶点都相邻





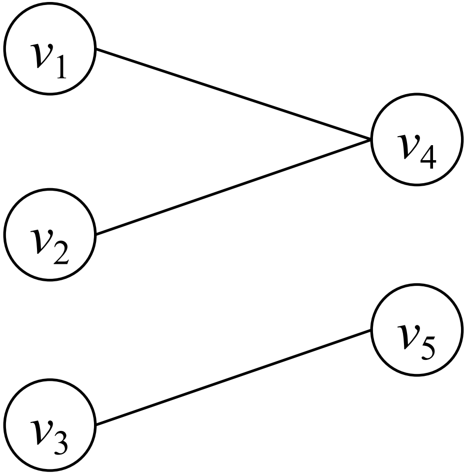
二分图

- 平凡图是二分图吗？



二分图

- 平凡图是二分图吗？
- 非平凡图G是二分图的充要条件是G不含奇圈。

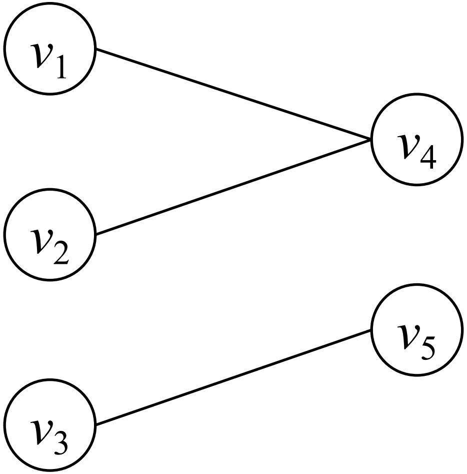


二分图

- 平凡图是二分图吗？

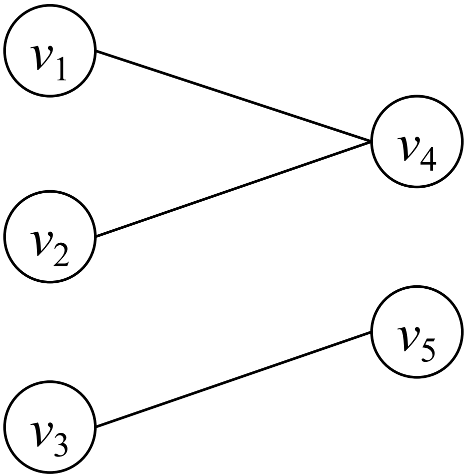
- 非平凡图G是二分图的充要条件是G不含奇圈。

- 完全图是二分图吗？



二分图

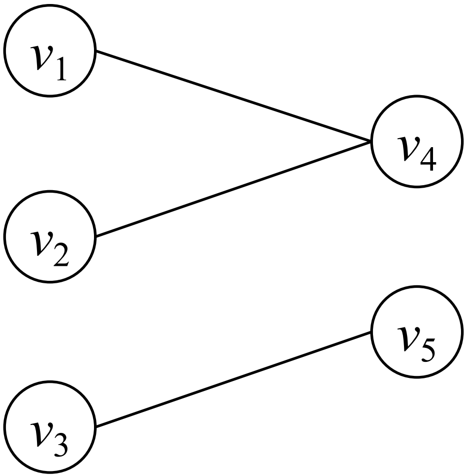
- 平凡图是二分图吗？
- 非平凡图G是二分图的充要条件是G不含奇圈。
- 完全图是二分图吗？
- 树是二分图吗？森林呢？



二分图

- 平凡图是二分图吗？
- 非平凡图G是二分图的充要条件是G不含奇圈。
- 完全图是二分图吗？
- 树是二分图吗？森林呢？
- 对于二分图$G = <V, E>$
  - 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 4: 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

u.visited $\leftarrow$ true;
foreach $(u, v) \in E$ **do**
  if $v.visited = false$ **then**
    v.samePart $\leftarrow \neg$ u.samePart;
    `DFSBpt(`G, v`);`
  else if $u.samePart = v.samePart$ **then**
    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 4: 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

u.visited $\leftarrow$ true;

foreach $(u, v) \in E$ **do**

    if $v.visited = false$ **then**

        v.samePart $\leftarrow \neg$ u.samePart;

        `DFSBpt(`G, v`);`

    else if $u.samePart = v.samePart$ **then**

        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 4: 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

u.visited $\leftarrow$ true;

foreach $(u, v) \in E$ **do**

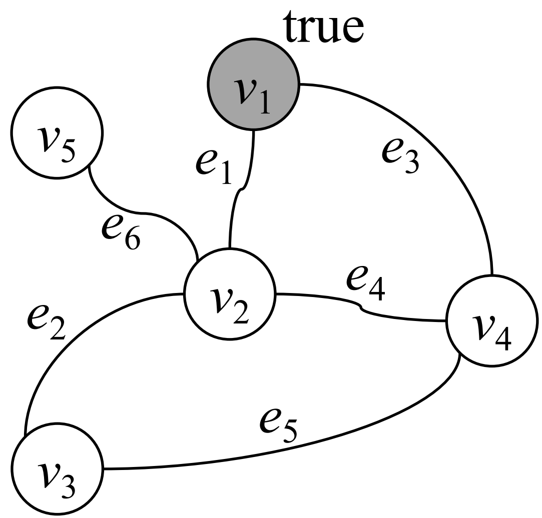
  if $v.visited = false$ **then**

    v.samePart $\leftarrow \neg$ u.samePart;

    `DFSBpt(`G`, `v`);`

  else if $u.samePart = v.samePart$ **then**

    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 4: 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

u.visited $\leftarrow$ true;

foreach $(u, v) \in E$ **do**

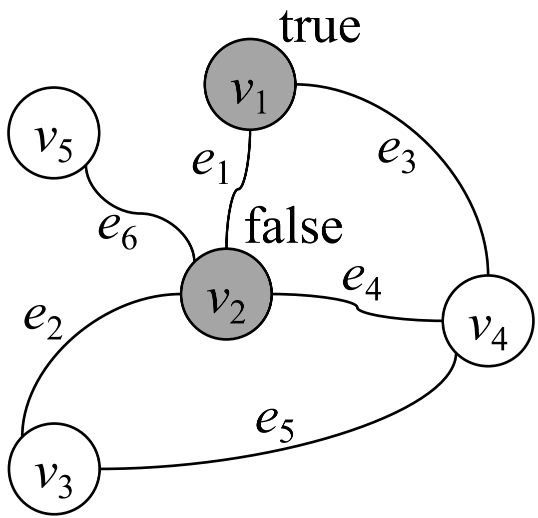
    if $v.visited = false$ **then**

        v.samePart $\leftarrow \neg$ u.samePart;

        `DFSBpt(`G, v`);`

    else if $u.samePart = v.samePart$ **then**

        判定（“G 非二分图”）



二分图

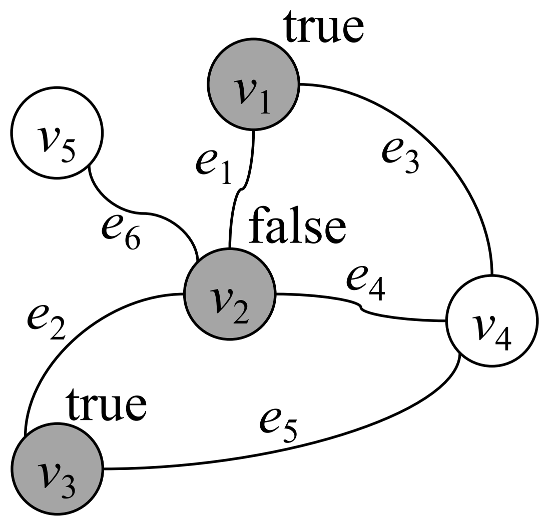
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 4: 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

u.visited $\leftarrow$ true;
foreach $(u, v) \in E$ **do**
    if $v.visited = false$ **then**
        v.samePart $\leftarrow \neg$ u.samePart;
        `DFSBpt(`G, v`);`
    else if $u.samePart = v.samePart$ **then**
        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 4: 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

u.visited $\leftarrow$ true;

foreach $(u, v) \in E$ **do**

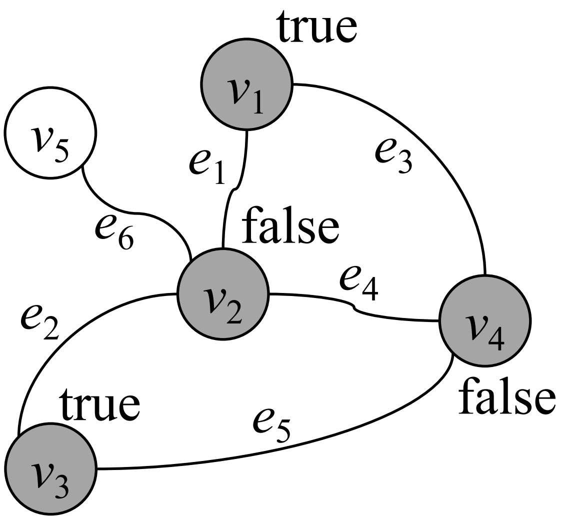
    if $v.visited = false$ **then**

        v.samePart $\leftarrow \neg\, u$.samePart;

        `DFSBpt(`G, v`);`

    else if $u.samePart = v.samePart$ **then**

        判定（“G 非二分图”）



二分图

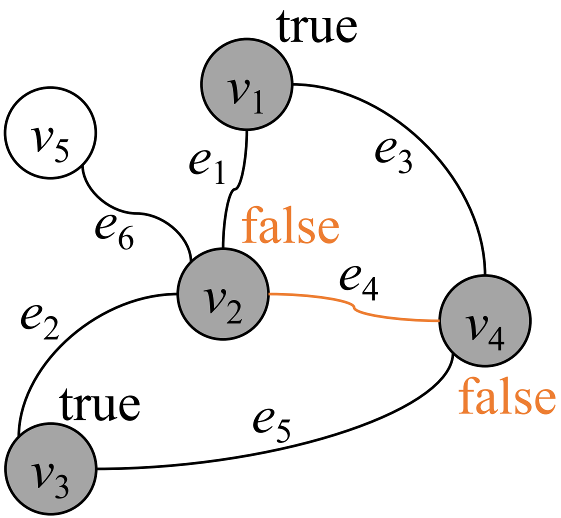
- 扩展DFS算法：尝试将每个顶点分到X或Y中

算法 4: 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

```
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        v.samePart ← ¬ u.samePart;
        DFSBpt(G, v);
    else if u.samePart = v.samePart then
        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 4: 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

u.visited ← true;

foreach $(u, v) \in E$ **do**

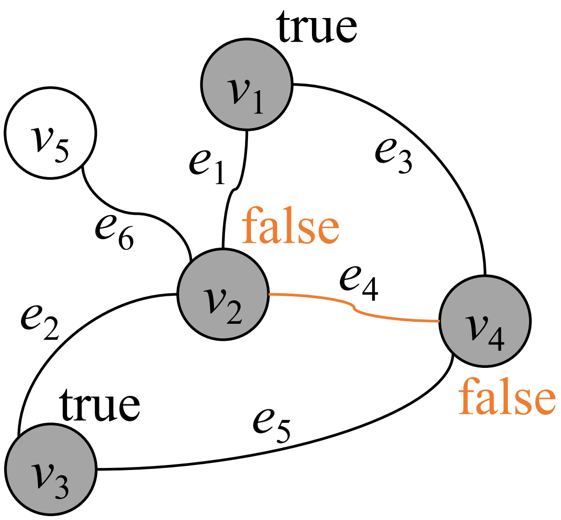
  if $v.visited = false$ **then**

    v.samePart ← ¬ u.samePart;

    DFSBpt(G, v);

  else if $u.samePart = v.samePart$ **then**

    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：发现奇圈

算法 4: 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

u.visited $\leftarrow$ true;

foreach $(u, v) \in E$ **do**

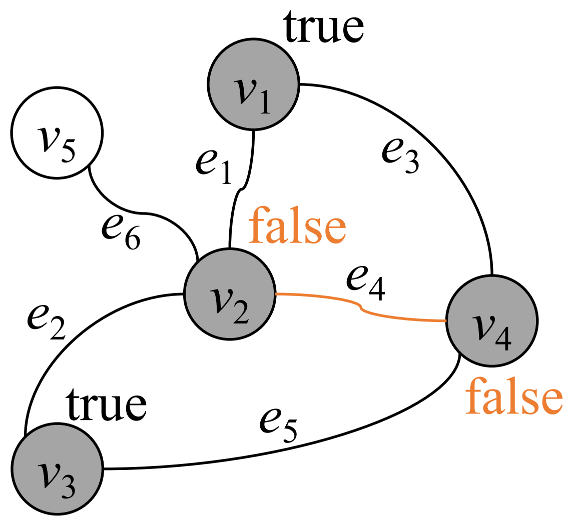
  if $v.visited = false$ **then**

    v.samePart $\leftarrow \neg$ u.samePart;

    `DFSBpt(`G, v`);`

  else if $u.samePart = v.samePart$ **then**

    判定（“G 非二分图”）



本次课的主要内容

3.1 圈和树

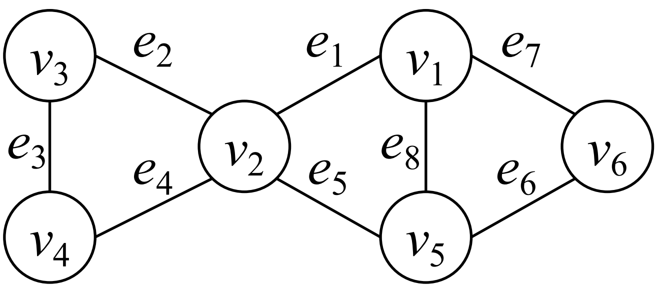
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

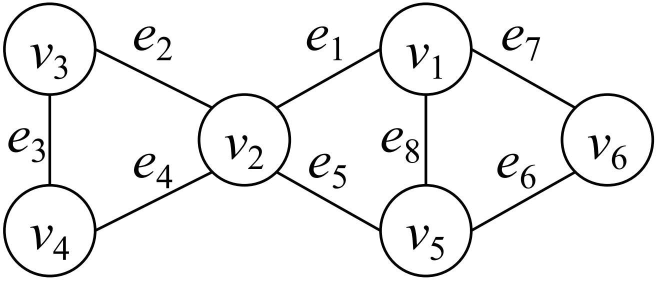
欧拉图

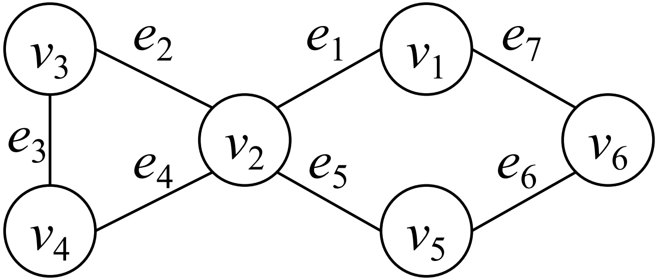
- **欧拉迹**：经过图的每条边恰一次的迹



欧拉图

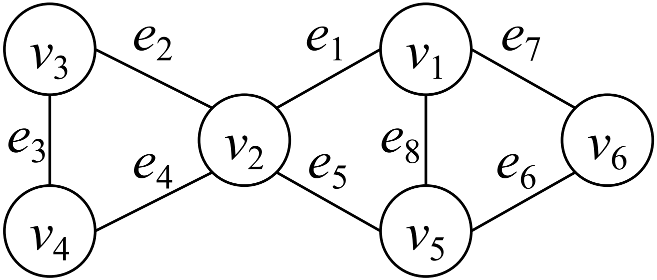
- 欧拉迹：经过图的每条边恰一次的迹
- **欧拉回路**：经过图的每条边恰一次的闭迹（回路）

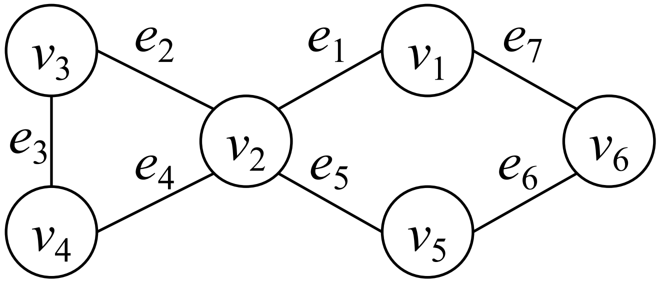




欧拉图

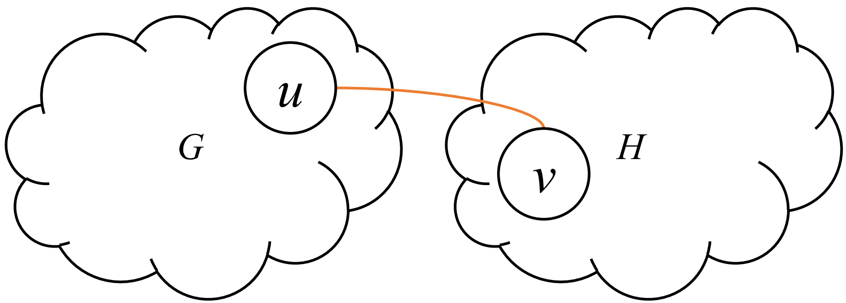
- 欧拉迹：经过图的每条边恰一次的迹
- 欧拉回路：经过图的每条边恰一次的闭迹（回路）
- **欧拉图**：含欧拉回路的图





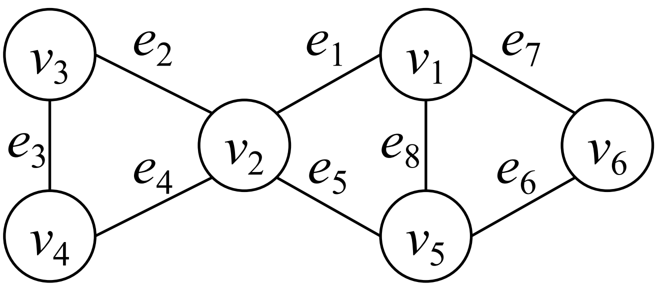
欧拉图

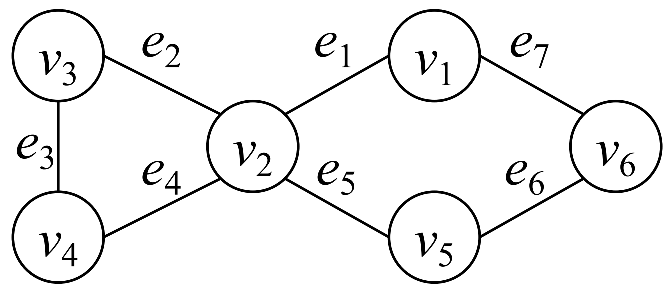
- 对于连通欧拉图$G = <V_G, E_G>$和$H = <V_H, E_H>$，任取顶点$u \in V_G$和$v \in V_H$，向G和H的不交并$G + H$中增加边(u, v)形成的图
  - 含欧拉回路吗？
  - 含欧拉迹吗？



欧拉图

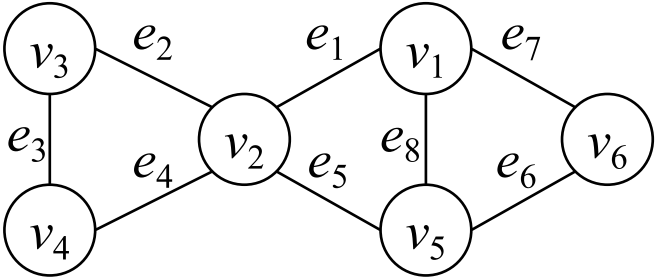
- 非空连通图G含欧拉回路当且仅当G没有顶点的度为奇数。

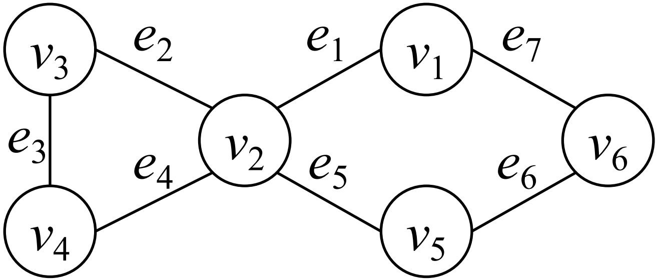




欧拉图

- 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。

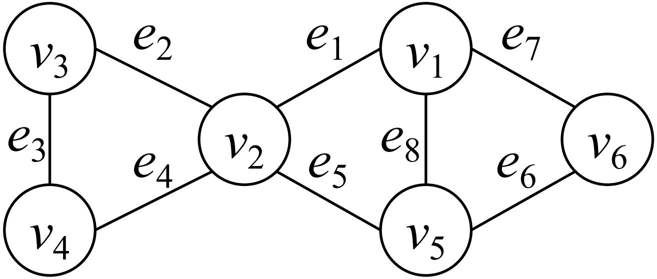


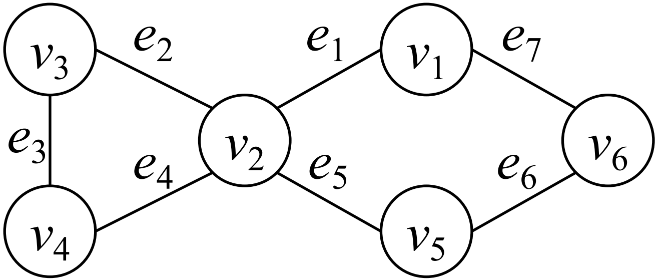


欧拉图

- 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？

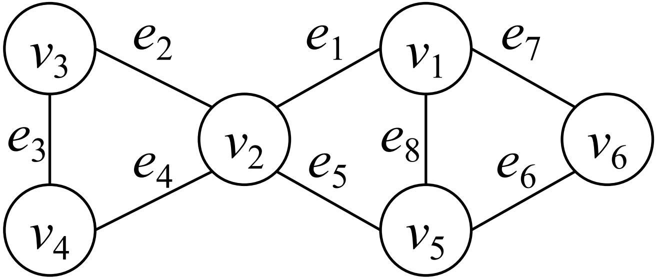


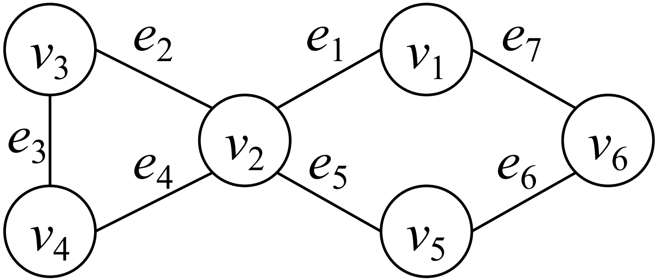


欧拉图

- 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：边集的边导出子图是非空连通图
  - 含欧拉迹的额外条件：空图，或边集的边导出子图是非空连通图

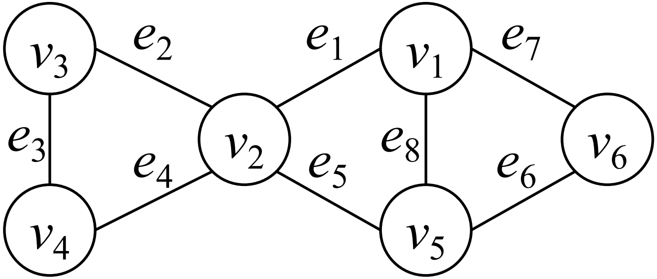


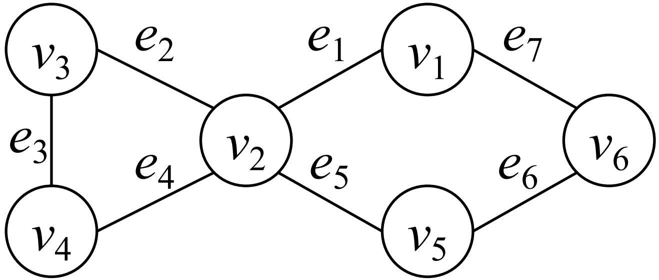


欧拉图

- 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。

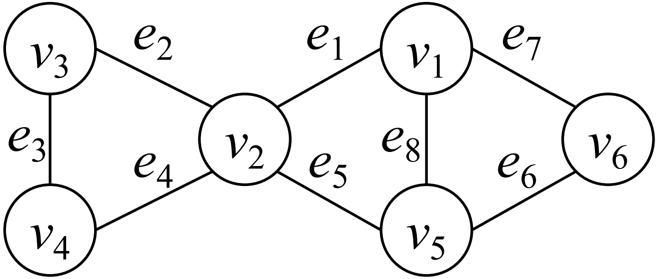
- 含欧拉迹的图一定连通吗？
  图G含欧拉回路和欧拉迹的充要条件分别是什么？
  - 含欧拉回路的额外条件：边集的边导出子图是非空连通图
  - 含欧拉迹的额外条件：空图，或边集的边导出子图是非空连通图
- 完全图是欧拉图吗？完全二分图呢？正则图呢？





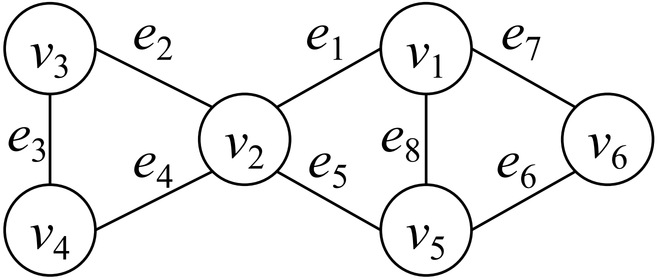
欧拉图

- 如何找出图中的一条欧拉迹？



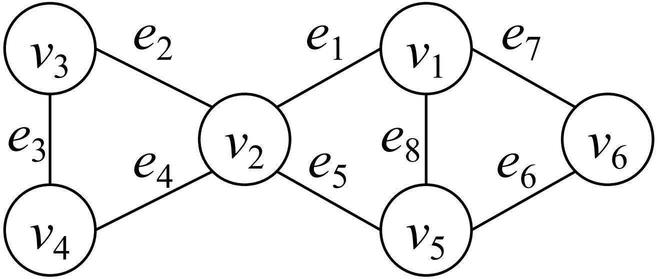
欧拉图

- 弗勒里算法
- 希尔霍尔策算法



欧拉图

- 弗勒里算法：不走割边
- 希尔霍尔策算法



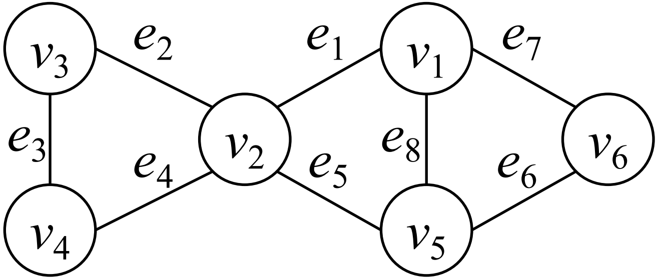
欧拉图

- 出发点如何选择？

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;



欧拉图

- 出发点如何选择？

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**

  $u \leftarrow v$;

else

  $u \leftarrow V$ 中任意一个非孤立点;

输出 (u);

while u 非孤立点 **do**

  if u 关联一条非割边 e' **then**

    $e \leftarrow e'$;

  else

    $e \leftarrow u$ 关联的任意一条边;

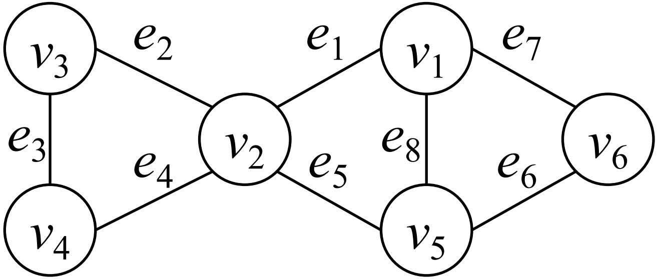
  输出 (e);

  $u \leftarrow e$ 的另一个端点;

  输出 (u);

  $G \leftarrow G - e$;

奇度顶点，
如果有的话



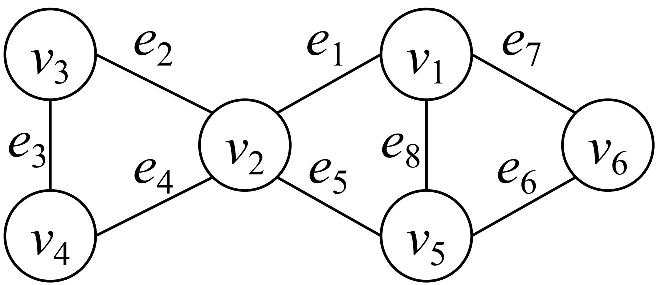
欧拉图

- 接下来，能走就走，并优先选择非割边

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E\rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
    $u \leftarrow v$;
else
    $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
    if u 关联一条非割边 e' **then**
        $e \leftarrow e'$;
    else
        $e \leftarrow u$ 关联的任意一条边;
    输出 (e);
    $u \leftarrow e$ 的另一个端点;
    输出 (u);
    $G \leftarrow G - e$;



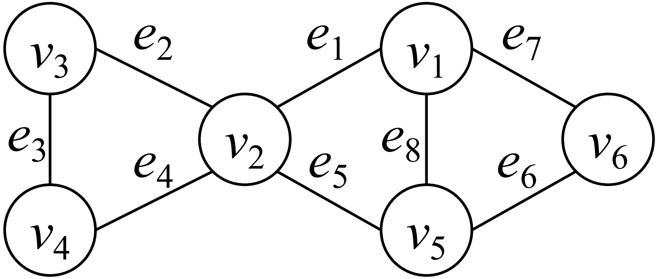
欧拉图

- 删除走过的边

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
　$u \leftarrow v$;
else
　$u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
　if u 关联一条非割边 e' **then**
　　$e \leftarrow e'$;
　else
　　$e \leftarrow u$ 关联的任意一条边;
　输出 (e);
　$u \leftarrow e$ 的另一个端点;
　输出 (u);
　$G \leftarrow G - e$;



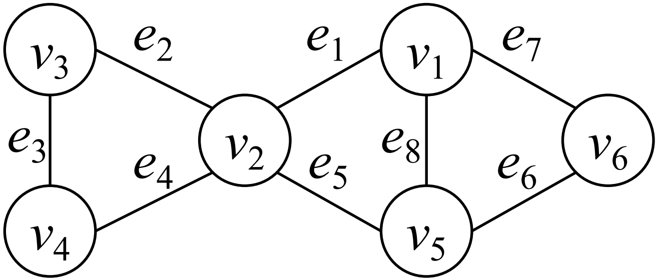
欧拉图

- 删除走过的边
  - 因此，每轮循环需要重新计算割边

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G=\langle V,E\rangle$

if $\exists v\in V,\ 2\nmid \mathrm{d}(v)$ **then**
　　$u\leftarrow v$;
else
　　$u\leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
　　if u 关联一条非割边 e' **then**
　　　　$e\leftarrow e'$;
　　else
　　　　$e\leftarrow u$ 关联的任意一条边;
　　输出 (e);
　　$u\leftarrow e$ 的另一个端点;
　　输出 (u);
　　$G\leftarrow G-e$;

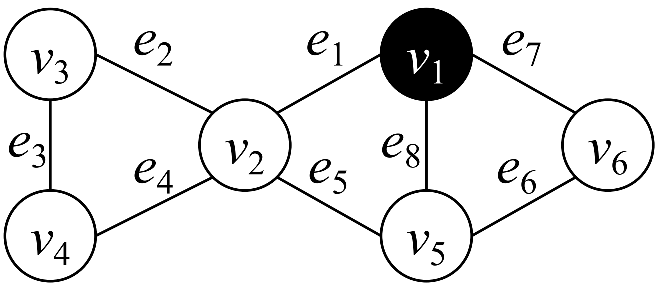


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;

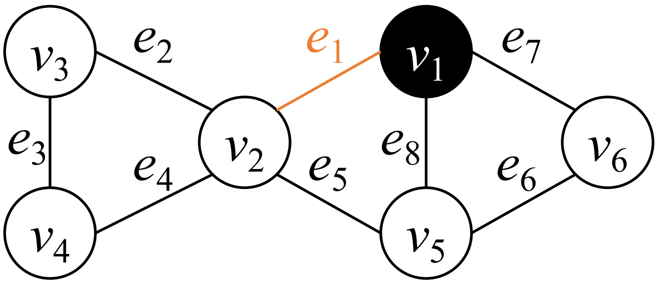


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;

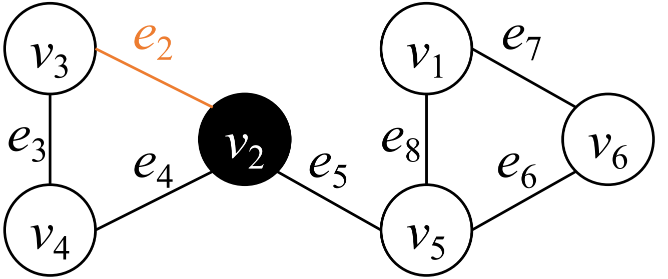


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
    $u \leftarrow v$;
else
    $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
    if u 关联一条非割边 e' **then**
        $e \leftarrow e'$;
    else
        $e \leftarrow u$ 关联的任意一条边;
    输出 (e);
    $u \leftarrow e$ 的另一个端点;
    输出 (u);
    $G \leftarrow G - e$;

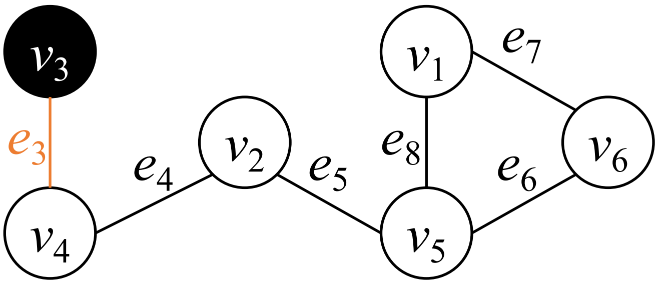


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathtt{d}(v)$ **then**
    $u \leftarrow v$;
else
    $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
    if u 关联一条非割边 e' **then**
        $e \leftarrow e'$;
    else
        $e \leftarrow u$ 关联的任意一条边;
    输出 (e);
    $u \leftarrow e$ 的另一个端点;
    输出 (u);
    $G \leftarrow G - e$;

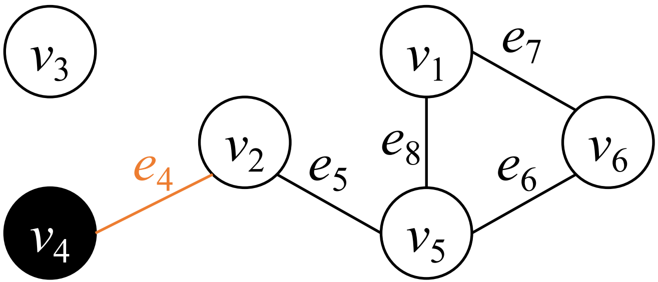


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathtt{d}(v)$ **then**
    $u \leftarrow v$;
else
    $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
    if u 关联一条非割边 e' **then**
        $e \leftarrow e'$;
    else
        $e \leftarrow u$ 关联的任意一条边;
    输出 (e);
    $u \leftarrow e$ 的另一个端点;
    输出 (u);
    $G \leftarrow G - e$;

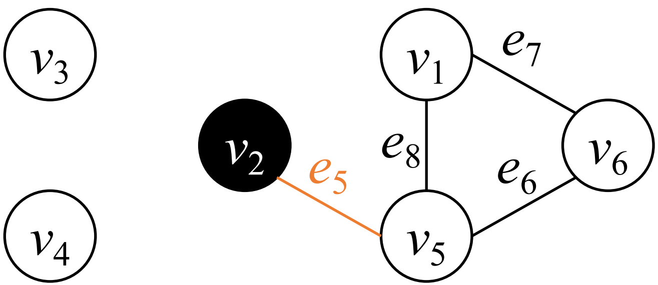


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathtt{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;

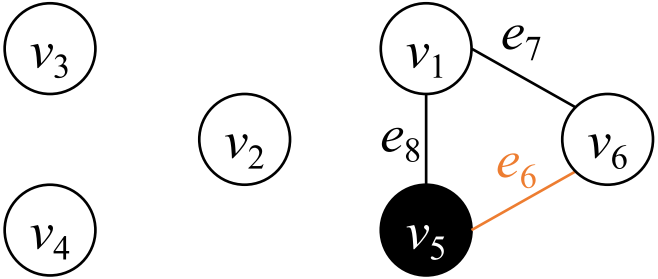


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

```
if ∃v ∈ V, 2 ∤ d(v) then
│ u ← v;
else
│ u ← V 中任意一个非孤立点;
输出 (u);
while u 非孤立点 do
│ if u 关联一条非割边 e′ then
│ │ e ← e′;
│ else
│ │ e ← u 关联的任意一条边;
│ 输出 (e);
│ u ← e 的另一个端点;
│ 输出 (u);
│ G ← G − e;
```

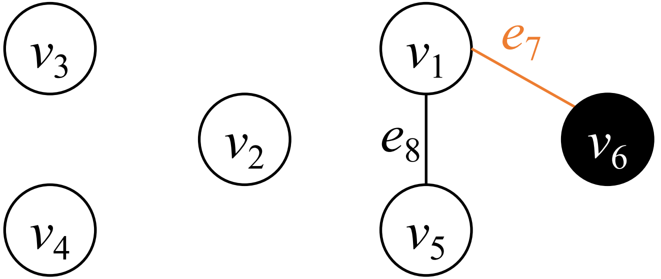


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E\rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;

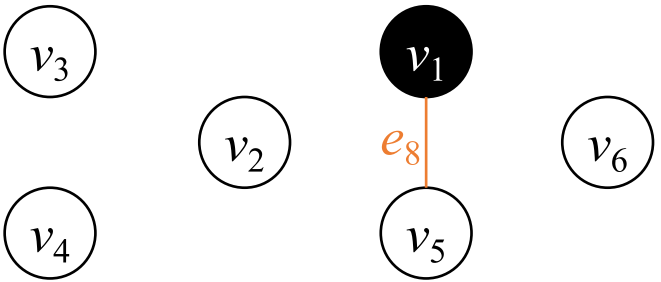


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;

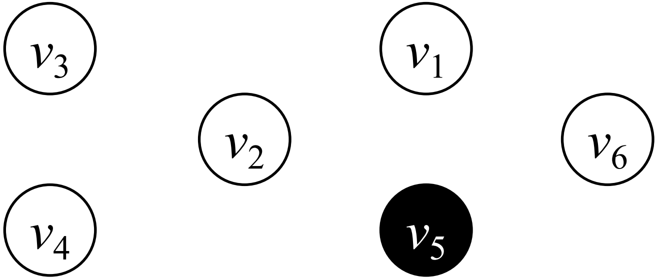


欧拉图

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
    $u \leftarrow v$;
else
    $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
    if u 关联一条非割边 e' **then**
        $e \leftarrow e'$;
    else
        $e \leftarrow u$ 关联的任意一条边;
    输出 (e);
    $u \leftarrow e$ 的另一个端点;
    输出 (u);
    $G \leftarrow G - e$;



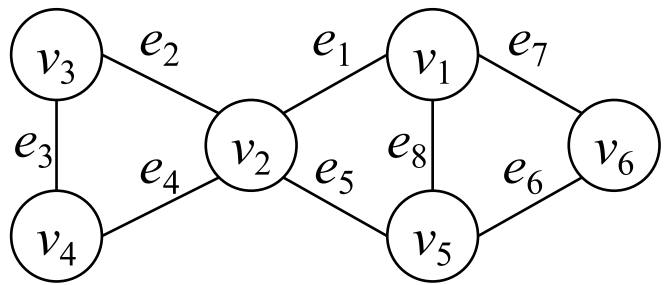
欧拉图

- 时间复杂度：$O(n + m(n + m))$
  - 假设用修改后的DFSCV算法来找割边

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;



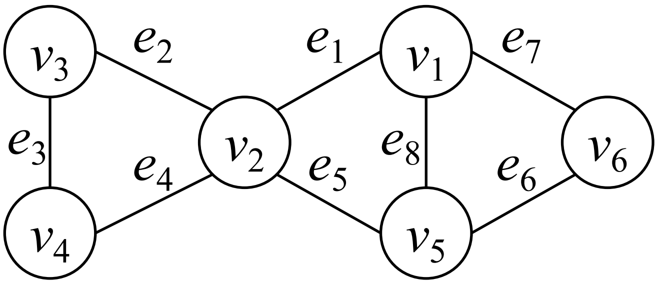
欧拉图

- 弗勒里算法每一轮循环条件判定前，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。

算法 5: 弗勒里算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;



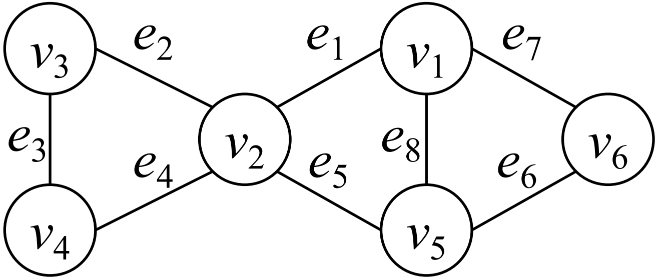
欧拉图

- 弗勒里算法每一轮循环条件判定前，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 5: 弗勒里算法

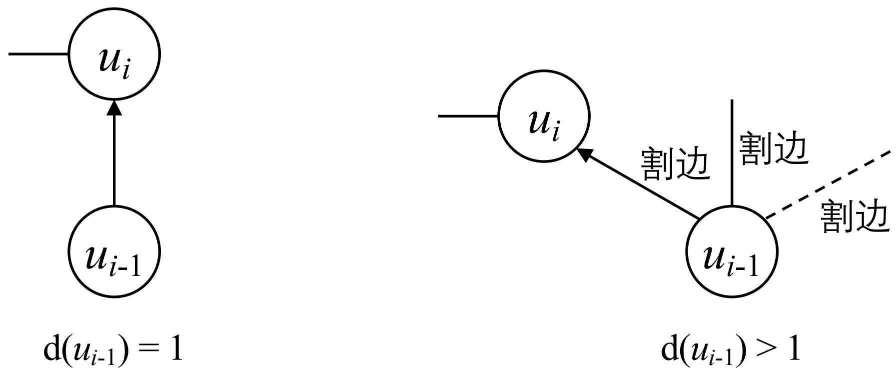
输入: 含欧拉迹的图 $G = \langle V, E\rangle$

if $\exists v \in V,\ 2 \nmid \mathrm{d}(v)$ **then**
  $u \leftarrow v$;
else
  $u \leftarrow V$ 中任意一个非孤立点;
输出 (u);
while u 非孤立点 **do**
  if u 关联一条非割边 e' **then**
    $e \leftarrow e'$;
  else
    $e \leftarrow u$ 关联的任意一条边;
  输出 (e);
  $u \leftarrow e$ 的另一个端点;
  输出 (u);
  $G \leftarrow G - e$;



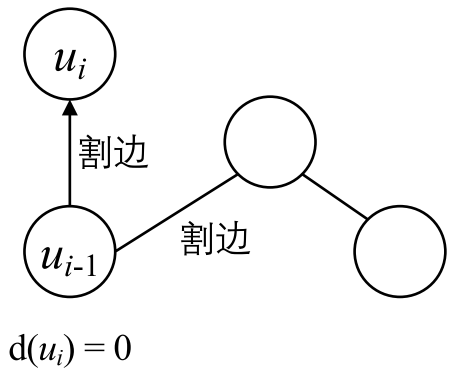
欧拉图

- 弗勒里算法每一轮循环条件判定前，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轮的u可能有2种情况，都会导致矛盾



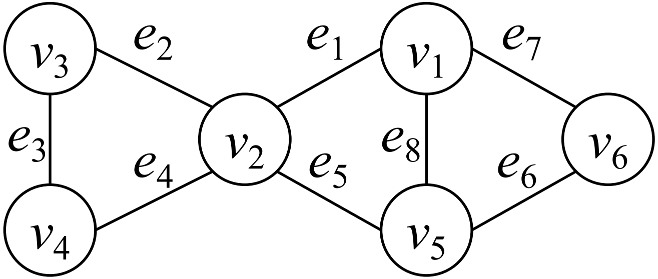
欧拉图

- 弗勒里算法每一轮循环条件判定前，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，会导致矛盾（原因类似）



欧拉图

- 弗勒里算法
- 希尔霍尔策算法：闭迹拼接



欧拉图

- 第一条迹如何选择？

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V,\ 2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**

  $T \leftarrow$ 任意一条 u-v 迹;

else

  $T \leftarrow$ 任意一条闭迹;

$G \leftarrow G - \{e \in E : T\text{经过}e\}$;

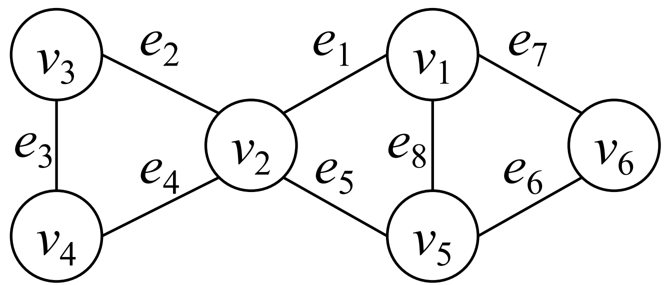
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**

  $T' \leftarrow$ 任意一条 w-w 闭迹;

  $T \leftarrow T$ 和 T' 拼接;

  $G \leftarrow G - \{e \in E : T'\text{经过}e\}$;

输出 (T)



欧拉图

- 第一条迹如何选择？

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V,\ 2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**

  $T \leftarrow$ 任意一条 u-v 迹;

else

  $T \leftarrow$ 任意一条闭迹;

$G \leftarrow G - \{e \in E : T$经过$e\}$;

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

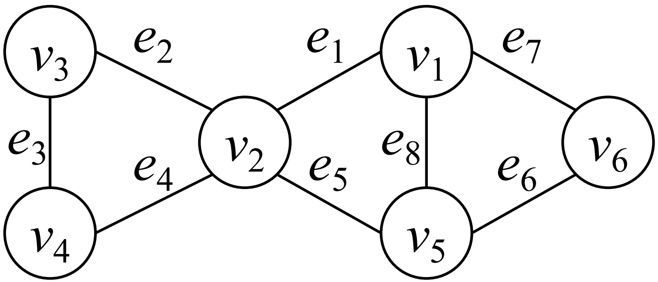
  $T' \leftarrow$ 任意一条 w-w 闭迹;

  $T \leftarrow T$ 和 T' 拼接;

  $G \leftarrow G - \{e \in E : T'$经过$e\}$;

输出 (T)

连接奇度顶点的迹，
如果有的话



欧拉图

- 接下来，能找到闭迹就拼接

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G=\langle V,E\rangle$

if $\exists u,v\in V,\ 2\nmid \mathrm{d}(u)$ 且 $2\nmid \mathrm{d}(v)$ **then**

  $T\leftarrow$ 任意一条 u-v 迹;

else

  $T\leftarrow$ 任意一条闭迹;

$G\leftarrow G-\{e\in E: T$经过$e\}$;

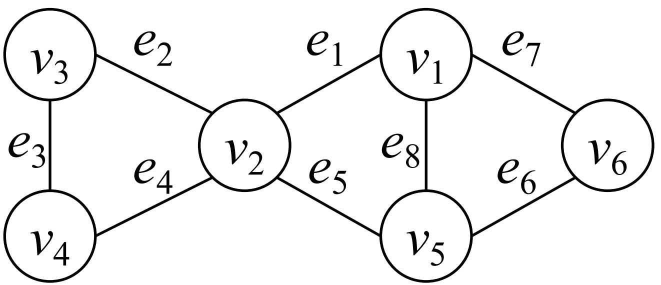
while $\exists w\in V$, w 非孤立点且 T 经过 w **do**

  $T'\leftarrow$ 任意一条 w-w 闭迹;

  $T\leftarrow T$ 和 T' 拼接;

  $G\leftarrow G-\{e\in E: T'$经过$e\}$;

输出 (T)



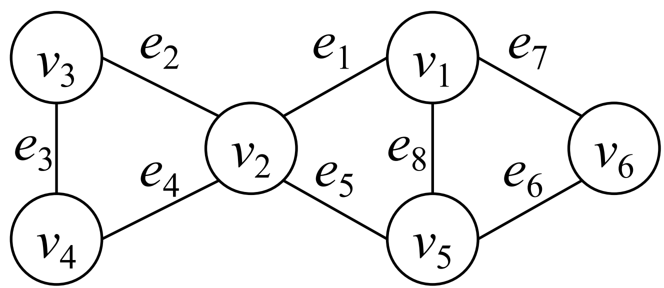
欧拉图

- 删除找到的闭迹

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G=\langle V,E\rangle$

if $\exists u,v\in V,\ 2\nmid \mathtt{d}(u)$ 且 $2\nmid \mathtt{d}(v)$ **then**
  $T\leftarrow$ 任意一条 u-v 迹;
else
  $T\leftarrow$ 任意一条闭迹;
$G\leftarrow G-\{e\in E: T\text{经过}e\}$;
while $\exists w\in V$, w 非孤立点且 T 经过 w **do**
  $T'\leftarrow$ 任意一条 w-w 闭迹;
  $T\leftarrow T$ 和 T' 拼接;
  $G\leftarrow G-\{e\in E: T'\text{经过}e\}$;
输出 (T)

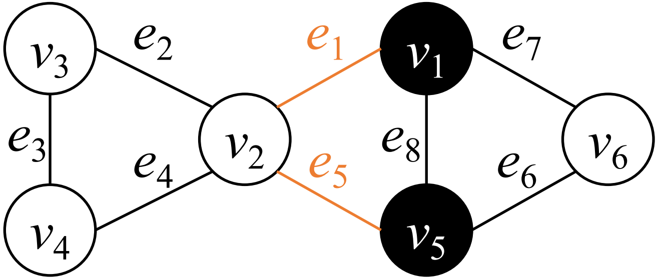


欧拉图

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V,\ 2 \nmid \mathtt{d}(u)$ 且 $2 \nmid \mathtt{d}(v)$ **then**
  $T \leftarrow$ 任意一条 u-v 迹;
else
  $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T\text{经过}e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
  $T' \leftarrow$ 任意一条 w-w 闭迹;
  $T \leftarrow T$ 和 T' 拼接;
  $G \leftarrow G - \{e \in E : T'\text{经过}e\}$;
输出 (T)

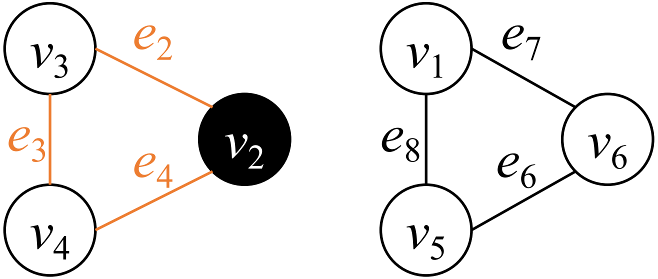


欧拉图

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V,\ 2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**
    $T \leftarrow$ 任意一条 u-v 迹;
else
    $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T$经过$e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
    $T' \leftarrow$ 任意一条 w-w 闭迹;
    $T \leftarrow T$ 和 T' 拼接;
    $G \leftarrow G - \{e \in E : T'$经过$e\}$;
输出 (T)



欧拉图

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V,\ 2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**

　$T \leftarrow$ 任意一条 u-v 迹;

else

　$T \leftarrow$ 任意一条闭迹;

$G \leftarrow G - \{e \in E : T$经过$e\}$;

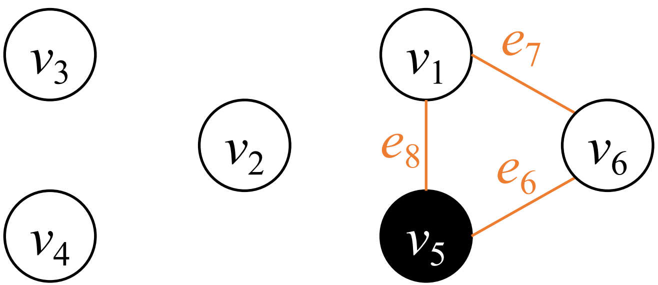
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**

　$T' \leftarrow$ 任意一条 w-w 闭迹;

　$T \leftarrow T$ 和 T' 拼接;

　$G \leftarrow G - \{e \in E : T'$经过$e\}$;

输出（T）

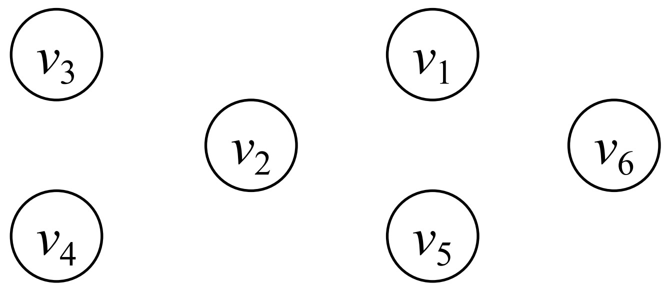


欧拉图

算法 6: 希尔霍尔策算法

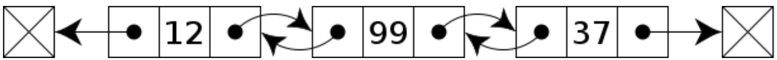
输入: 含欧拉迹的图 $G = \langle V, E\rangle$

if $\exists u, v \in V,\ 2 \nmid \mathtt{d}(u)$ 且 $2 \nmid \mathtt{d}(v)$ **then**
    $T \leftarrow$ 任意一条 u-v 迹;
else
    $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T$经过$e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
    $T' \leftarrow$ 任意一条 w-w 闭迹;
    $T \leftarrow T$ 和 T' 拼接;
    $G \leftarrow G - \{e \in E : T'$经过$e\}$;
输出 (T)



欧拉图

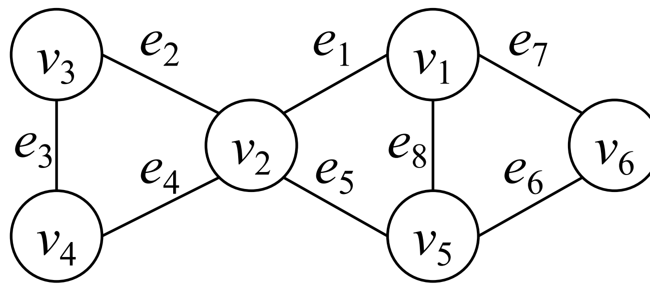
- 时间复杂度：$O(n+m)$
  - 假设采用双向链表等数据结构来存储迹、维护所有满足条件的w、维护每个w关联的T尚未经过的边



算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G=\langle V,E\rangle$

if $\exists u,v\in V,\ 2\nmid \mathrm{d}(u)$ 且 $2\nmid \mathrm{d}(v)$ **then**
  $T\leftarrow$ 任意一条 u-v 迹;
else
  $T\leftarrow$ 任意一条闭迹;
$G\leftarrow G-\{e\in E: T$经过$e\}$;
while $\exists w\in V$, w 非孤立点且 T 经过 w **do**
  $T'\leftarrow$ 任意一条 w-w 闭迹;
  $T\leftarrow T$ 和 T' 拼接;
  $G\leftarrow G-\{e\in E: T'$经过$e\}$;
输出（T）



https://upload.wikimedia.org/wikipedia/commons/thumb/5/5e/Doubly-linked-list.svg/610px-Doubly-linked-list.svg.png

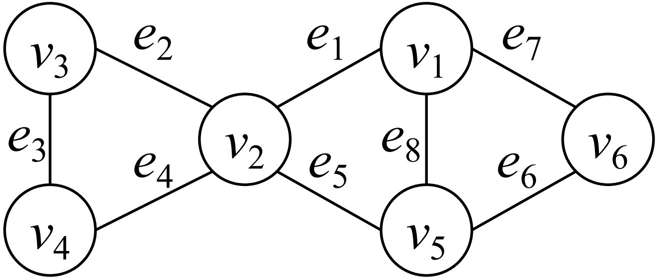
欧拉图

- 希尔霍尔策算法每一轮循环条件判定前，G没有顶点的度为奇数。

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E\rangle$

if $\exists u, v \in V$, $2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**
  $T \leftarrow$ 任意一条 u-v 迹;
else
  $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T经过e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
  $T' \leftarrow$ 任意一条 w-w 闭迹;
  $T \leftarrow T$ 和 T' 拼接;
  $G \leftarrow G - \{e \in E : T'经过e\}$;
输出（T）



欧拉图

- 希尔霍尔策算法每一轮循环条件判定前，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 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V$, $2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**

  $T \leftarrow$ 任意一条 u-v 迹;

else

  $T \leftarrow$ 任意一条闭迹;

$G \leftarrow G - \{e \in E : T$经过$e\}$;

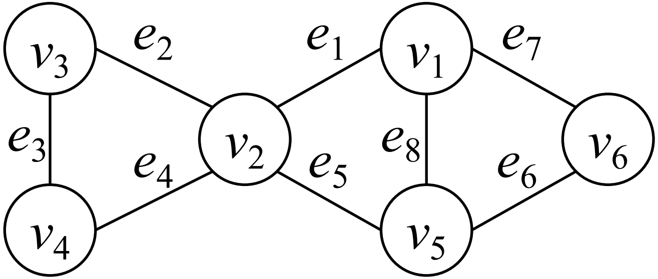
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**

  $T' \leftarrow$ 任意一条 w-w 闭迹;

  $T \leftarrow T$ 和 T' 拼接;

  $G \leftarrow G - \{e \in E : T'$经过$e\}$;

输出 (T)



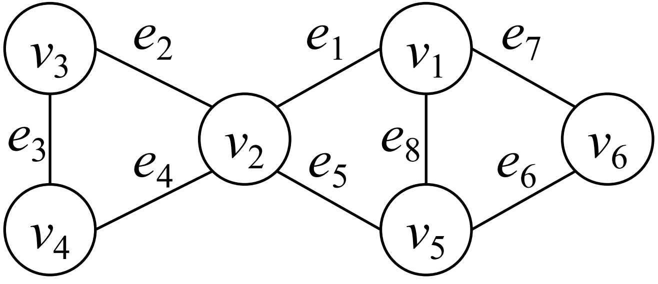
欧拉图

- 希尔霍尔策算法每一轮循环条件判定前，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。
- 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 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V$, $2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**
  $T \leftarrow$ 任意一条 u-v 迹;
else
  $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T$经过$e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
  $T' \leftarrow$ 任意一条 w-w 闭迹;
  $T \leftarrow T$ 和 T' 拼接;
  $G \leftarrow G - \{e \in E : T'$经过$e\}$;
输出（T）



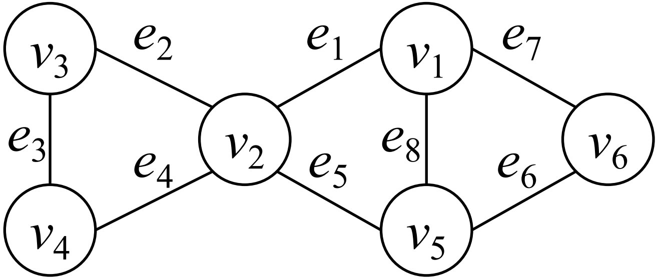
欧拉图

- 希尔霍尔策算法每一轮循环条件判定前，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。
- 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？
  - 根据顶点度的奇偶性，一定能走，且一定能走到v或走回u或w

算法 6: 希尔霍尔策算法

输入: 含欧拉迹的图 $G = \langle V, E \rangle$

if $\exists u, v \in V$, $2 \nmid \mathrm{d}(u)$ 且 $2 \nmid \mathrm{d}(v)$ **then**
    $T \leftarrow$ 任意一条 u-v 迹;
else
    $T \leftarrow$ 任意一条闭迹;
$G \leftarrow G - \{e \in E : T$经过$e\}$;
while $\exists w \in V$, w 非孤立点且 T 经过 w **do**
    $T' \leftarrow$ 任意一条 w-w 闭迹;
    $T \leftarrow T$ 和 T' 拼接;
    $G \leftarrow G - \{e \in E : T'$经过$e\}$;
输出 （T）



本次课的主要内容

3.1 圈和树

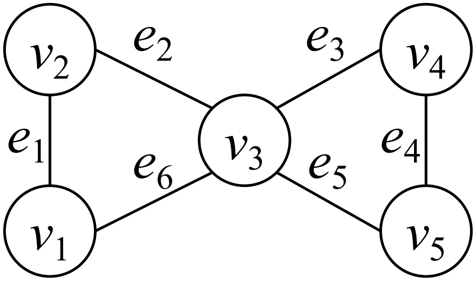
3.2 二分图

3.3 欧拉图

3.4 哈密尔顿图

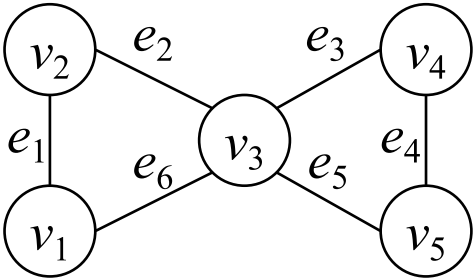
哈密尔顿图

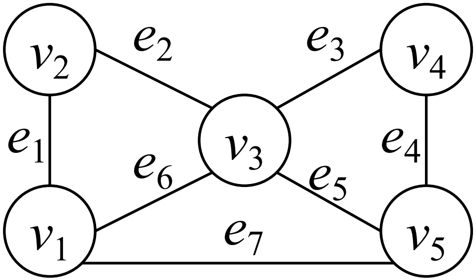
- 哈密尔顿路：经过图的每个顶点的路



哈密尔顿图

- 哈密尔顿路：经过图的每个顶点的路
- 哈密尔顿圈：经过图的每个顶点的圈
- 哈密尔顿图：含哈密尔顿圈的图

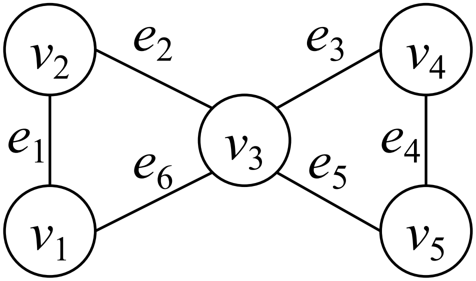


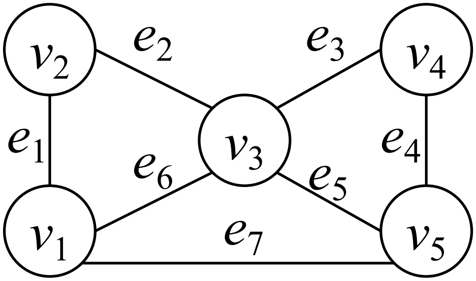


哈密尔顿图

- 哈密尔顿路：经过图的每个顶点的路
- 哈密尔顿圈：经过图的每个顶点的圈
- 哈密尔顿图：含哈密尔顿圈的图

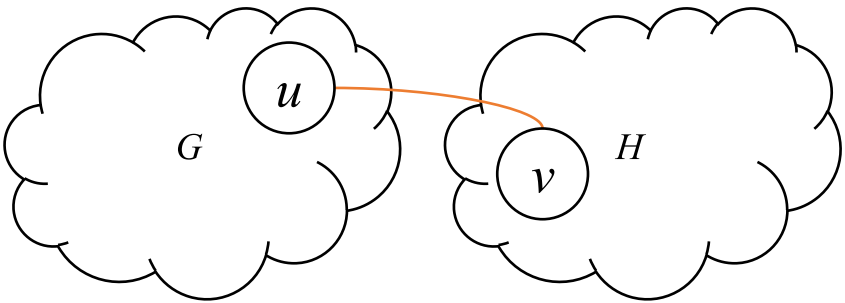
- 完全图是哈密尔顿图吗？完全二分图是哈密尔顿图吗？





哈密尔顿图

- 对于哈密尔顿图$G = <V_G, E_G>$和$H = <V_H, E_H>$，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？

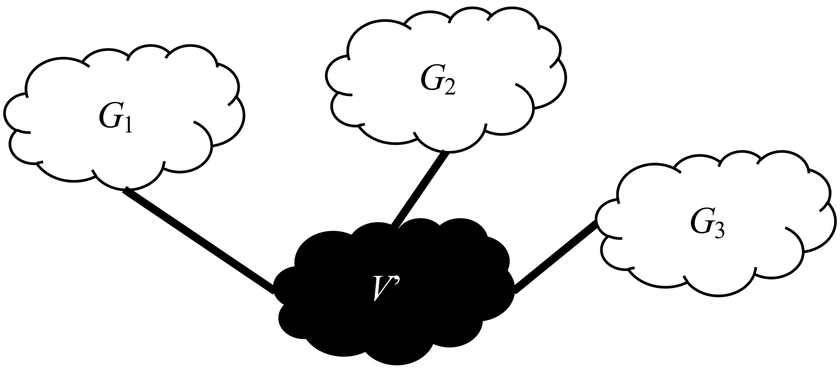
哈密尔顿图

- 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？

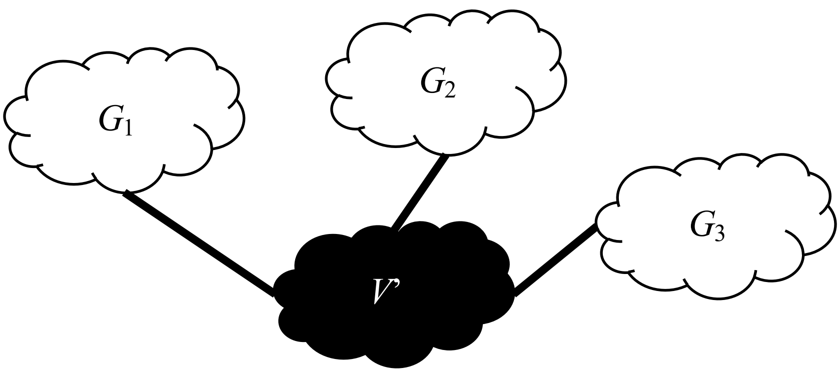
- 若图$G = <V, E>$是哈密尔顿图，则对于任意$\emptyset \subseteq V' \subseteq V$，$G - V'$含至多$|V'|$个连通分支。



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$G = <V, E>$是哈密尔顿图，则对于任意$\emptyset \subseteq V' \subseteq V$，$G - V'$含至多$|V'|$个连通分支。
  - 若图G连通，该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？

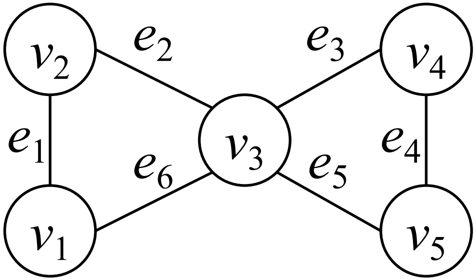
- 若图$G = <V, E>$是哈密尔顿图，则对于任意$\varnothing \subseteq V' \subseteq V$，$G - V'$含至多$|V'|$个连通分支。
  - 若图G连通，该必要条件是充分条件吗？
- 对于阶为n的图$G = <V, E>$，若$n \geq 3$且任意两个不相邻顶点$u,v \in V$都满足$\mathrm{d}(u) + \mathrm{d}(v) \geq n$，则$G$为哈密尔顿图。
  - 该充分条件是必要条件吗？

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？

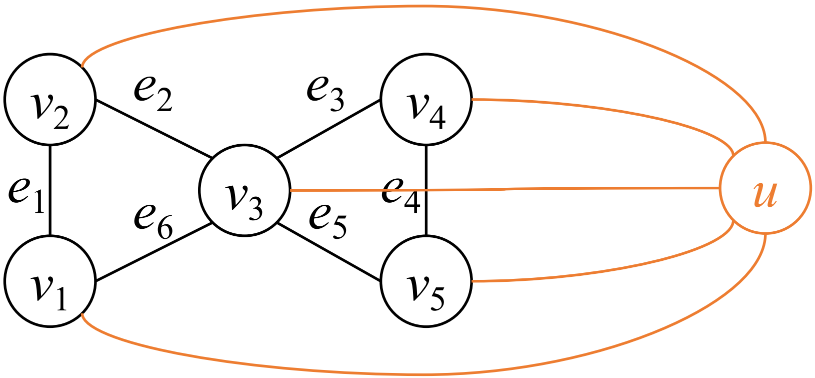
哈密尔顿图

- 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TSP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

实战应用

通信系统由若干台设备以及设备间的线缆组成，只和其它一台设备有线缆连接的设备称作“终端”。若某通信系统中的任意两台设备均可直接或间接通信，并且信道唯一，则该通信系统称作“最简系统”。

练习 3.4. *从含至少 3 台设备的最简系统中去除所有终端，剩余部分仍是最简系统吗？*

最简系统中，两台设备间信道经过的线缆数量称作“跳数”，某设备到其它所有设备的最远跳数称作该设备的“边缘程度”。

练习 3.5. *对于含至少 2 台设备的最简系统中的任意设备，与之跳数最远的设备一定是一台终端吗？*

练习 3.6. *对于含至少 3 台设备的最简系统，请比较终端及其相连的那台设备的边缘程度。*

最简系统中，边缘程度最小的设备称作“核心设备”。

练习 3.7. *满足下述要求的最简系统存在吗？若存在，请给出一个例子；若不存在，请证明。*

1. *含至少 3 台设备，且其中只有 1 台核心设备。*
2. *含至少 3 台设备，且其中恰有 2 台核心设备。*
3. *含至少 3 台设备，且其中有至少 3 台核心设备。*

实战应用

某国若干（但非全部）城市的主导产业是旅游业，旅游城市和非旅游城市间可申请开通旅游客运专线。该国若干（但非全部）城市种植粮食，产粮城市和非产粮城市间可申请开通粮食货运快线。有趣的是，该国任意两座城市间，或者开通了旅游客运专线，或者开通了粮食货运快线，但不兼有两者。

练习 3.10. 若该国共有 *2* 座城市，这种情况有可能存在吗？若有可能，请给出一个例子；若不可能，请证明。

练习 3.11. 若该国共有 *3* 座城市，这种情况有可能存在吗？若有可能，请给出一个例子；若不可能，请证明。

练习 3.12. 若该国共有 *4* 座城市，这种情况有可能存在吗？若有可能，请给出一个例子；若不可能，请证明。

练习 3.13. 若该国至少有 *5* 座城市，这种情况有可能存在吗？若有可能，请给出一个例子；若不可能，请证明。

实战应用

练习 3.23. 某国有 $2n$ 座城市 $(n \geq 2)$，任意两座城市间，或者开通了直飞航线，或者开通了直达班车，但不兼有两者。若每座城市开通直飞航线的数量相同，则是否有一位旅游爱好者可以全程只乘坐一种交通工具不间断、不重复地环游所有城市并回到出发城市？

练习 3.24. 若干位同学围圆桌而坐，若通过换座位可使每位同学身旁的同学完全不同，则至少有多少位同学？