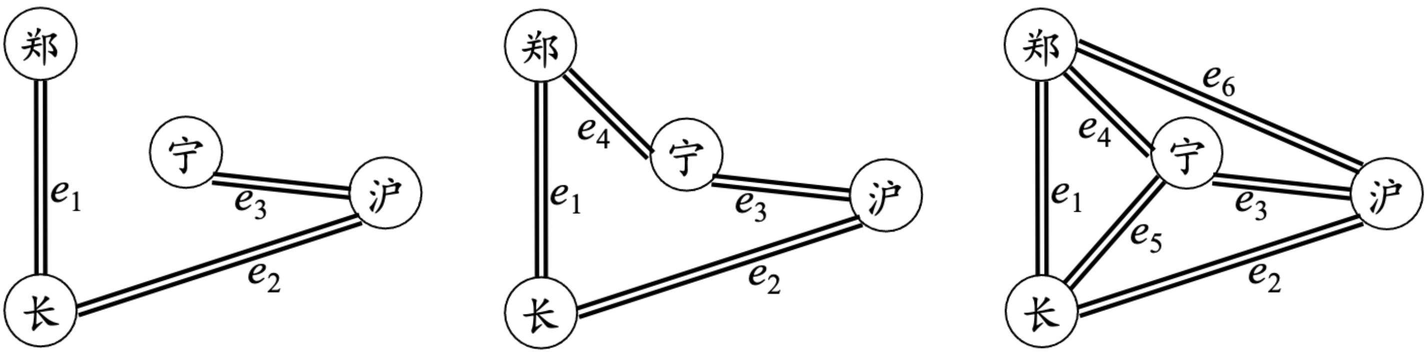
# 4 连通度

程龚

# 连通的强度



# 本次课的主要内容

4.1 块

4.2 割集和连通度

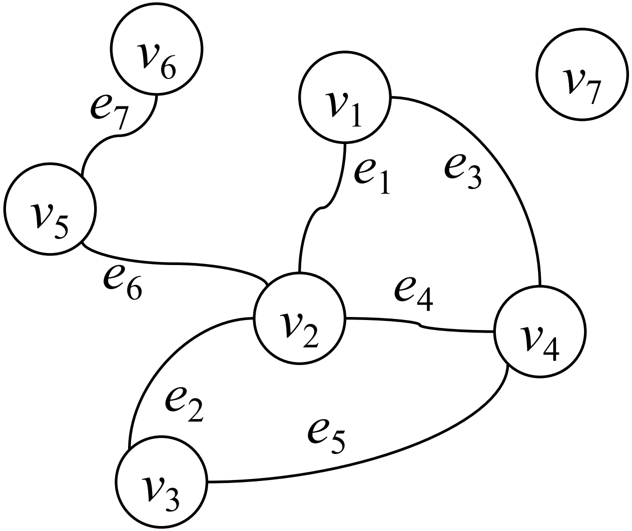
# 本次课的主要内容

4.1 块

4.2 割集和连通度

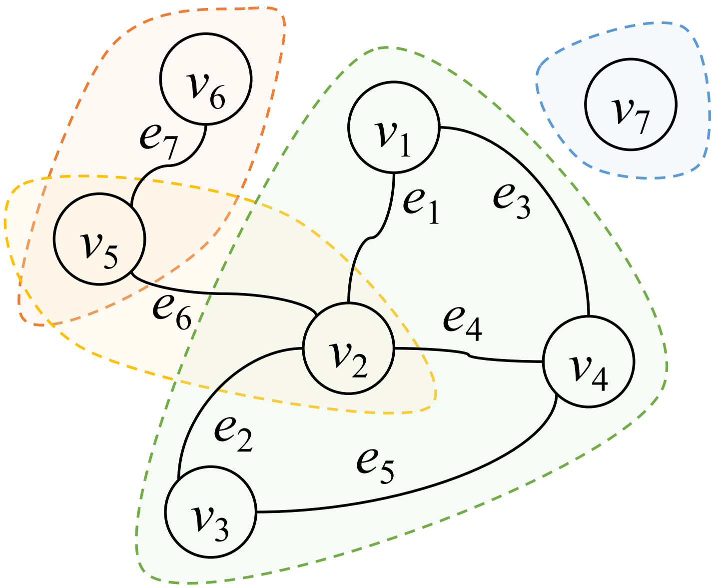
# 块

- **块**：极大的没有割点的连通子图



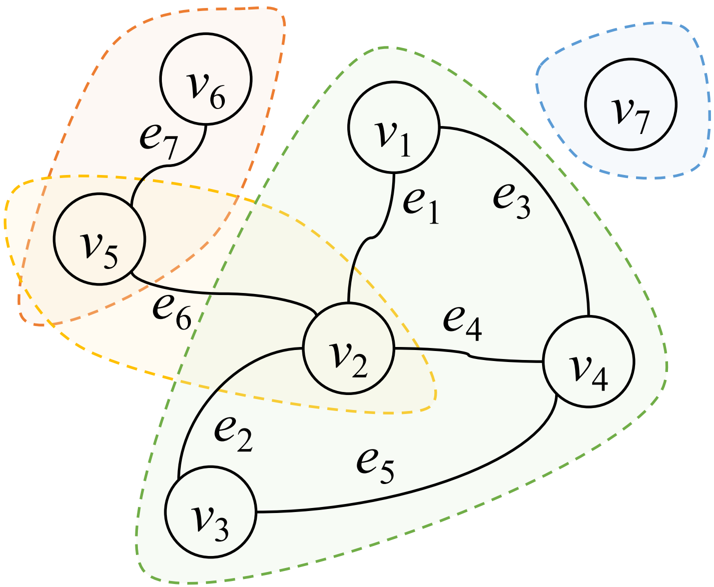
# 块

- **块**：极大的没有割点的连通子图



# 块

- **块**：极大的没有割点的连通子图
  - 若$G$只含1个块，即$G$连通且没有割点，则G自身称作一个块



# 块

- 完全图是块吗？

# 块

- 完全图是块吗？
- 树是块吗？

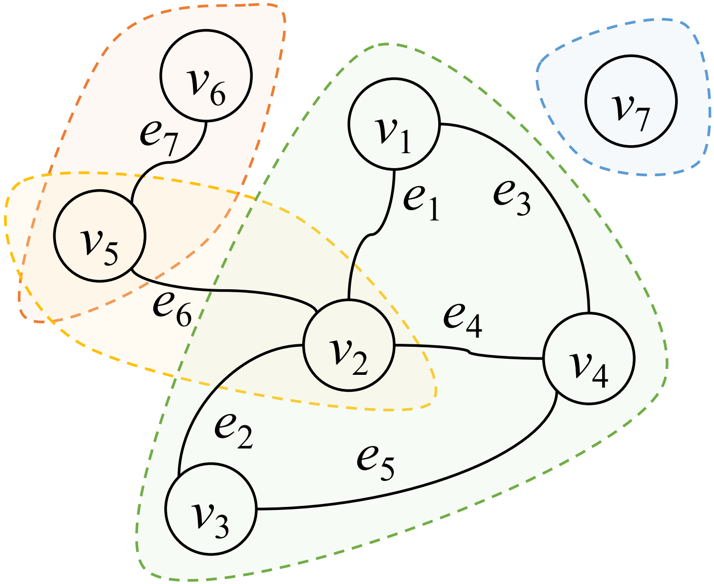
# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，
  这种顶点有什么特征？

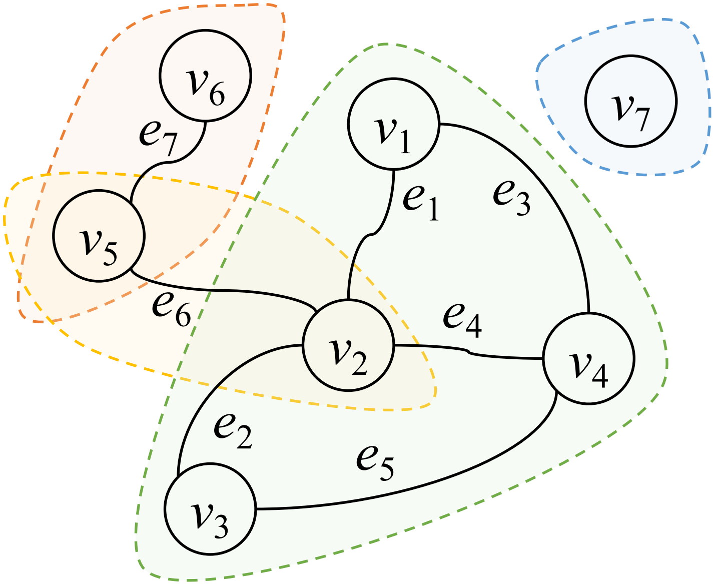


# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，这种顶点有什么特征？
- 若图的块只含一条边，这种边有什么特征？（iff）

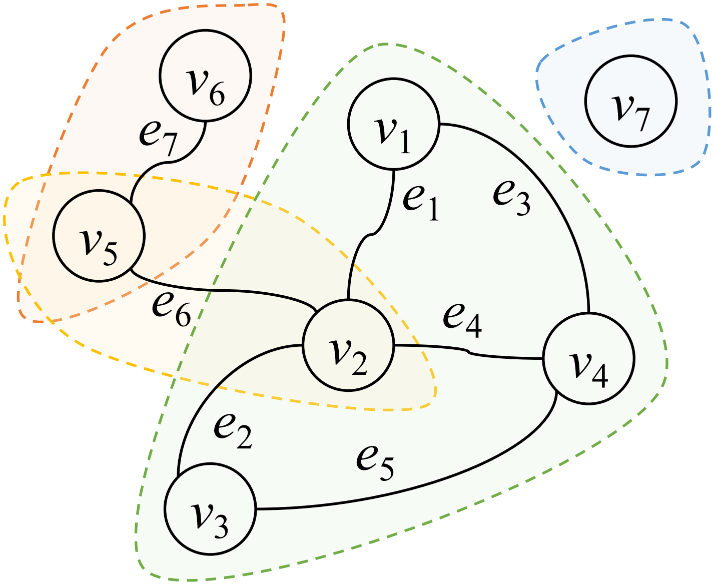
随堂小测



# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，
  这种顶点有什么特征？
- 若图的块只含一条边，
  这种边有什么特征？（iff）
  - 非割边为什么不行？
  - 割边为什么满足极大性？

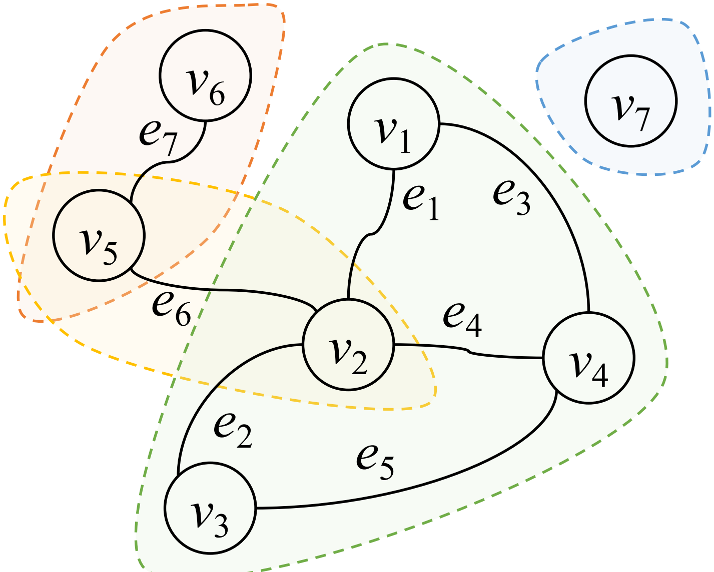


# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，这种顶点有什么特征？
- 若图的块只含一条边，这种边有什么特征？（iff）

- 两个块至多含几个公共顶点？这种顶点有什么特征？

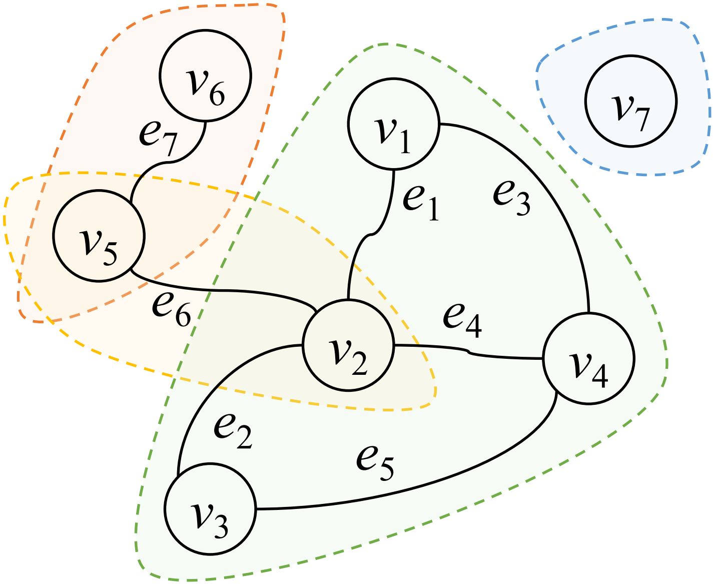


# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，这种顶点有什么特征？
- 若图的块只含一条边，这种边有什么特征？（iff）

- 两个块至多含几个公共顶点？这种顶点有什么特征？
- 两个块至多含几条公共边？

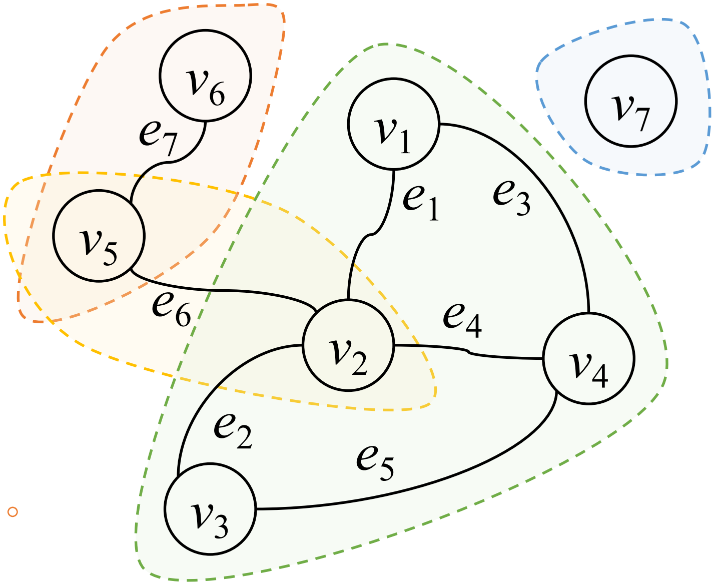


# 块

- 完全图是块吗？
- 树是块吗？
- 欧拉图和哈密尔顿图是块吗？

- 若图的块只含一个顶点，
  这种顶点有什么特征？
- 若图的块只含一条边，
  这种边有什么特征？（iff）

- 两个块至多含几个公共顶点？
  这种顶点有什么特征？
- 两个块至多含几条公共边？
- 块为边集定义了一种等价关系。
  - 划分是什么？



# 块

- 对于阶至少为3的连通图$G$，以下是块的等价定义
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。
  7. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路经过$w$。
  8. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路不经过$w$。

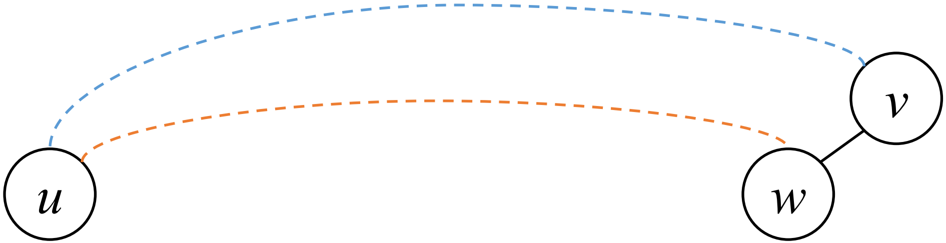
# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含两条无公共内顶点的$u$-$v$路。

# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

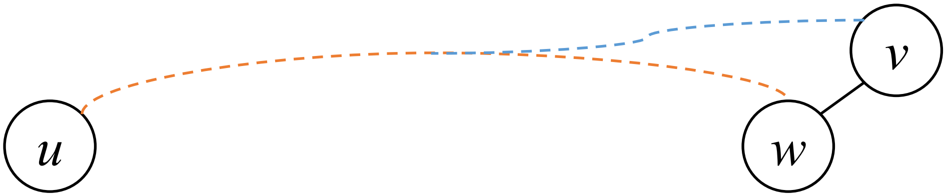
- 删去$v$的邻点$w$，仍存在其它的$u$-$v$路，与经过$w$的$u$-$v$路组成圈。
  这样证明存在什么问题？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

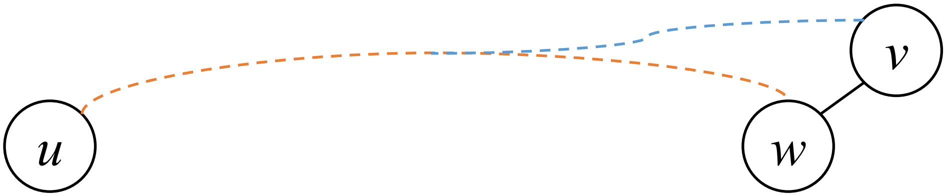
- 删去$v$的邻点$w$，仍存在其它的$u$-$v$路，与经过$w$的$u$-$v$路组成圈。这样证明存在什么问题？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

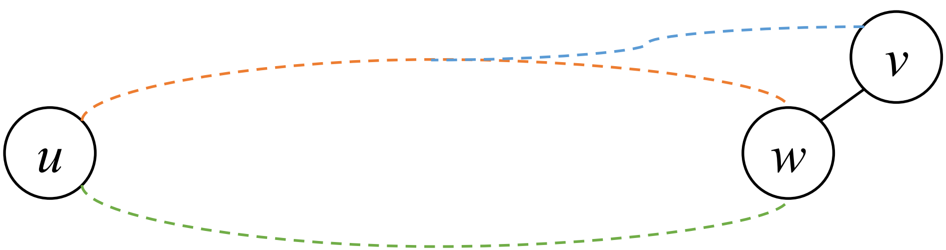
- 删去$v$的邻点$w$，仍存在其它的$u$-$v$路，与经过$w$的$u$-$v$路组成圈。这样证明存在什么问题？如何解决这个问题？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G=<V, E>$和任意两个顶点$u, v \in V$， $G$含两条无公共内顶点的$u$-$v$路。

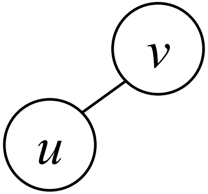
- 删去$v$的邻点$w$，仍存在其它的$u$-$v$路，与经过$w$的$u$-$v$路组成圈。
  这样证明存在什么问题？如何解决这个问题？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

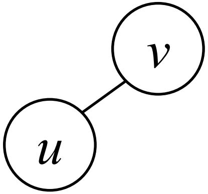
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\mathrm{dist}(u, v) = 1$时，为什么成立？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

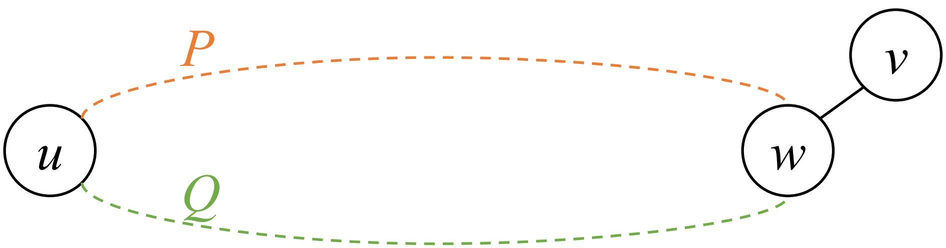
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\text{dist}(u, v) = 1$时，为什么成立？
    - $G$没有割点 → 没有割边 → $u$和$v$共圈



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = \langle V, E\rangle$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

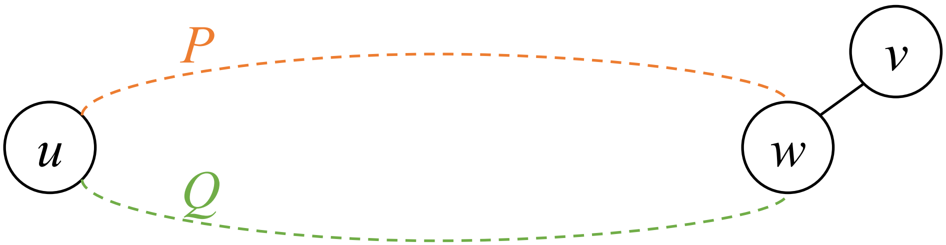
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\mathrm{dist}(u, v) = 1$时，为什么成立？
    – $G$没有割点 → 没有割边 → $u$和$v$共圈
  - 假设$\mathrm{dist}(u, v) = k$时成立，则$\mathrm{dist}(u, v) = k + 1$时
    – $\mathrm{dist}(u, w) = k$ → 共圈



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

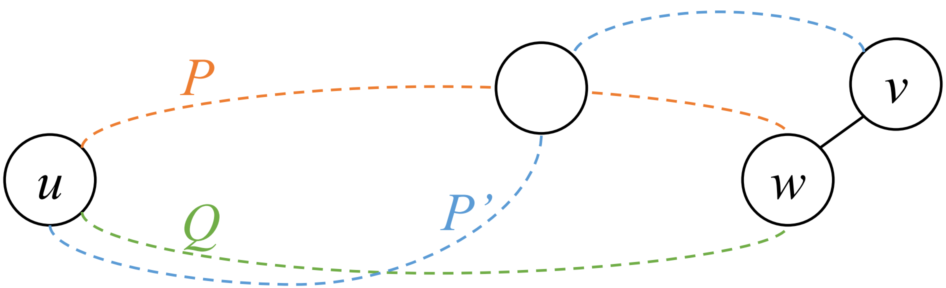
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\text{dist}(u, v) = 1$时，为什么成立？
    - $G$没有割点 → 没有割边 → $u$和$v$共圈
  - 假设$\text{dist}(u, v) = k$时成立，则$\text{dist}(u, v) = k + 1$时
    - $\text{dist}(u, w) = k$ → 共圈
    - 若这个圈经过$v$？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

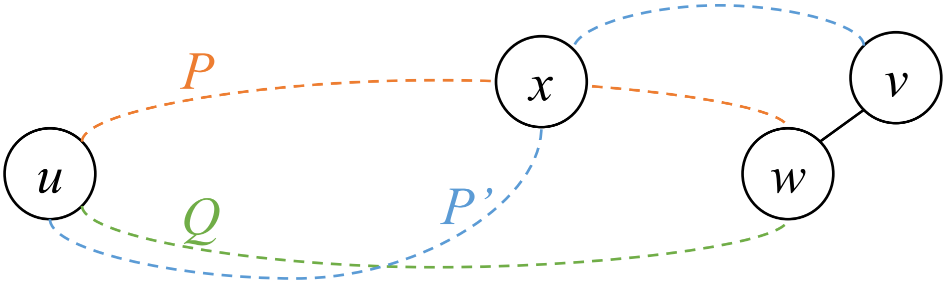
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\text{dist}(u, v) = 1$时，为什么成立？
    - $G$没有割点 → 没有割边 → $u$和$v$共圈
  - 假设$\text{dist}(u, v) = k$时成立，则$\text{dist}(u, v) = k + 1$时
    - $\text{dist}(u, w) = k$ → 共圈
    - 不经过$w$的$u$-$v$路与上述圈有公共顶点怎么办？



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

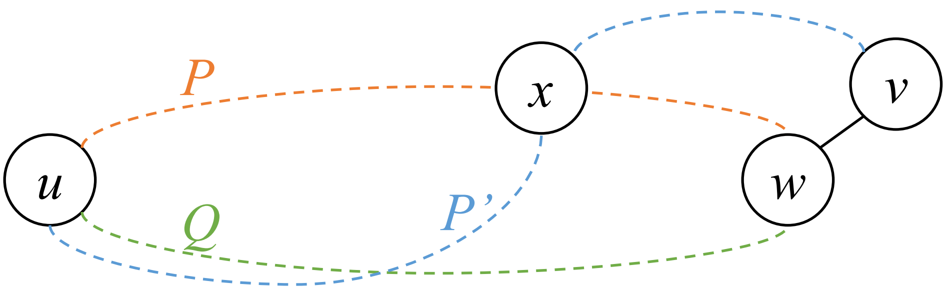
- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\text{dist}(u, v) = 1$时，为什么成立？
    - $G$没有割点 → 没有割边 → $u$和$v$共圈
  - 假设$\text{dist}(u, v) = k$时成立，则$\text{dist}(u, v) = k + 1$时
    - $\text{dist}(u, w) = k$ → 共圈
    - $x$：不经过$w$的$u$-$v$路与上述圈的最后一个公共顶点



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = \langle V, E \rangle$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。

- 采用数学归纳法，对两个顶点间的距离归纳。
  - 当$\mathrm{dist}(u, v) = 1$时，为什么成立？
    - $G$没有割点 → 没有割边 → $u$和$v$共圈
  - 假设$\mathrm{dist}(u, v) = k$时成立，则$\mathrm{dist}(u, v) = k + 1$时
    - $\mathrm{dist}(u, w) = k$ → 共圈
    - $x$：不经过$w$的$u$-$v$路与上述圈的最后一个公共顶点
    - 两条无公共内顶点的$u$-$v$路：$P$中的$u$-$x$路 拼接 $P'$中的$x$-$v$路，$Q$ 拼接 $(w, v)$



# 块

- 对于阶至少为3的连通图$G$
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含两条无公共内顶点的$u$-$v$路。

- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含两条无公共内顶点的$u$-$v$路。
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含圈经过$u$和$v$。

- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含两条无公共内顶点的$u$-$v$路。
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$， $G$含圈经过$u$和$v$。

- 你能自己证明吗？

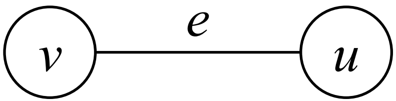
# 块

- 对于阶至少为3的连通图$G$
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。

# 块

- 对于阶至少为3的连通图$G$
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。

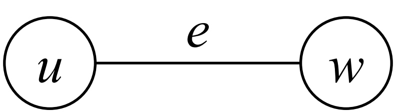
- 若$v$是$e$的端点？

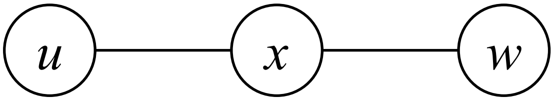


# 块

- 对于阶至少为3的连通图$G$
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。

- 若$v$是$e$的端点？
- 若$v$不是$e$的端点
  - 对$e$剖分，得到$G'$ → $G'$也是块 → $v$和$x$共圈 → $v$和$e$共圈



# 块

- 对于阶至少为3的连通图$G$
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。

- 你能自己证明吗？

# 块

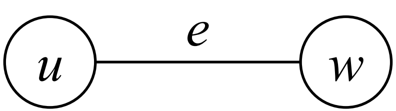
- 对于阶至少为3的连通图$G$
  4. 对于图$G=<V,E>$、任意一个顶点$v\in V$、任意一条边$e\in E$，$G$含圈经过$v$和$e$。
  5. 对于图$G=<V,E>$和任意两条边$e,f\in E$，$G$含圈经过$e$和$f$。

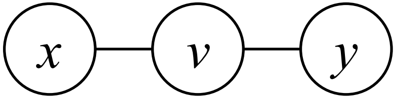
- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$
  4. 对于图$G=<V,E>$、任意一个顶点$v\in V$、任意一条边$e\in E$，$G$含圈经过$v$和$e$。
  5. 对于图$G=<V,E>$和任意两条边$e,f\in E$，$G$含圈经过$e$和$f$。

- 你能自己证明吗？
  - 对$f$剖分



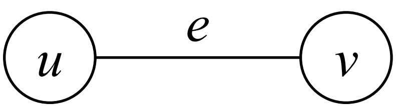
# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。

# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$， $G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$ 、任意两个顶点$u, v \in V$ 、任意一条边$e \in E$， $G$含$u$-$v$路经过$e$ 。

- 若$u$和$v$是$e$的端点？



# 块

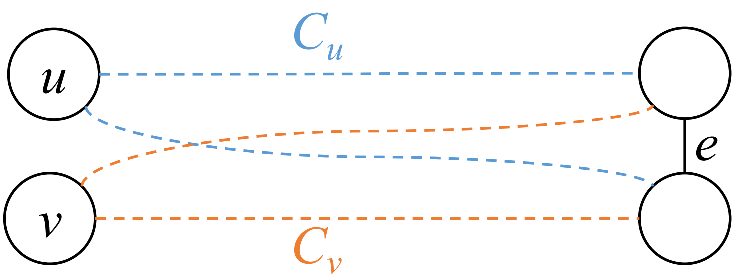
- 对于阶至少为3的连通图$G$
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。

- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？

# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。

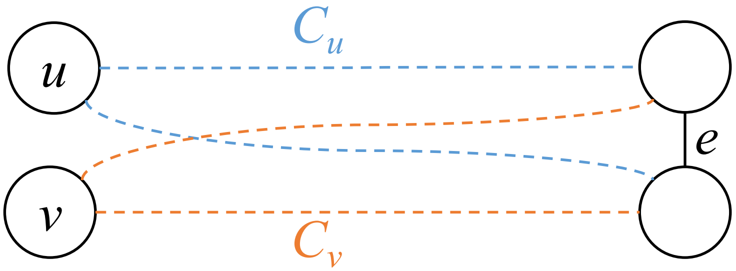
- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？
- 若$u$和$v$不是$e$的端点



# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G=<V, E>$和任意两条边$e, f\in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G=<V, E>$、任意两个顶点$u, v\in V$、任意一条边$e\in E$，$G$含$u$-$v$路经过$e$。

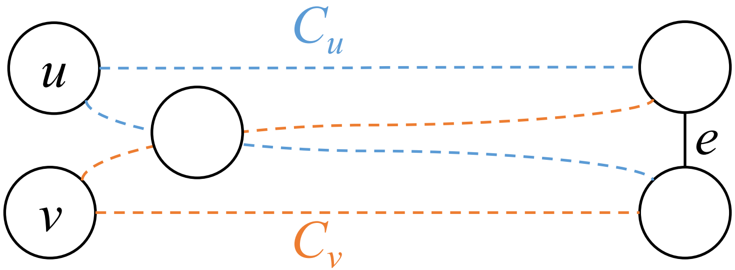
- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？
- 若$u$和$v$不是$e$的端点
  - 若$C_u$经过$v$或$C_v$经过$u$？



# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G=<V,E>$和任意两条边$e,f\in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G=<V,E>$、任意两个顶点$u,v\in V$、任意一条边$e\in E$，$G$含$u$-$v$路经过$e$。

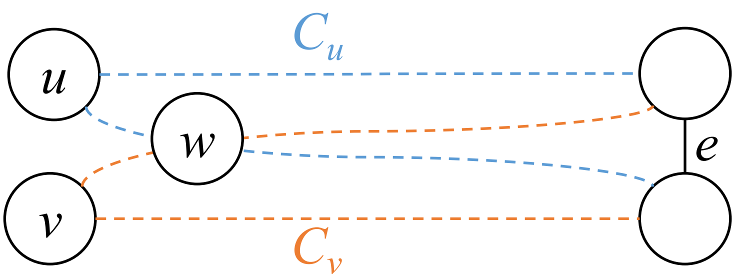
- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？
- 若$u$和$v$不是$e$的端点
  - $C_u$和$C_v$有公共顶点怎么办？



# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。

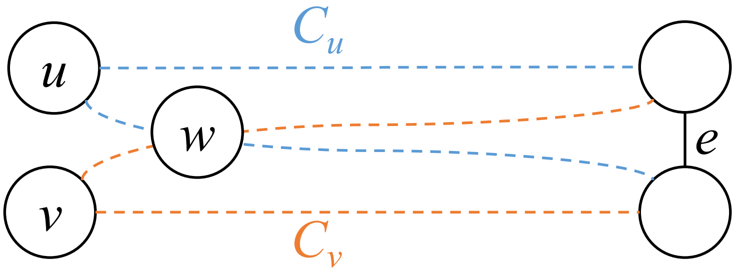
- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？
- 若$u$和$v$不是$e$的端点
  - $w$：$C_u$和$C_v$的公共顶点中，距离$u$最近的一个



# 块

- 对于阶至少为3的连通图$G$
  5. 对于图$G=<V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G=<V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。

- 若$u$和$v$是$e$的端点？
- 若$u$是$e$的端点而$v$不是？
- 若$u$和$v$不是$e$的端点
  - $w$：$C_u$和$C_v$的公共顶点中，距离$u$最近的一个
  - 经过$e$的$u$-$v$路：$C_u$中内顶点不被$C_v$经过的$u$-$w$路 拼接 $C_v$中经过$e$的$w$-$v$路



# 块

- 对于阶至少为3的连通图$G$
  6. 对于图$G = <V, E>$ 、任意两个顶点$u, v \in V$ 、任意一条边$e \in E$， $G$含$u$-$v$路经过$e$ 。
  7. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$， $G$含$u$-$v$路经过$w$。

- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$
  7. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$， $G$含$u$-$v$路经过$w$。
  8. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$， $G$含$u$-$v$路不经过$w$ 。

- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$
  8. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$， $G$含$u$-$v$路不经过$w$ 。
  1. 图$G$是块。

- 你能自己证明吗？

# 块

- 对于阶至少为3的连通图$G$，以下是块的等价定义
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。
  7. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路经过$w$。
  8. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路不经过$w$。

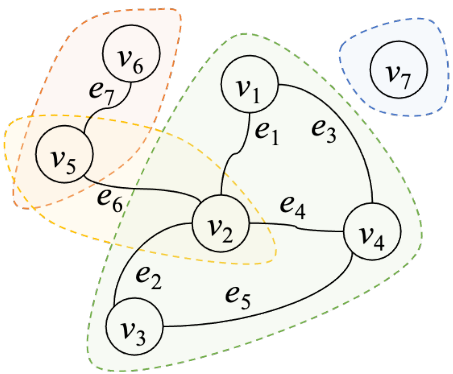
# 块

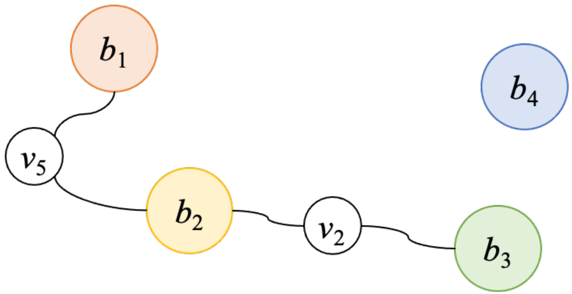
- 对于阶至少为3的连通图$G$，以下是块的等价定义
  1. 图$G$是块。
  2. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含两条无公共内顶点的$u$-$v$路。
  3. 对于图$G = <V, E>$和任意两个顶点$u, v \in V$，$G$含圈经过$u$和$v$。
  4. 对于图$G = <V, E>$、任意一个顶点$v \in V$、任意一条边$e \in E$，$G$含圈经过$v$和$e$。
  5. 对于图$G = <V, E>$和任意两条边$e, f \in E$，$G$含圈经过$e$和$f$。
  6. 对于图$G = <V, E>$、任意两个顶点$u, v \in V$、任意一条边$e \in E$，$G$含$u$-$v$路经过$e$。
  7. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路经过$w$。
  8. 对于图$G = <V, E>$和任意三个顶点$u, v, w \in V$，$G$含$u$-$v$路不经过$w$。

- 块为边集定义了一种等价关系，这种等价关系的内涵是什么？

# 块

- **块-割点图**
  - 二分图

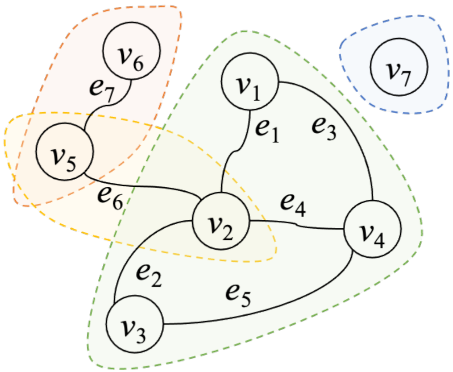


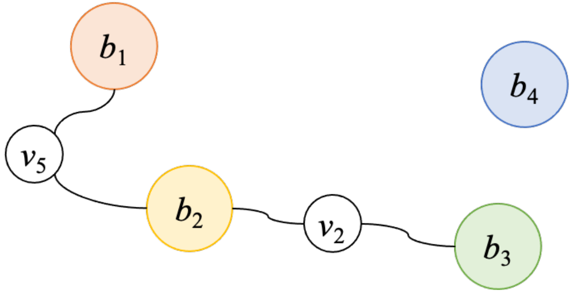



# 块

- 块-割点图
  - 二分图

- 块-割点图含圈吗？

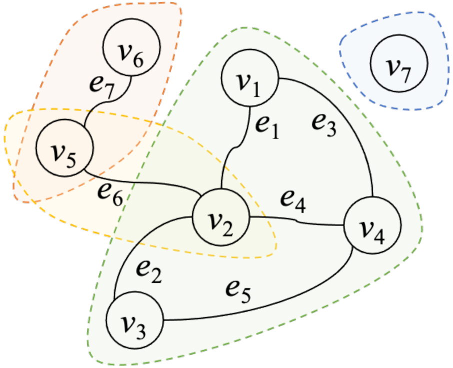


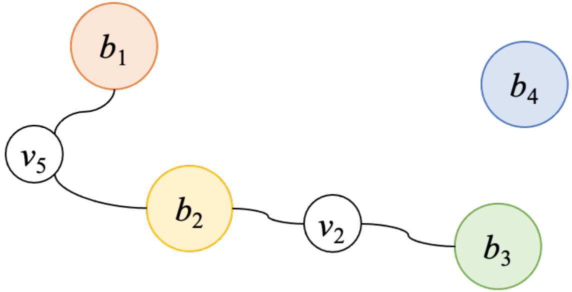



# 块

- 块-割点图
  - 二分图

- 块-割点图含圈吗？
- 对于图$G$的块-割点图$H$，$H$的叶顶点有可能是$G$的割点吗？

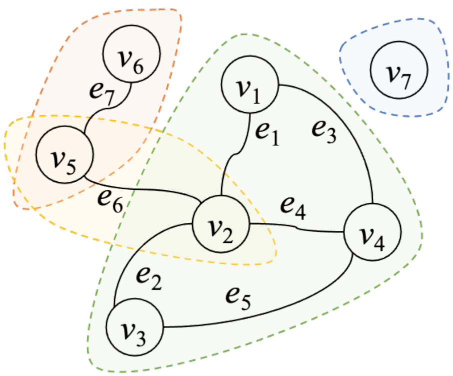


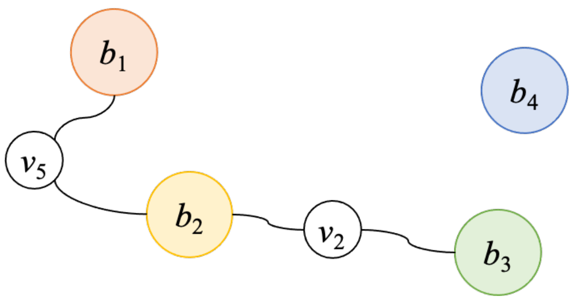



# 块

- 块-割点图
  - 二分图

- 块-割点图含圈吗？
- 对于图$G$的块-割点图$H$，$H$的叶顶点有可能是$G$的割点吗？
- 对于图$G = <V, E>$和顶点$v \in V$，$v$是$G$的割点当且仅当$G$的至少2个块含$v$。



# 块

- 如何找出图中的所有块？

# 块

- John Hopcroft，1939-，出生于美国，1986年获图灵奖
- Robert Tarjan，1948-，出生于美国，1986年获图灵奖

https://en.wikipedia.org/wiki/John_Hopcroft
https://en.wikipedia.org/wiki/Robert_Tarjan

# 块

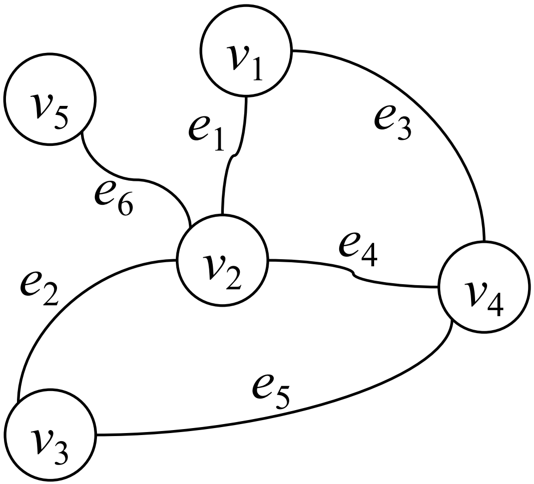
- 修改DFSCV算法：将发现的每条边入栈

**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow$ $time$;<br>
$u$.low $\leftarrow$ $u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
 **if** $v.visited = false$ **then**<br>
  入栈 ($S$, $(u, v)$);<br>
  $v$.parent $\leftarrow$ $u$;<br>
  $u$.children $\leftarrow$ $u$.children + 1;<br>
  DFSBlk($G$, $v$);<br>
  $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
  **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**<br>
   输出 (以下边组成一个块);<br>
   **do**<br>
    $(x, y) \leftarrow$ 出栈 ($S$);<br>
    输出 ($(x, y)$);<br>
   **while** $(x, y) \neq (u, v)$;<br>
 **else if** $v \neq u.parent$ **then**<br>
  入栈 ($S$, $(u, v)$);<br>
  $u$.low $\leftarrow$ min{$u$.low, $v$.d};



# 块

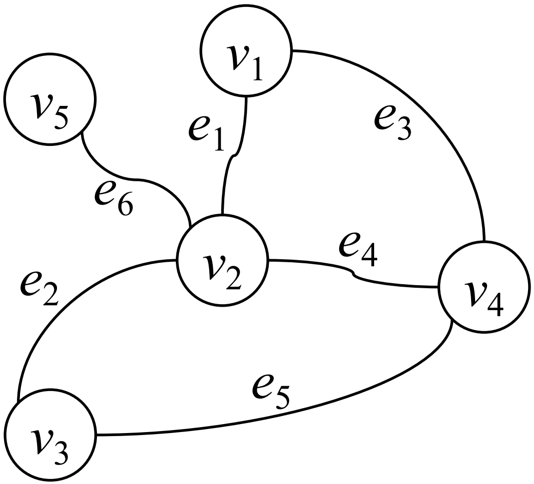
- 修改DFSCV算法：割点判定改为出栈一个块的边集

**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow time$;<br>
$u$.low $\leftarrow u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
    **if** $v.visited = false$ **then**<br>
        入栈 ($S$, $(u, v)$);<br>
        $v$.parent $\leftarrow u$;<br>
        $u$.children $\leftarrow u$.children + 1;<br>
        DFSBlk($G$, $v$);<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
        **if** *($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$)* **then**<br>
            输出 (以下边组成一个块);<br>
            **do**<br>
                $(x, y) \leftarrow$ 出栈 ($S$);<br>
                输出 ($(x, y)$);<br>
            **while** $(x, y) \neq (u, v)$;<br>
    **else if** $v \neq u.parent$ **then**<br>
        入栈 ($S$, $(u, v)$);<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.d};



# 块

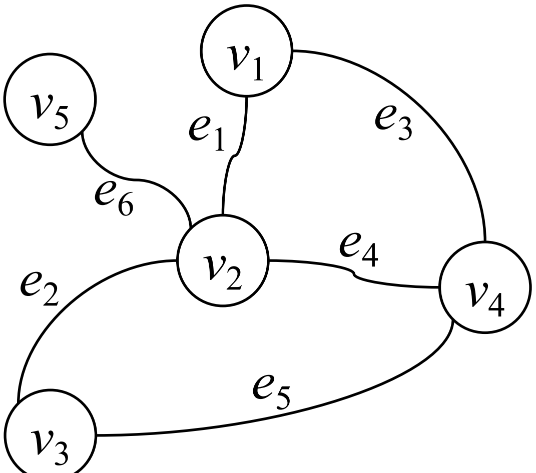
- 修改DFSCV算法：若结束时栈非空，出栈这个块的边集

**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

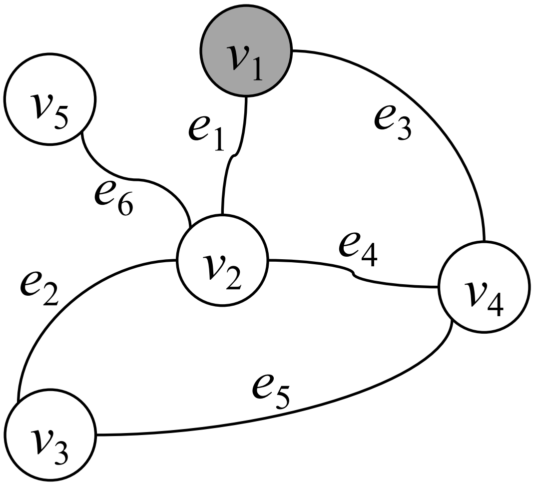
## ■ DFSBlk算法

**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow$ $time$;<br>
$u$.low $\leftarrow$ $u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
    **if** $v.visited = false$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $v$.parent $\leftarrow$ $u$;<br>
        $u$.children $\leftarrow$ $u$.children + 1;<br>
        `DFSBlk`($G$, $v$);<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
        **if** *($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$)* **then**<br>
            输出（以下边组成一个块）;<br>
            **do**<br>
                $(x, y) \leftarrow$ 出栈（$S$）;<br>
                输出（$(x, y)$）;<br>
            **while** $(x, y) \neq (u, v)$;<br>
    **else if** $v \neq u.parent$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.d};



# 块

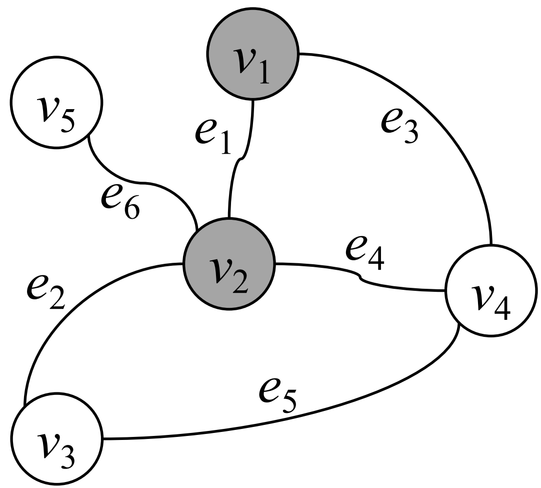
- DFSBlk算法

**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow$ $time$;<br>
$u$.low $\leftarrow$ $u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
  **if** $v.visited = false$ **then**<br>
    入栈（$S$, $(u, v)$）;<br>
    $v$.parent $\leftarrow$ $u$;<br>
    $u$.children $\leftarrow$ $u$.children + 1;<br>
    `DFSBlk`($G$, $v$);<br>
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
    **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**<br>
      输出（以下边组成一个块）;<br>
      **do**<br>
        $(x, y) \leftarrow$ 出栈（$S$）;<br>
        输出（$(x, y)$）;<br>
      **while** $(x, y) \neq (u, v)$;<br>
  **else if** $v \neq u.parent$ **then**<br>
    入栈（$S$, $(u, v)$）;<br>
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};

# 块

- **DFSBlk算法**

---

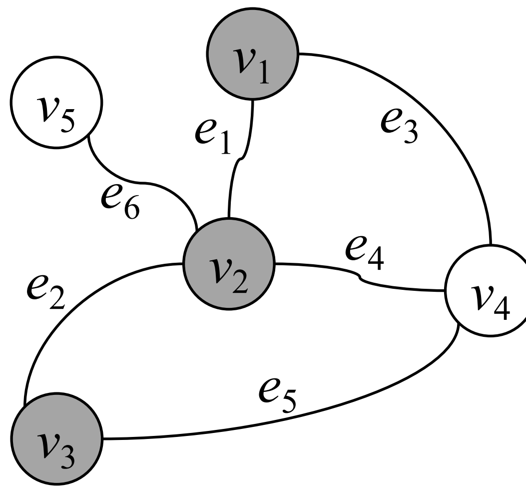
**算法 7:** DFSBlk

---

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;
$u$.d $\leftarrow$ $time$;
$u$.low $\leftarrow$ $u$.d;
$u$.visited $\leftarrow$ true;
**foreach** $(u, v) \in E$ **do**
  **if** $v.visited = false$ **then**
    入栈（$S$, $(u, v)$）;
    $v$.parent $\leftarrow$ $u$;
    $u$.children $\leftarrow$ $u$.children + 1;
    `DFSBlk`($G$, $v$);
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};
    **if** *($u.parent$ = null 且 $u.children \geq 2$) 或 ($u.parent \neq$ null 且 $v.low \geq u.d$)* **then**
      输出（以下边组成一个块）;
      **do**
        $(x, y) \leftarrow$ 出栈（$S$）;
        输出（$(x, y)$）;
      **while** $(x, y) \neq (u, v)$;
  **else if** $v \neq u.parent$ **then**
    入栈（$S$, $(u, v)$）;
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};

# 块

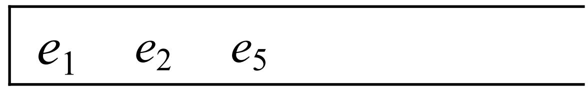
- **DFSBlk算法**

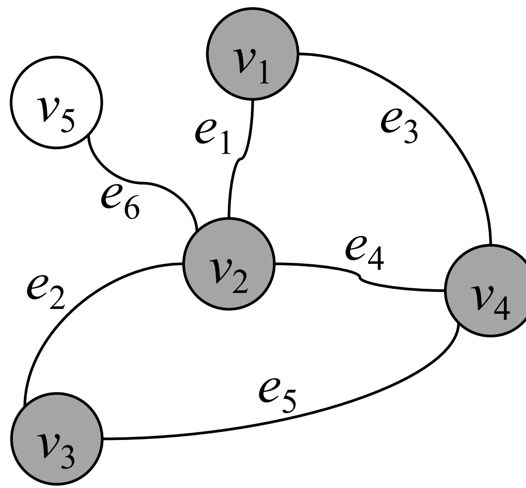
**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

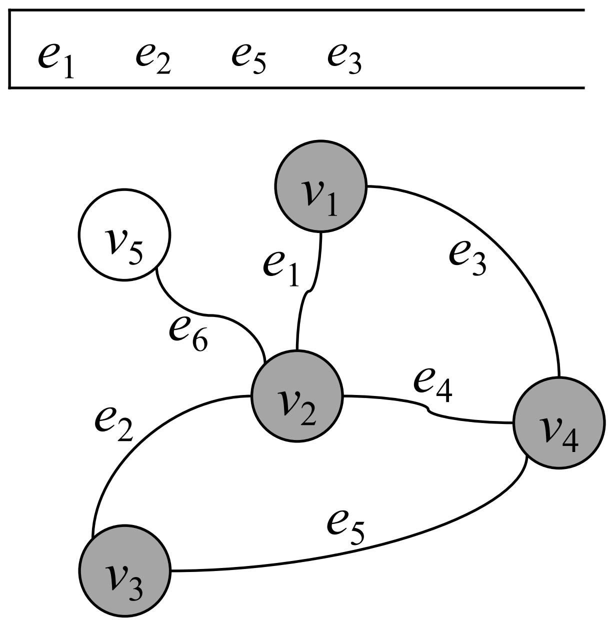
## ■ DFSBlk算法

**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

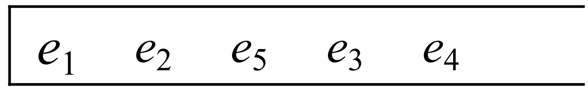
## DFSBlk算法

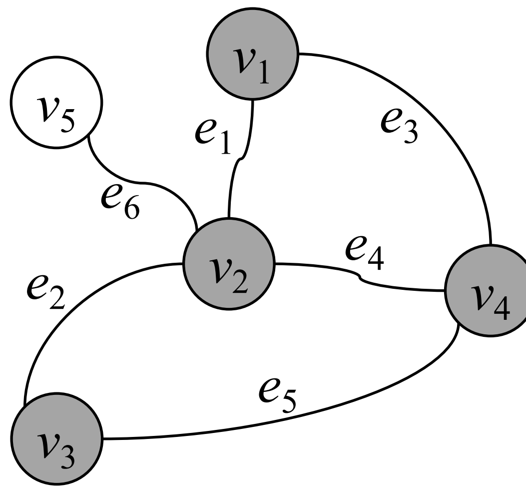
**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow time$;<br>
$u$.low $\leftarrow u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
 **if** $v.visited = false$ **then**<br>
  入栈（$S$, $(u, v)$）;<br>
  $v$.parent $\leftarrow u$;<br>
  $u$.children $\leftarrow u$.children + 1;<br>
  `DFSBlk`($G$, $v$);<br>
  $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
  **if** $(u.parent = null$ 且 $u.children \geq 2)$ 或 $(u.parent \neq null$ 且 $v.low \geq u.d)$ **then**<br>
   输出（以下边组成一个块）;<br>
   **do**<br>
    $(x, y) \leftarrow$ 出栈（$S$）;<br>
    输出（$(x, y)$）;<br>
   **while** $(x, y) \neq (u, v)$;<br>
 **else if** $v \neq u.parent$ **then**<br>
  入栈（$S$, $(u, v)$）;<br>
  $u$.low $\leftarrow$ min{$u$.low, $v$.d};



# 块

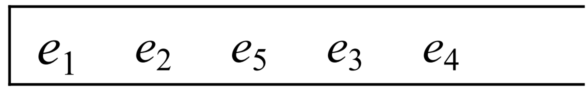
## ■ DFSBlk算法

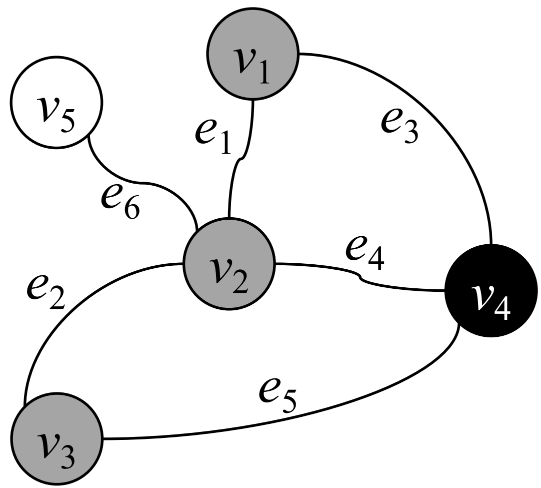
**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow$ $time$;<br>
$u$.low $\leftarrow$ $u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
  **if** $v.visited = false$ **then**<br>
    入栈（$S$, $(u, v)$）;<br>
    $v$.parent $\leftarrow$ $u$;<br>
    $u$.children $\leftarrow$ $u$.children + 1;<br>
    DFSBlk($G$, $v$);<br>
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
    **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**<br>
      输出（以下边组成一个块）;<br>
      **do**<br>
        $(x, y) \leftarrow$ 出栈（$S$）;<br>
        输出（$(x, y)$）;<br>
      **while** $(x, y) \neq (u, v)$;<br>
  **else if** $v \neq u.parent$ **then**<br>
    入栈（$S$, $(u, v)$）;<br>
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};



# 块

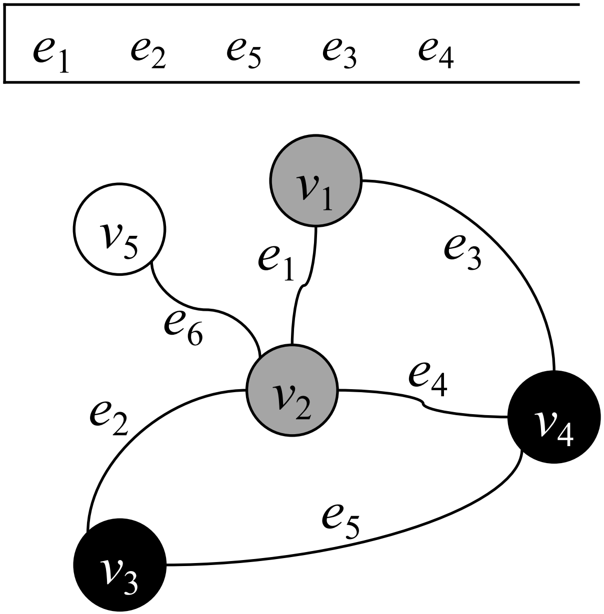
## ■ DFSBlk算法

**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

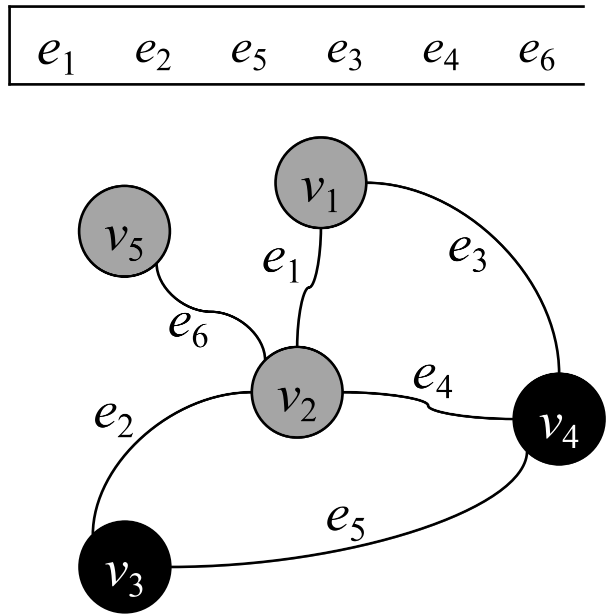
```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

## ■ DFSBlk算法

```
算法 7: DFSBlk
  输入: 非平凡连通图 G = ⟨V, E⟩，顶点 u
  初值: time 初值为 0；V 中所有顶点的 visited 初值为 false，
        parent 初值为 null，children 初值为 0；S 初值为空栈
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

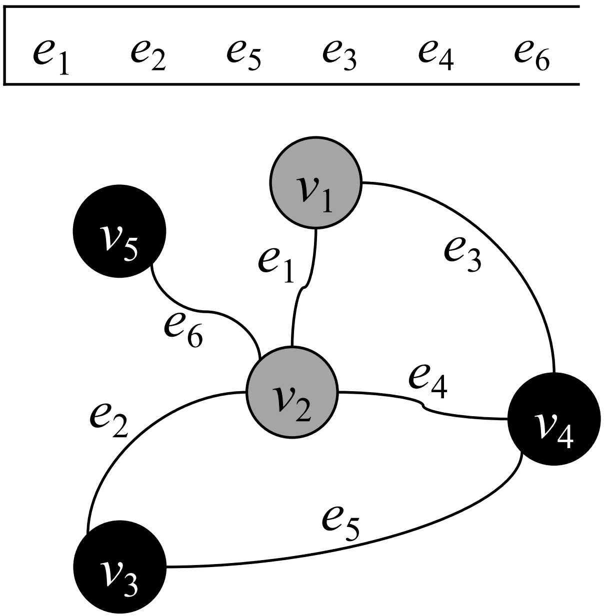
## DFSBlk算法

**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

- DFSBlk算法

**算法 7:** DFSBlk

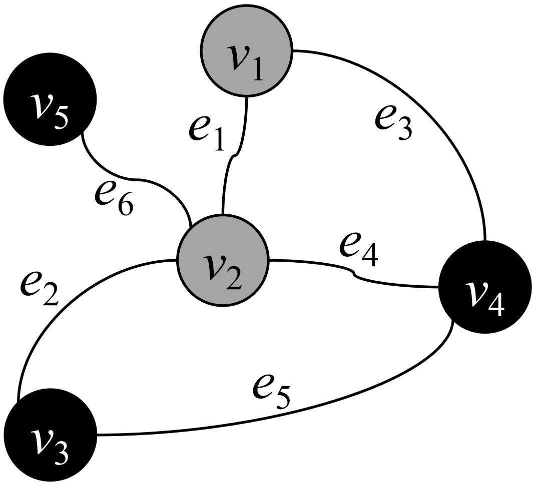
**输入:** 非平凡连通图 $G = \langle V, E \rangle$，顶点 $u$

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```

以下边组成一个块：$e_6$

| | | | | |
|---|---|---|---|---|
| $e_1$ | $e_2$ | $e_5$ | $e_3$ | $e_4$ |



# 块

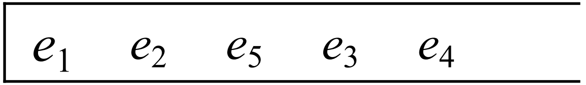
## DFSBlk算法

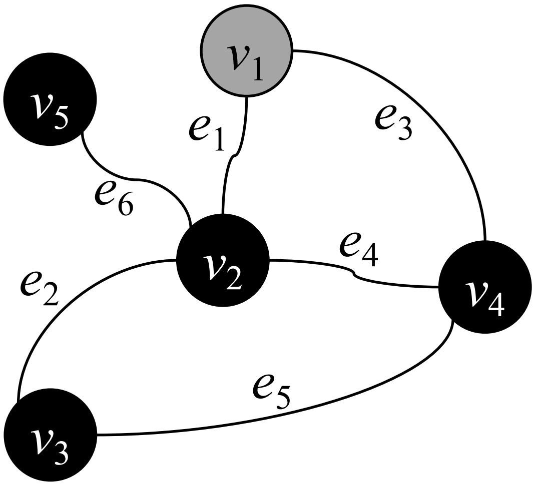
**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow time$;<br>
$u$.low $\leftarrow u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
    **if** $v.visited = false$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $v$.parent $\leftarrow u$;<br>
        $u$.children $\leftarrow u$.children + 1;<br>
        `DFSBlk`$(G, v)$;<br>
        $u$.low $\leftarrow$ min$\{u$.low, $v$.low$\}$;<br>
        **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**<br>
            输出（以下边组成一个块）;<br>
            **do**<br>
                $(x, y) \leftarrow$ 出栈（$S$）;<br>
                输出（$(x, y)$）;<br>
            **while** $(x, y) \neq (u, v)$;<br>
    **else if** $v \neq u.parent$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $u$.low $\leftarrow$ min$\{u$.low, $v$.d$\}$;



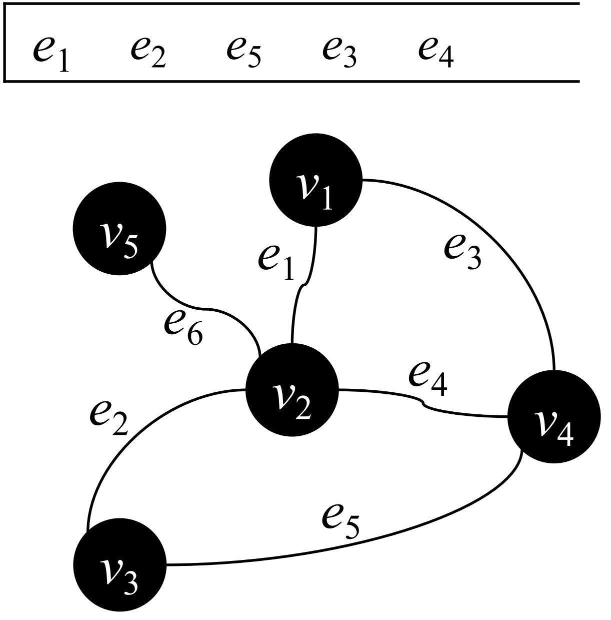



# 块

■ DFSBlk算法

**算法 7:** DFSBlk

```
输入: 非平凡连通图 G = ⟨V, E⟩, 顶点 u
初值: time 初值为 0; V 中所有顶点的 visited 初值为 false,
      parent 初值为 null, children 初值为 0; S 初值为空栈
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



# 块

## ■ DFSBlk算法

**算法 7:** DFSBlk

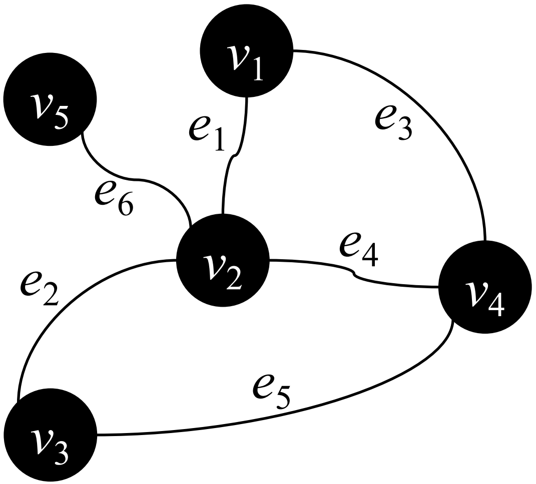
**输入:** 非平凡连通图 $G = \langle V, E \rangle$，顶点 $u$

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;
$u$.d $\leftarrow time$;
$u$.low $\leftarrow u$.d;
$u$.visited $\leftarrow$ true;
**foreach** $(u, v) \in E$ **do**
  **if** $v.visited = false$ **then**
    入栈（$S$, $(u, v)$）;
    $v$.parent $\leftarrow u$;
    $u$.children $\leftarrow u$.children + 1;
    `DFSBlk`($G$, $v$);
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};
    **if** *($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$)* **then**
      输出（以下边组成一个块）;
      **do**
        $(x, y) \leftarrow$ 出栈（$S$）;
        输出（$(x, y)$）;
      **while** $(x, y) \neq (u, v)$;
  **else if** $v \neq u.parent$ **then**
    入栈（$S$, $(u, v)$）;
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};

以下边组成一个块：$e_4$

| $e_1$ | $e_2$ | $e_5$ | $e_3$ |
|---|---|---|---|



# 块

## ■ DFSBlk算法

**算法 7:** DFSBlk

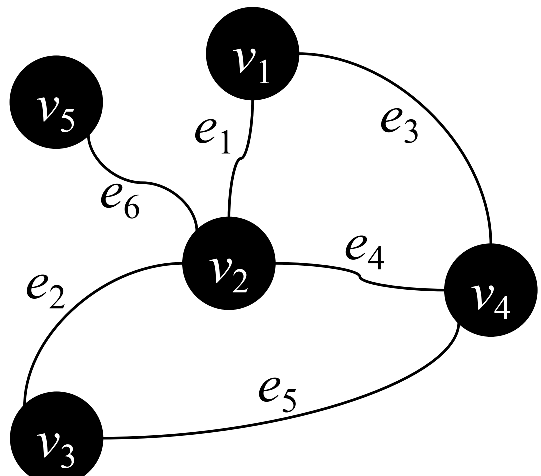
**输入:** 非平凡连通图 $G = \langle V, E\rangle$，顶点 $u$

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```

以下边组成一个块：$e_4$ $e_3$

$e_1$ $e_2$ $e_5$



# 块

- DFSBlk算法

**算法 7:** DFSBlk

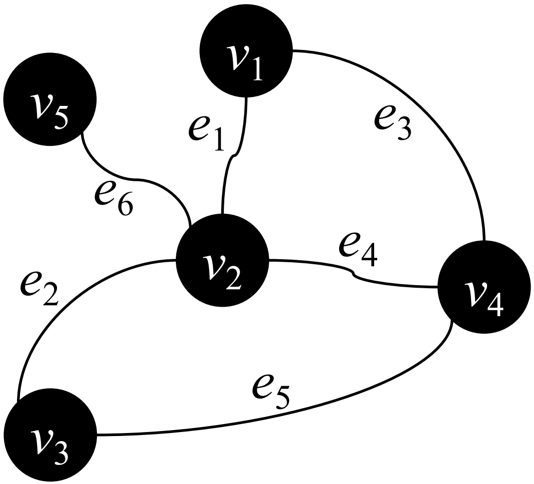
**输入:** 非平凡连通图 $G = \langle V, E \rangle$，顶点 $u$

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;
$u$.d $\leftarrow$ $time$;
$u$.low $\leftarrow$ $u$.d;
$u$.visited $\leftarrow$ true;
**foreach** $(u, v) \in E$ **do**
    **if** $v.visited = false$ **then**
        入栈（$S$, $(u, v)$）;
        $v$.parent $\leftarrow$ $u$;
        $u$.children $\leftarrow$ $u$.children + 1;
        DFSBlk($G$, $v$);
        $u$.low $\leftarrow$ min{$u$.low, $v$.low};
        **if** $(u.parent = null$ 且 $u.children \geq 2)$ 或 $(u.parent \neq null$ 且 $v.low \geq u.d)$ **then**
            输出（以下边组成一个块）;
            **do**
                $(x, y) \leftarrow$ 出栈（$S$）;
                输出（$(x, y)$）;
            **while** $(x, y) \neq (u, v)$;
    **else if** $v \neq u.parent$ **then**
        入栈（$S$, $(u, v)$）;
        $u$.low $\leftarrow$ min{$u$.low, $v$.d};

以下边组成一个块：$e_4$ $e_3$ $e_5$

| | |
|---|---|
| $e_1$ | $e_2$ |



# 块

## ■ DFSBlk算法

**算法 7:** DFSBlk

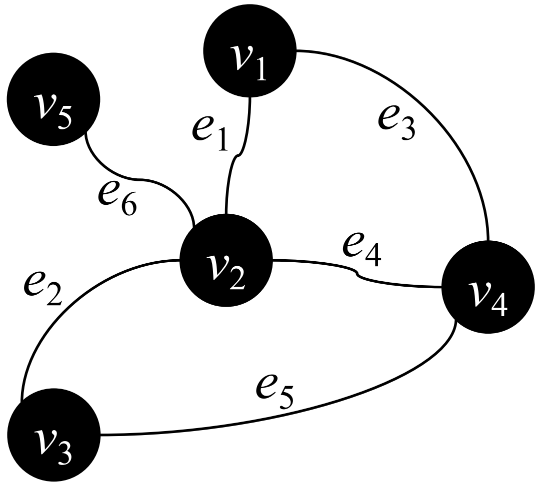
**输入:** 非平凡连通图 $G = \langle V, E\rangle$，顶点 $u$

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
          且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```

以下边组成一个块：$e_4\ e_3\ e_5\ e_2$

$e_1$



# 块

- **DFSBlk算法**

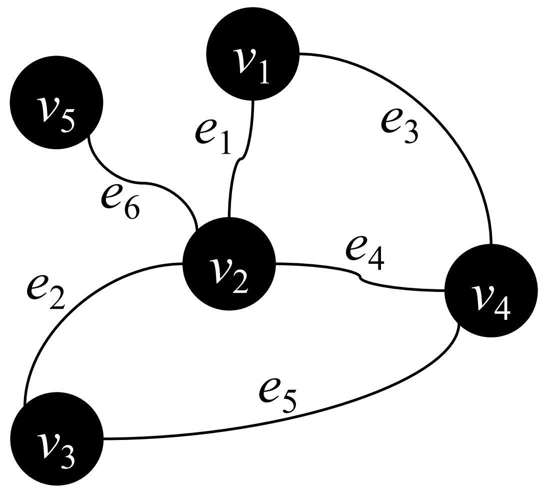
**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;<br>
$u$.d $\leftarrow$ $time$;<br>
$u$.low $\leftarrow$ $u$.d;<br>
$u$.visited $\leftarrow$ true;<br>
**foreach** $(u, v) \in E$ **do**<br>
    **if** $v.visited = false$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $v$.parent $\leftarrow$ $u$;<br>
        $u$.children $\leftarrow$ $u$.children + 1;<br>
        `DFSBlk`($G$, $v$);<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.low};<br>
        **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**<br>
            输出（以下边组成一个块）;<br>
            **do**<br>
                $(x, y) \leftarrow$ 出栈（$S$）;<br>
                输出（$(x, y)$）;<br>
            **while** $(x, y) \neq (u, v)$;<br>
    **else if** $v \neq u.parent$ **then**<br>
        入栈（$S$, $(u, v)$）;<br>
        $u$.low $\leftarrow$ min{$u$.low, $v$.d};

以下边组成一个块：$e_4\, e_3\, e_5\, e_2\, e_1$



# 块

- 从$u$发现$(u, v)$，到完成对$v$的递归调用并判定$u$为割点

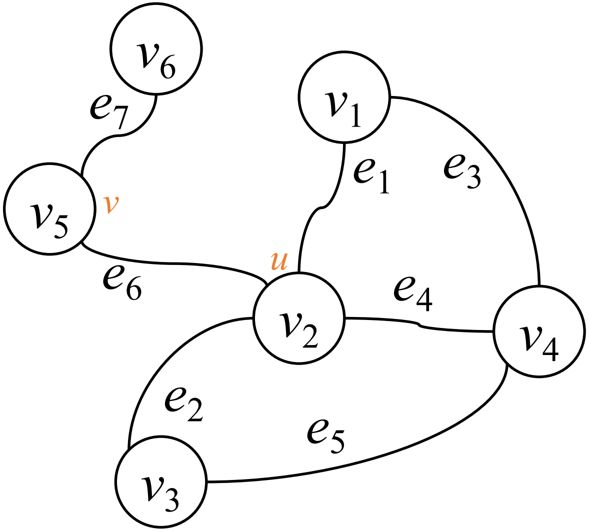
**算法 7:** DFSBlk

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

**初值:** $time$ 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;
$u$.d $\leftarrow$ $time$;
$u$.low $\leftarrow$ $u$.d;
$u$.visited $\leftarrow$ true;
**foreach** $(u, v) \in E$ **do**
  **if** $v.visited = false$ **then**
    入栈（$S$, $(u, v)$）;
    $v$.parent $\leftarrow$ $u$;
    $u$.children $\leftarrow$ $u$.children + 1;
    DFSBlk($G$, $v$);
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};
    **if** $(u.parent = null$ 且 $u.children \geq 2)$ 或 $(u.parent \neq null$ 且 $v.low \geq u.d)$ **then**
      输出（以下边组成一个块）;
      **do**
        $(x, y) \leftarrow$ 出栈（$S$）;
        输出（$(x, y)$）;
      **while** $(x, y) \neq (u, v)$;
  **else if** $v \neq u.parent$ **then**
    入栈（$S$, $(u, v)$）;
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};

这个时间段内，发现了哪些边？



# 块

- 从$u$发现$(u, v)$，到完成对$v$的递归调用并判定$u$为割点

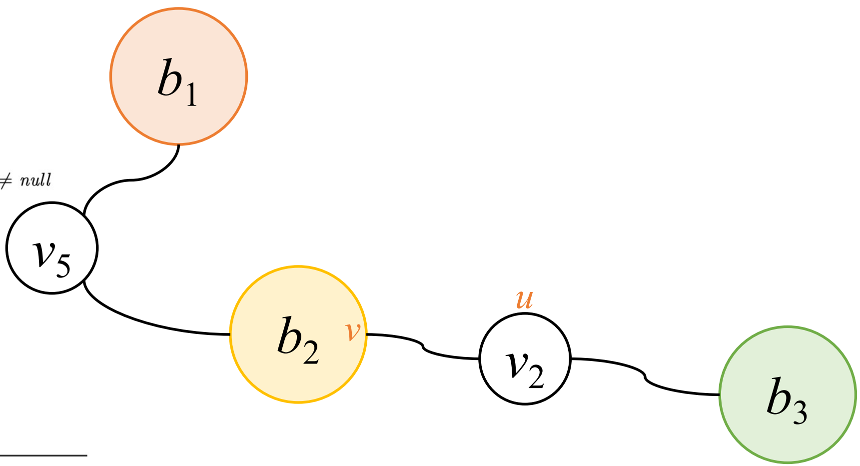
**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

$time \leftarrow time + 1$;
$u$.d $\leftarrow$ $time$;
$u$.low $\leftarrow$ $u$.d;
$u$.visited $\leftarrow$ true;
**foreach** $(u, v) \in E$ **do**
  **if** $v.visited = false$ **then**
    入栈（$S$, $(u, v)$）;
    $v$.parent $\leftarrow$ $u$;
    $u$.children $\leftarrow$ $u$.children + 1;
    DFSBlk($G$, $v$);
    $u$.low $\leftarrow$ min{$u$.low, $v$.low};
    **if** ($u.parent = null$ 且 $u.children \geq 2$) 或 ($u.parent \neq null$ 且 $v.low \geq u.d$) **then**
      输出（以下边组成一个块）;
      **do**
        $(x, y) \leftarrow$ 出栈（$S$）;
        输出（$(x, y)$）;
      **while** $(x, y) \neq (u, v)$;
  **else if** $v \neq u.parent$ **then**
    入栈（$S$, $(u, v)$）;
    $u$.low $\leftarrow$ min{$u$.low, $v$.d};

这个时间段内，发现了哪些边？

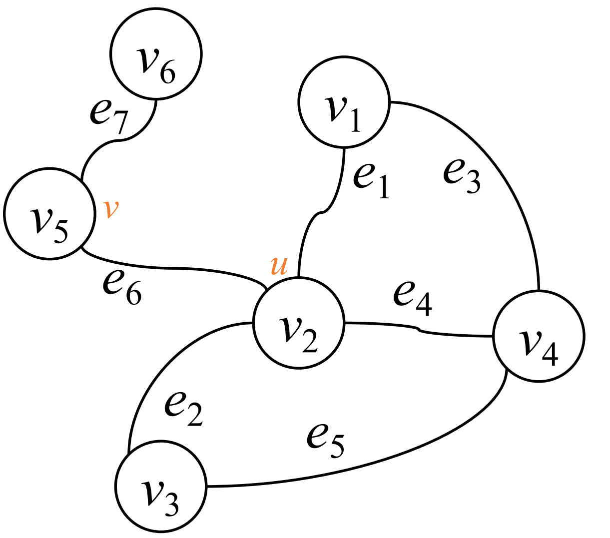


# 块

- 从$u$发现$(u, v)$，到完成对$v$的递归调用并判定$u$为割点

```
算法 7: DFSBlk
  输入: 非平凡连通图 G = ⟨V, E⟩，顶点 u
  初值: time 初值为 0；V 中所有顶点的 visited 初值为 false，
        parent 初值为 null，children 初值为 0；S 初值为空栈
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
          且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```

这个时间段结束时，输出哪些边？



# 块

- 从$u$发现$(u, v)$，到完成对$v$的递归调用并判定$u$为割点

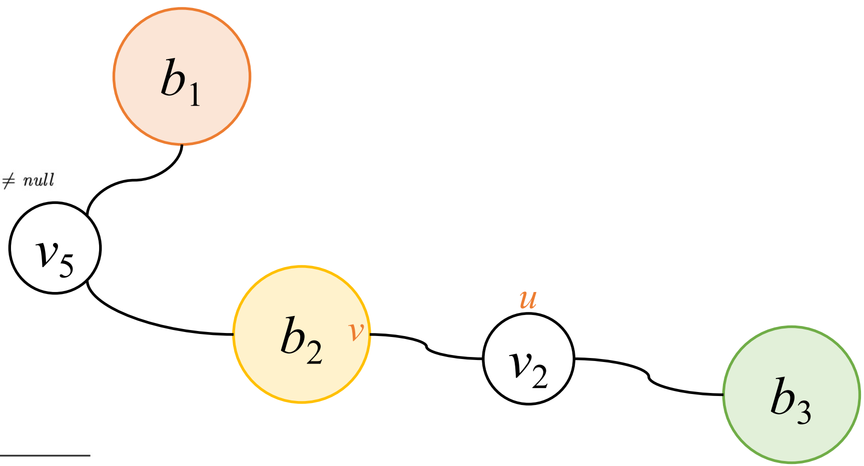
**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```

这个时间段结束时，输出哪些边？



# 块

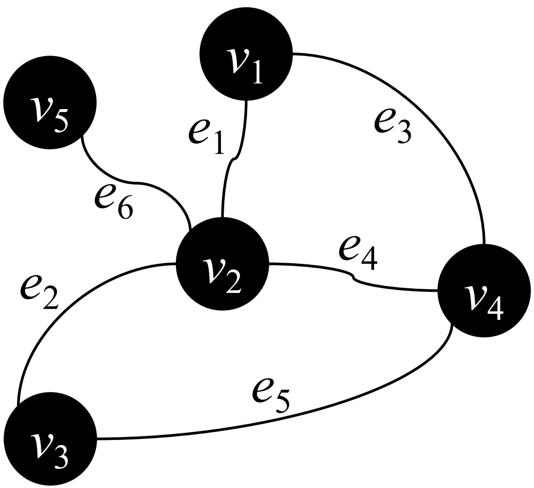
- 算法结束时，何时栈空，何时栈非空？

**算法 7:** DFSBlk

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

**初值:** *time* 初值为 0；$V$ 中所有顶点的 visited 初值为 false，parent 初值为 null，children 初值为 0；$S$ 初值为空栈

```
time ← time + 1;
u.d ← time;
u.low ← u.d;
u.visited ← true;
foreach (u, v) ∈ E do
    if v.visited = false then
        入栈 (S, (u, v));
        v.parent ← u;
        u.children ← u.children + 1;
        DFSBlk(G, v);
        u.low ← min{u.low, v.low};
        if (u.parent = null 且 u.children ≥ 2) 或 (u.parent ≠ null
         且 v.low ≥ u.d) then
            输出 (以下边组成一个块);
            do
                (x, y) ← 出栈 (S);
                输出 ((x, y));
            while (x, y) ≠ (u, v);
    else if v ≠ u.parent then
        入栈 (S, (u, v));
        u.low ← min{u.low, v.d};
```



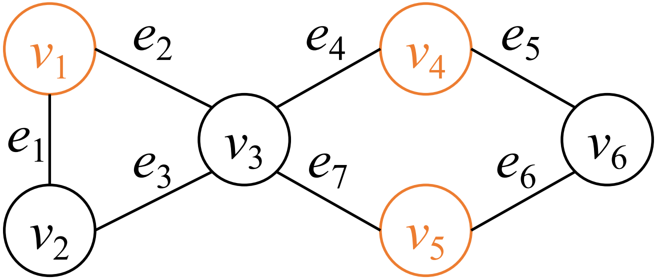
# 本次课的主要内容

4.1 块

4.2 割集和连通度

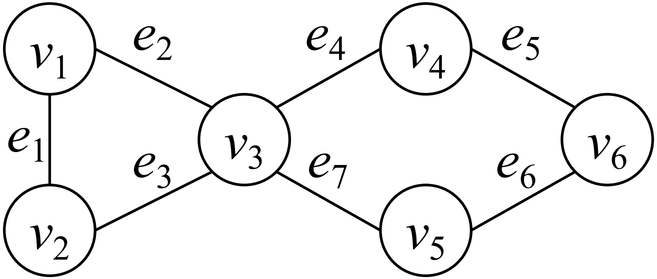
# 割集和连通度

- **点割集**（分离集）
  - 对于图$G = <V, E>$和顶点子集$S \subseteq V$,
    $G - S$的连通分支数量大于$G$的连通分支数量



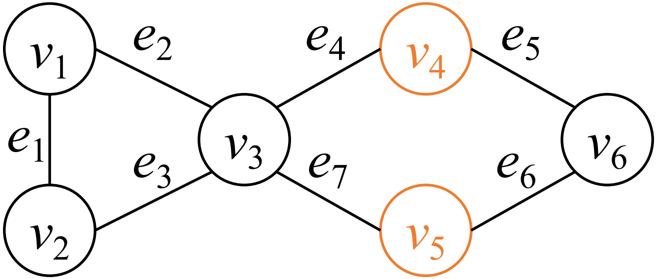
# 割集和连通度

- 点割集（分离集）
  - 对于图$G = <V, E>$和顶点子集$S \subseteq V$，
    $G - S$的连通分支数量大于$G$的连通分支数量
- 点割集和割点的关系？



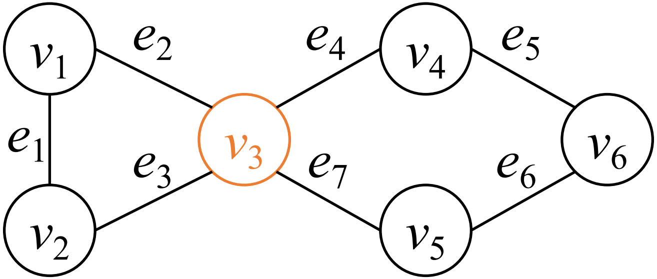
# 割集和连通度

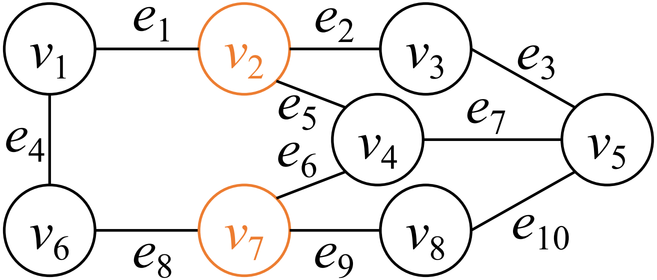
- 点割集（分离集）
  - 对于图$G = <V, E>$和顶点子集$S \subseteq V$,
    $G - S$的连通分支数量大于$G$的连通分支数量
- 点割集和割点的关系？
- **极小点割集**



# 割集和连通度

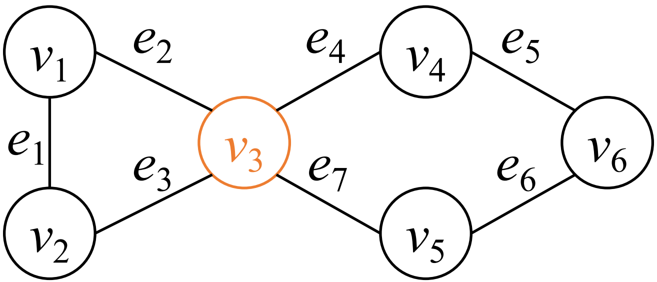
- 点割集（分离集）
  - 对于图$G = <V, E>$和顶点子集$S \subseteq V$,
    $G - S$的连通分支数量大于$G$的连通分支数量
- 点割集和割点的关系？
- 极小点割集
- **最小点割集**

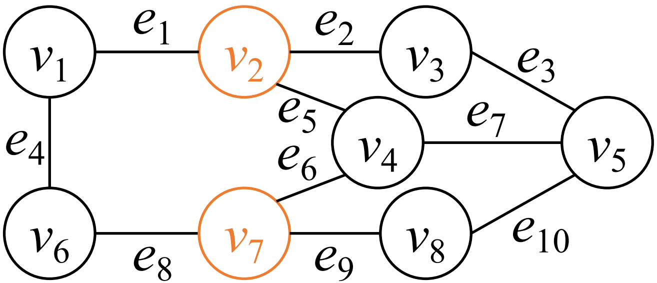


# 割集和连通度

- **点连通度**（连通度）
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的顶点数量，记作$\kappa(G)$

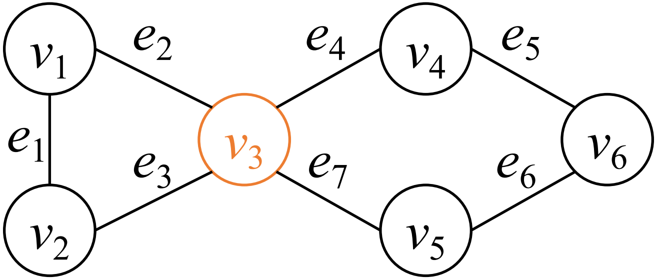


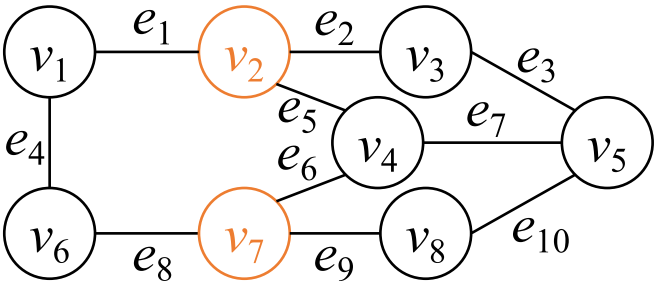



# 割集和连通度

- 点连通度（连通度）
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的顶点数量，记作$\kappa(G)$

- 若连通图$G$有点割集，则$G$的最小点割集和$\kappa(G)$有什么关系？

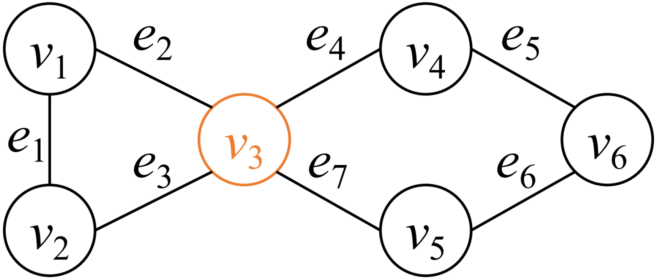


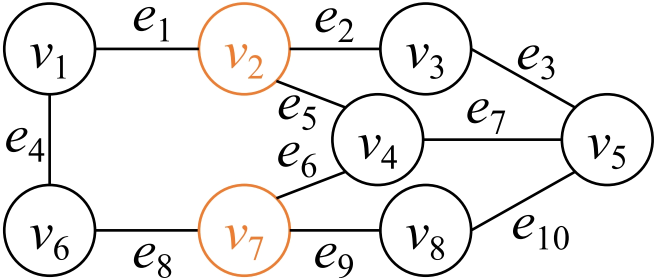



# 割集和连通度

- 点连通度（连通度）
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的顶点数量，记作$\kappa(G)$

- 若连通图$G$有点割集，则$G$的最小点割集和$\kappa(G)$有什么关系？
- 完全图$K_n$的连通度是多少？

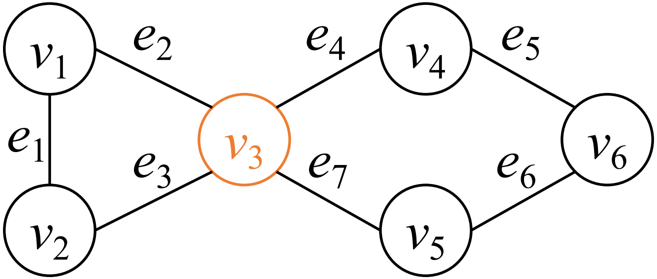


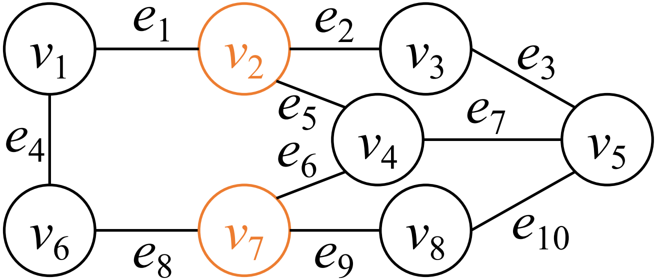



# 割集和连通度

- 点连通度（连通度）
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的顶点数量，记作$\kappa(G)$

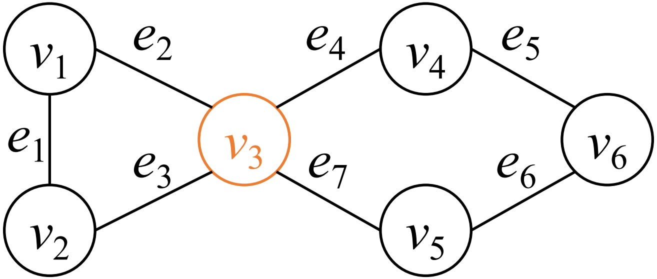
- 若连通图$G$有点割集，则$G$的最小点割集和$\kappa(G)$有什么关系？
- 完全图$K_n$的连通度是多少？
- 不连通图的连通度是多少？

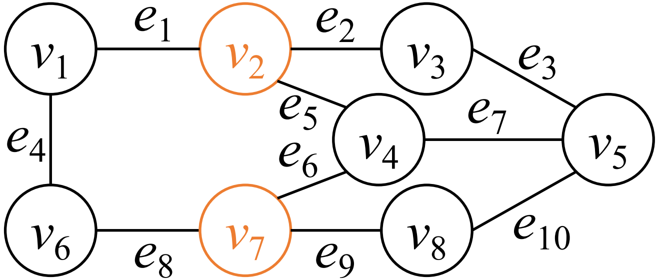


# 割集和连通度

- ***k*-点连通图**（$k$-连通图）
  - $k \leq \kappa(G)$

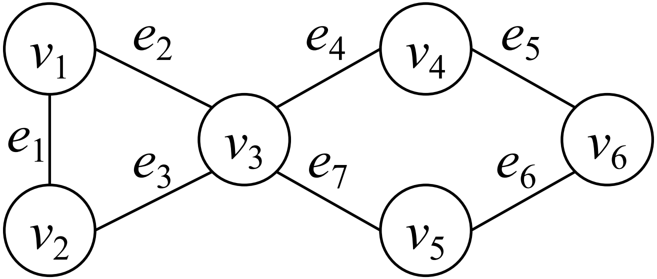


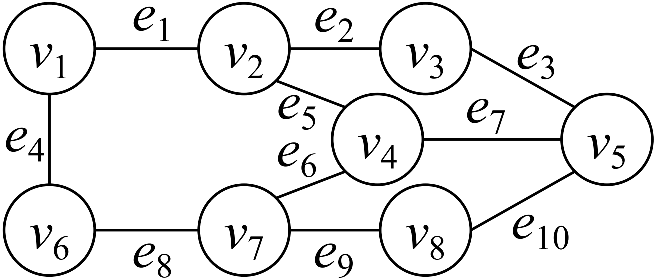



# 割集和连通度

- $k$-点连通图（$k$-连通图）
  - $k \le \kappa(G)$

- 1-连通图一定是连通图吗？反之呢？

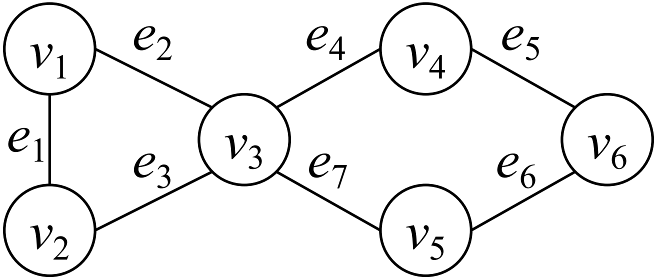


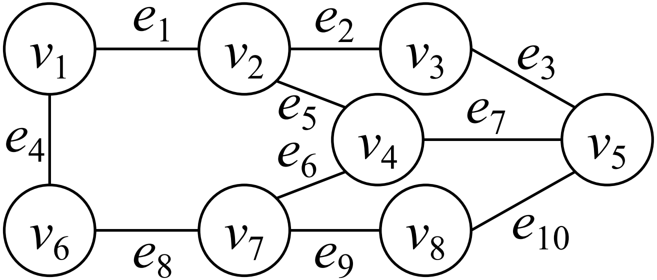



# 割集和连通度

- $k$-点连通图（$k$-连通图）
  - $k \le \kappa(G)$

- 1-连通图一定是连通图吗？反之呢？
- 2-连通图一定是块吗？反之呢？

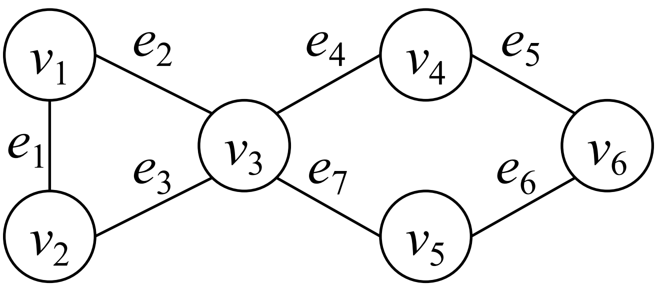


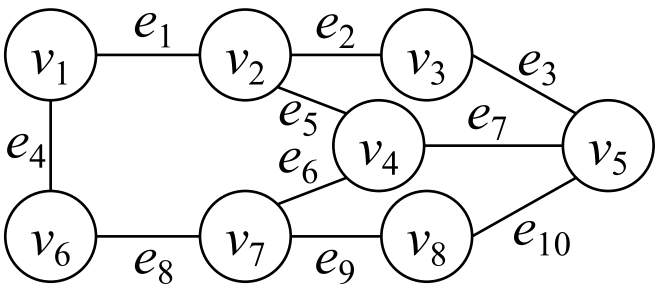



# 割集和连通度

- $k$-点连通图（$k$-连通图）
  - $k \le \kappa(G)$

- 1-连通图一定是连通图吗？反之呢？
- 2-连通图一定是块吗？反之呢？
- 从$k$-连通图中删除任意$k$ - 1个顶点，剩余图一定连通吗？

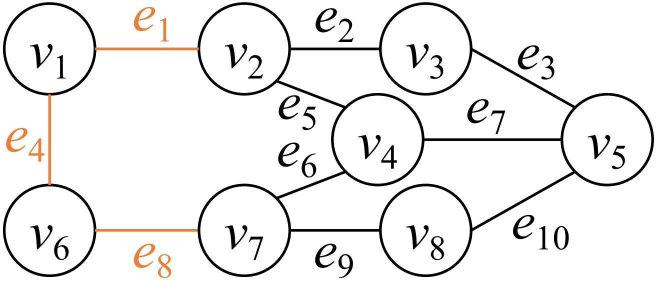


# 割集和连通度

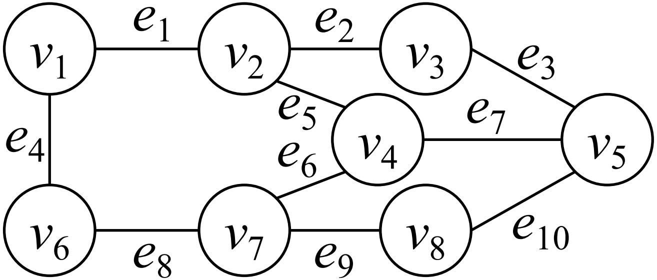
## 边割集

- 对于图$G = <V, E>$和边子集$S \subseteq E$,
  $G - S$的连通分支数量大于$G$的连通分支数量



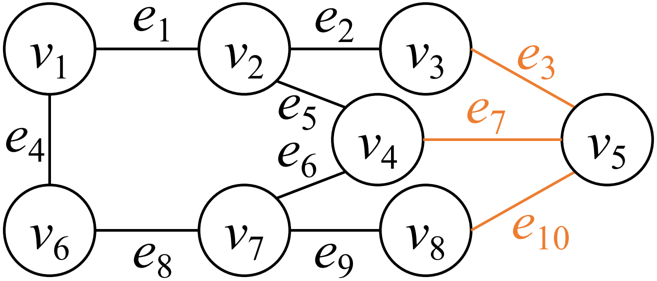
# 割集和连通度

- 边割集
  - 对于图$G = <V, E>$和边子集$S \subseteq E$，$G - S$的连通分支数量大于$G$的连通分支数量
- 边割集和割边的关系？



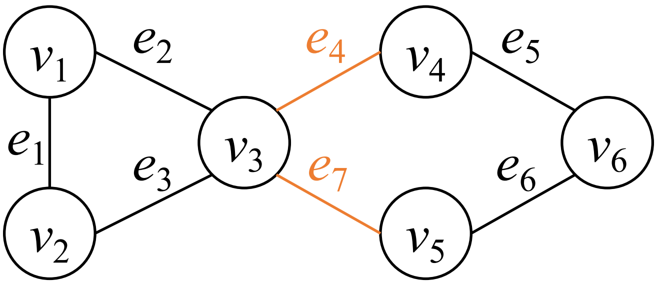
# 割集和连通度

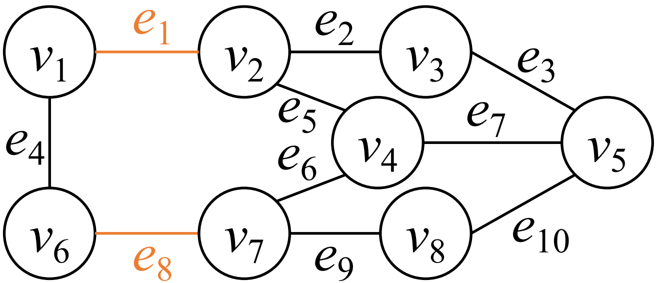
- 边割集
  - 对于图$G = <V, E>$和边子集$S \subseteq E$，
    $G - S$的连通分支数量大于$G$的连通分支数量
- 边割集和割边的关系？
- **极小边割集**



# 割集和连通度

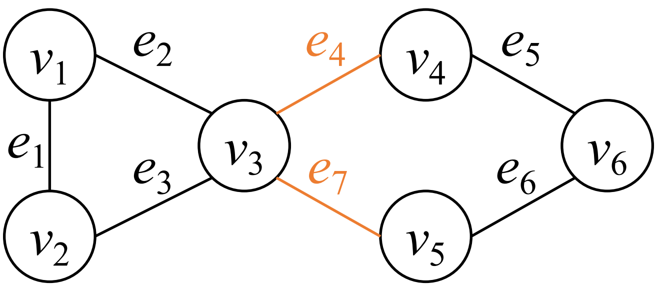
- 边割集
  - 对于图$G = <V, E>$和边子集$S \subseteq E$,
    $G$ - $S$的连通分支数量大于$G$的连通分支数量
- 边割集和割边的关系？
- 极小边割集
- **最小边割集**

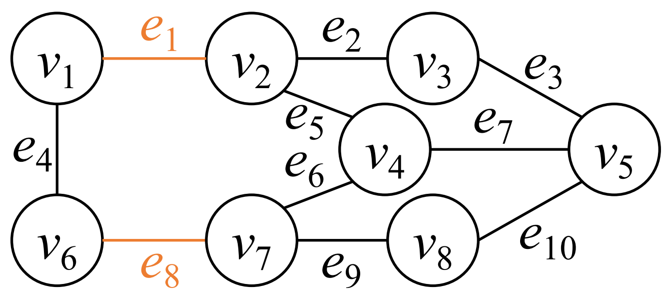


# 割集和连通度

- **边连通度**
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作κ’($G$)

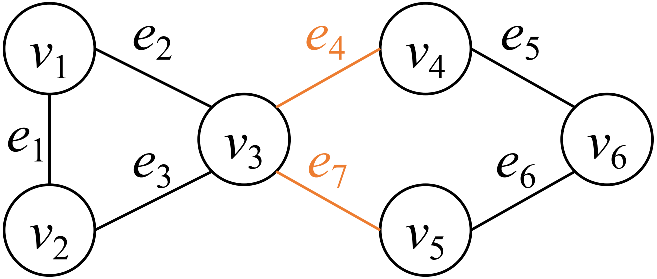


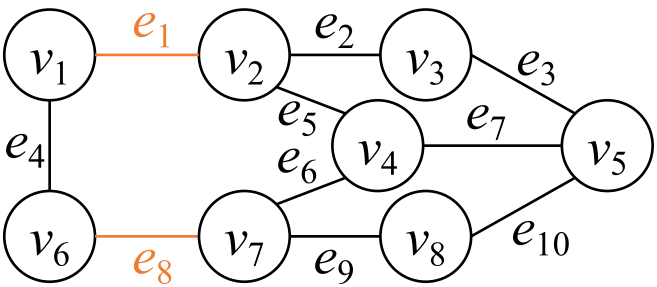



# 割集和连通度

- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作κ’($G$)

- 若连通图$G$有边割集，则$G$的最小边割集和κ’($G$)有什么关系？

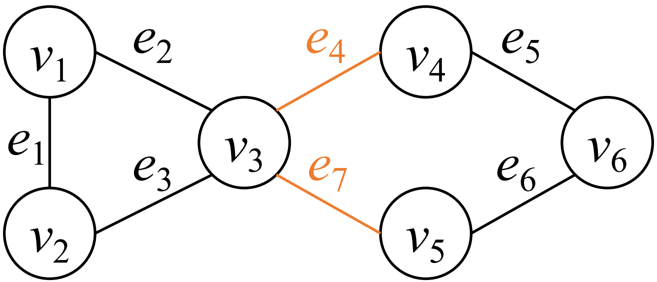


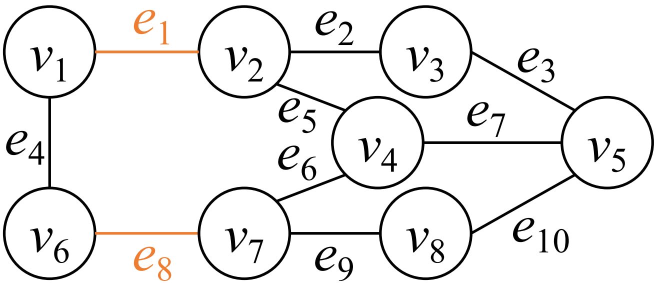



# 割集和连通度

- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作κ’($G$)

- 若连通图$G$有边割集，则$G$的最小边割集和κ’($G$)有什么关系？
- 完全图$K_n$的边连通度是多少？

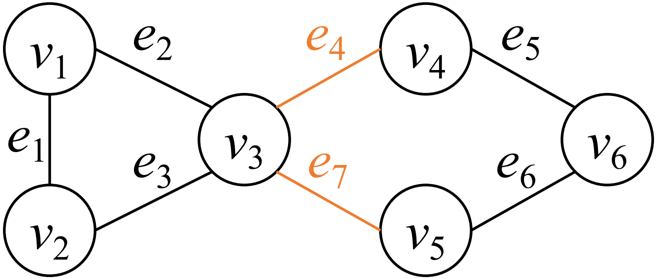


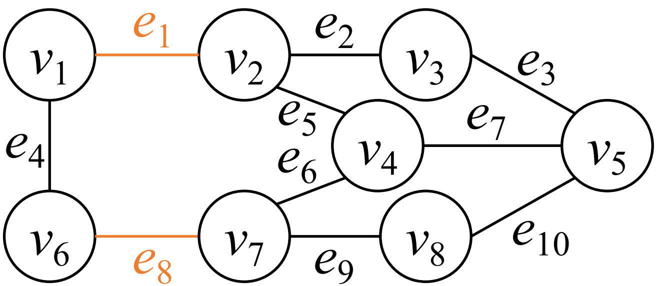



# 割集和连通度

- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作κ'($G$)

- 若连通图$G$有边割集，则$G$的最小边割集和κ'($G$)有什么关系？
- 完全图$K_n$的边连通度是多少？
  - 为什么$\kappa'(G) \le n-1$？

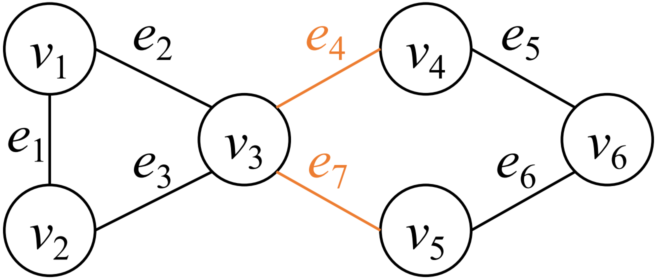


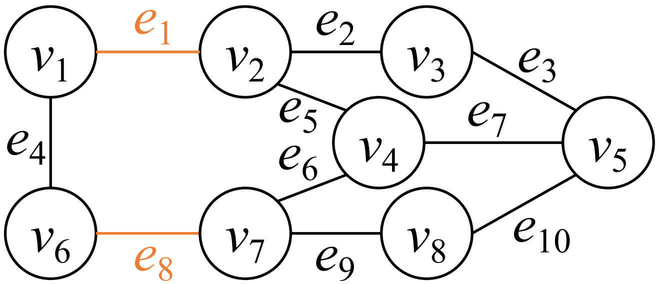



# 割集和连通度

- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作$\kappa'(G)$

- 若连通图$G$有边割集，则$G$的最小边割集和$\kappa'(G)$有什么关系？
- 完全图$K_n$的边连通度是多少？
  - 为什么$\kappa'(G) \le n-1$？
  - 为什么$\kappa'(G) \ge n-1$？

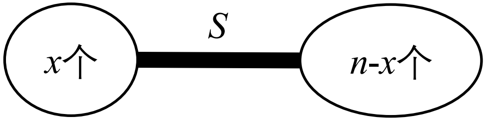


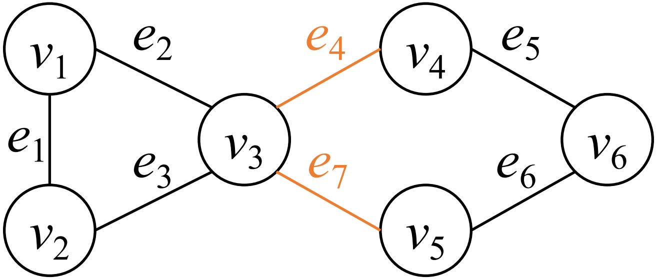
# 割集和连通度

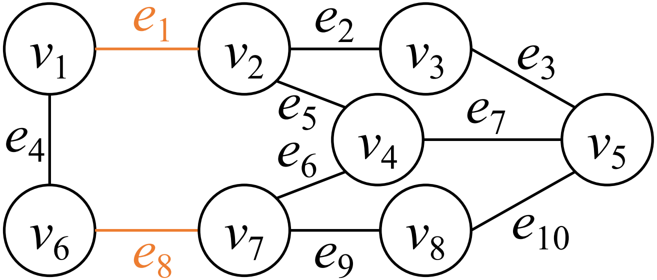
- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作$\kappa'(G)$

- 若连通图$G$有边割集，则$G$的最小边割集和$\kappa'(G)$有什么关系？
- 完全图$K_n$的边连通度是多少？
  - 为什么$\kappa'(G) \le n-1$？
  - 为什么$\kappa'(G) \ge n-1$？




$|S| = x(n-x) < n-1 \rightarrow (x-1)(n-x-1) < 0$，矛盾

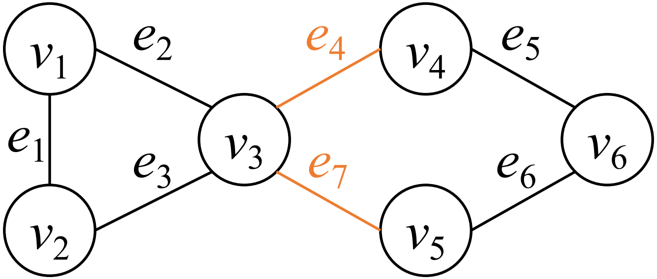


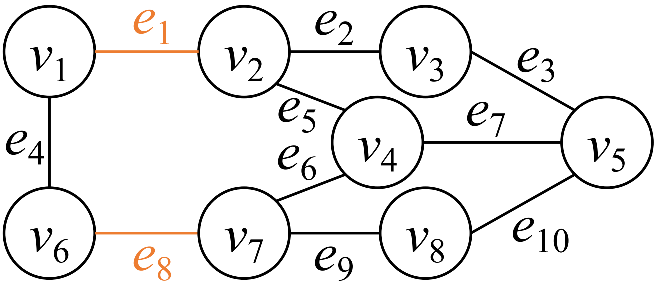



# 割集和连通度

- 边连通度
  - 为使图$G$不连通或成为平凡图，至少需要从$G$中删除的边的数量，记作κ’($G$)

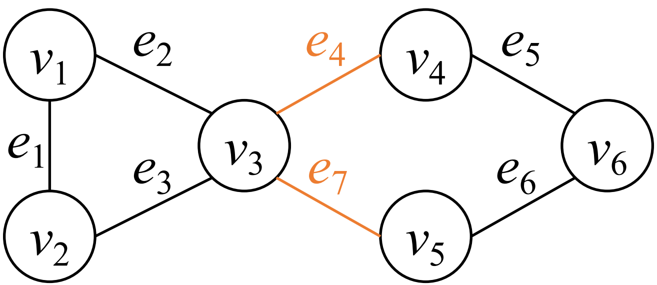
- 若连通图$G$有边割集，则$G$的最小边割集和κ’($G$)有什么关系？
- 完全图$K_n$的边连通度是多少？
- 不连通图的边连通度是多少？

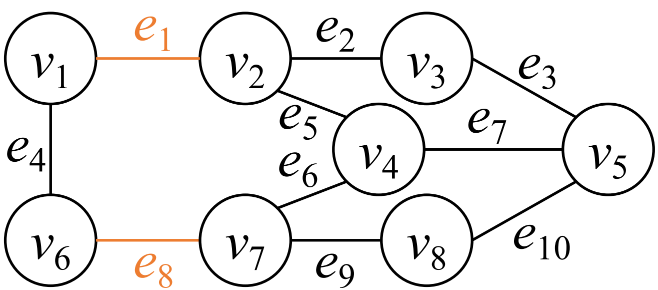


# 割集和连通度

- **$k$–边连通图**
  - $k \leq \kappa'(G)$

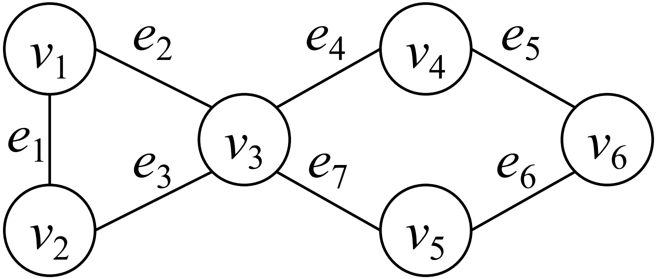


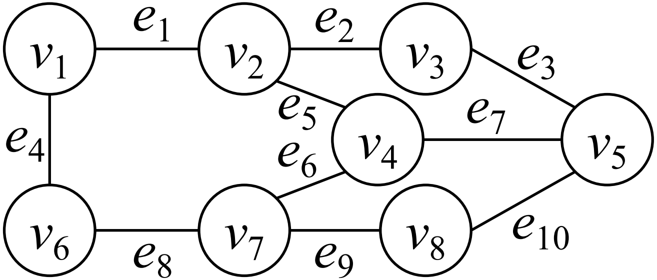



# 割集和连通度

- $k$-边连通图
  - $k \le \kappa'(G)$

- 1-边连通图一定是连通图吗？反之呢？

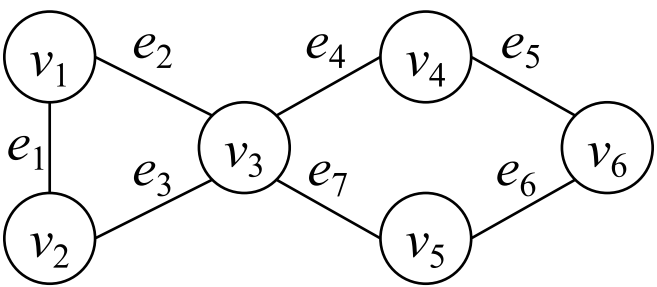


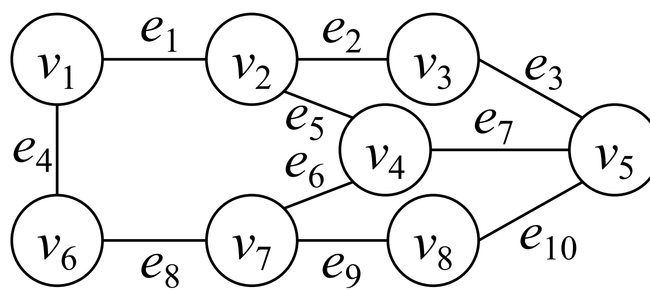



# 割集和连通度

- $k$-边连通图
  - $k \le \kappa'(G)$

- 1-边连通图一定是连通图吗？反之呢？
- 2-边连通图一定是块吗？反之呢？

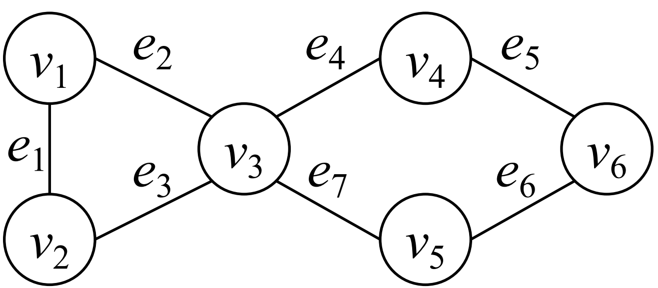


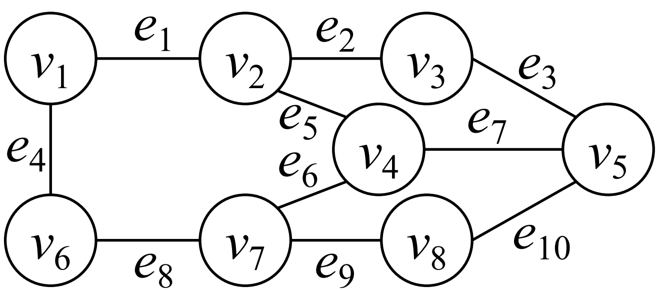



# 割集和连通度

- $k$-边连通图
  - $k \le \kappa'(G)$

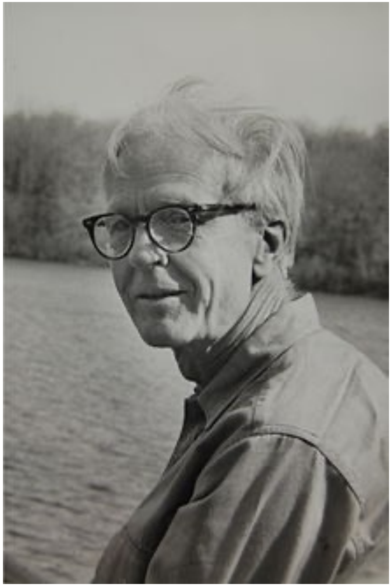
- 1-边连通图一定是连通图吗？反之呢？
- 2-边连通图一定是块吗？反之呢？
- 从$k$-边连通图中删除任意$k$ - 1条边，剩余图一定连通吗？



# 割集和连通度

- Hassler Whitney，1907-1989，出生于美国



奇点理论奠基人之一
拟阵理论奠基人之一
……

https://en.wikipedia.org/wiki/Hassler_Whitney

# 割集和连通度

- 对于任意一个图$G$：$\kappa(G) \le \kappa'(G) \le \delta(G)$

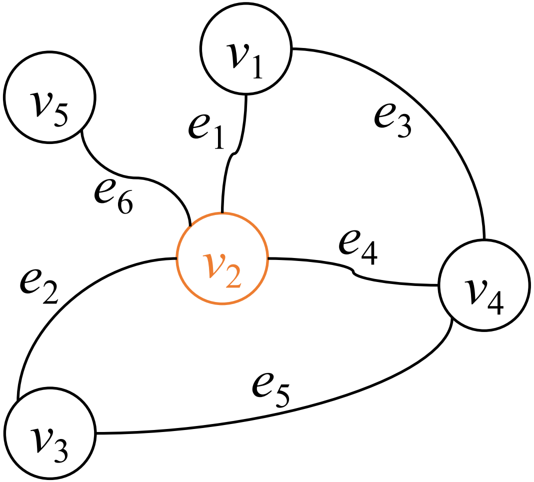
# 割集和连通度

- 决定图的点（边）连通度的本质要素是什么？
  向图中增加边，图的点连通度和边连通度一定提高吗？

# 割集和连通度

- 决定图的点（边）连通度的本质要素是什么？
  向图中增加边，图的点连通度和边连通度一定提高吗？

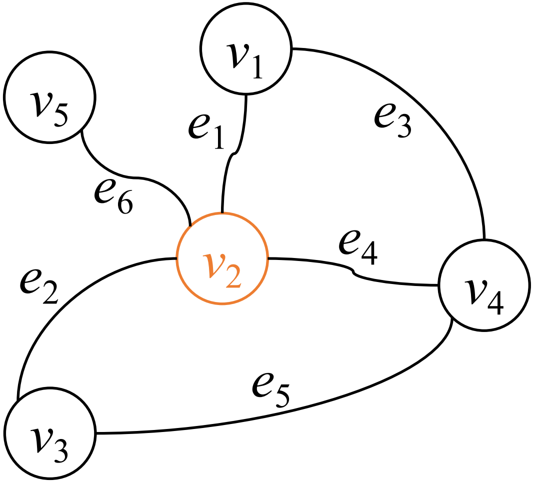
- 对于有割点的连通图，删除割点后不连通的本质原因是什么？



# 割集和连通度

- 决定图的点（边）连通度的本质要素是什么？
  向图中增加边，图的点连通度和边连通度一定提高吗？

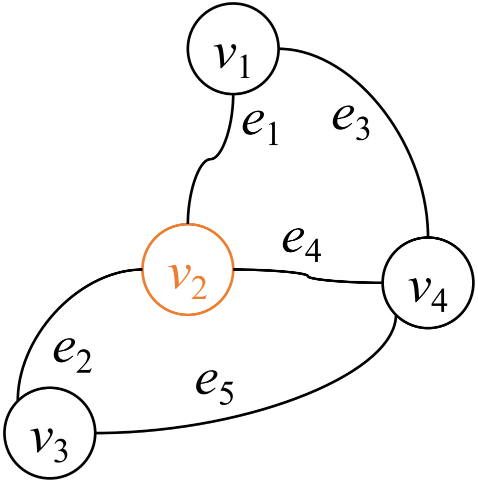
- 对于有割点的连通图，删除割点后不连通的本质原因是什么？
  - 割点是某两个顶点间的所有路的公共内顶点



# 割集和连通度

- 决定图的点（边）连通度的本质要素是什么？
  向图中增加边，图的点连通度和边连通度一定提高吗？

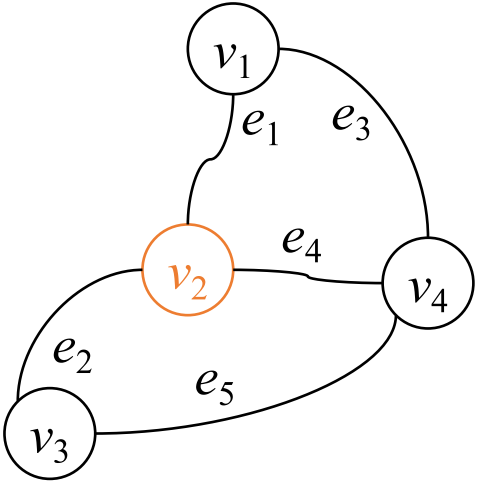
- 对于块，删除任意顶点后仍连通的本质原因是什么？



# 割集和连通度

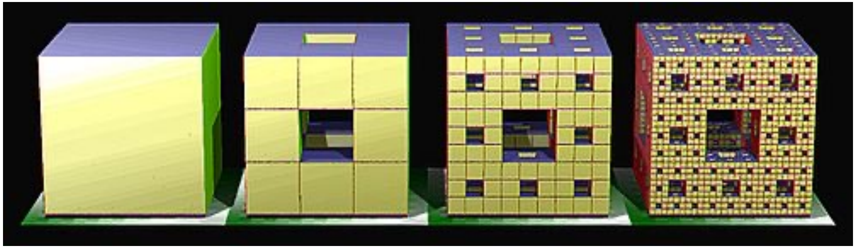
- 决定图的点（边）连通度的本质要素是什么？
  向图中增加边，图的点连通度和边连通度一定提高吗？

- 对于块，删除任意顶点后仍连通的本质原因是什么？
  - 任意两个顶点间存在至少2条无公共内顶点的路



# 割集和连通度

- Karl Menger， 1902-1985，出生于奥匈帝国



https://en.wikipedia.org/wiki/Karl_Menger
https://en.wikipedia.org/wiki/Menger_sponge

# 割集和连通度

- 门格尔定理：对于图$G = <V, E>$和两个不相邻的顶点$u, v \in V$，使$u$和$v$不连通至少需要从$G$中删除的顶点数量等于$G$中两两无公共内顶点的$u$-$v$路的最大数量。

- 推论：非平凡图$G$是$k$-连通图当且仅当$G$中任意两个顶点间存在至少$k$条两两无公共内顶点的路。

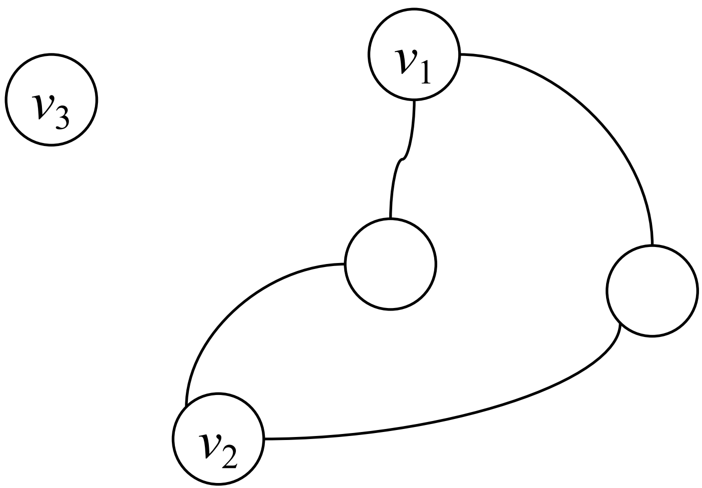
- 上述定理和推论面向的是点连通度，面向边连通度的结论是类似的。

# 割集和连通度

- 对于$k$-连通图$G = <V, E>$和任意$k$个顶点$v_1, \ldots, v_k \in V$（$k \geq 2$），$G$含圈经过$v_1, \ldots, v_k$。

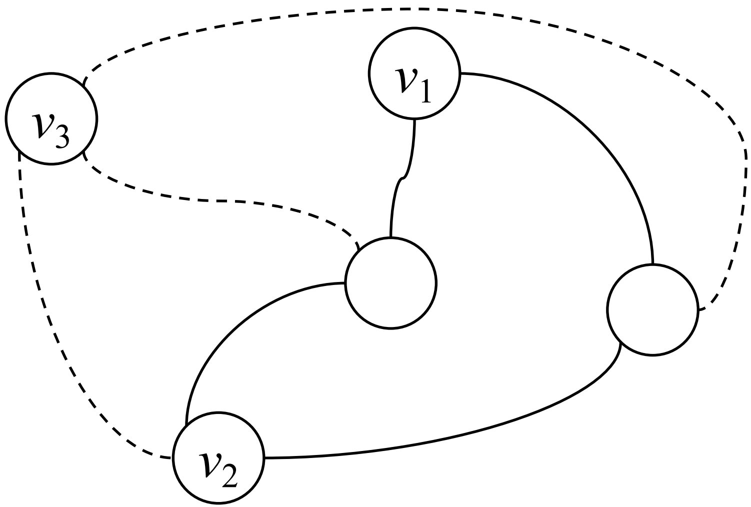
# 割集和连通度

- 对于$k$-连通图$G = <V, E>$和任意$k$个顶点$v_1, \ldots, v_k \in V$（$k \geq 2$），$G$含圈经过$v_1, \ldots, v_k$。



# 割集和连通度

- 对于$k$-连通图$G = <V, E>$和任意$k$个顶点$v_1, \ldots, v_k \in V$（$k \geq 2$），$G$含圈经过$v_1, \ldots, v_k$。



# 书面作业

- 练习4.1
- 练习4.3、4.4、4.5、4.6

## 实战应用

**练习 4.1.** 某国的一些城市间开通了直飞航线，任意两座城市间均可直飞或通过中转到达。起初，每家航空公司只运营一条航线。经过一段时间的运营，为提高运营效率，各航空公司都积极酝酿合并。为避免无序合并，民航总局要求：每家合并后的航空公司，其航线网络必须具有通达性和容错性，即：其航线网络覆盖的所有城市间必须有航路，且不得由于任何一座城市机场的临时关闭导致其航线网络覆盖的其它城市间的航路中断。经过一段时间的合并，在不违反上述要求的前提下，任何新的合并都不可能再发生。此时，只剩下 $k$ 家航空公司。能否通过每家航空公司的航线网络覆盖的城市数量 $n_1,\ldots,n_k$ 推算出城市的总数量？

## 实战应用

某国有至少 3 座城市，部分（非全部）城市间开通了直飞航线，任意两座城市间均可直飞或通过中转到达。整个航线网络的“城市容错系数”为 $p$，是指关闭至少 $p$ 座城市的机场才会导致其它某些城市间的航路中断。整个航线网络的“航线容错系数”为 $l$，是指关闭至少 $l$ 条直飞航线才会导致某些城市间的航路中断。

**练习 4.3.** 若每座城市恰开通 *2* 条直飞航线，请比较整个航线网络的城市容错系数和航线容错系数。

**练习 4.4.** 若每座城市恰开通 *3* 条直飞航线，请比较整个航线网络的城市容错系数和航线容错系数。

**练习 4.5.** 当前整个航线网络的城市容错系数为 $p$，若新建一座城市并和其它 $p$ 座城市开通直飞航线，则整个航线网络的城市容错系数如何变化？

**练习 4.6.** 有一条直飞航线因故临时关闭，尽管未导致任何城市间的航路中断，但有可能降低整个航线网络的城市容错系数和航线容错系数吗？若有可能，降低幅度是多少？