《算法设计与分析》

第二章 图与遍历算法

马丙鹏

2024年09月23日

第二章 图与遍历算法

- 2.1 图的基本概念和性质
- 2.2 图的遍历算法
- 2.3 双联通图与网络可靠性
- 2.4 对策树

2.3 双联通图与网络可靠性

■ 1. 问题描述

□通信网

- ➢图中结点表示通信站，边表示通信线路。

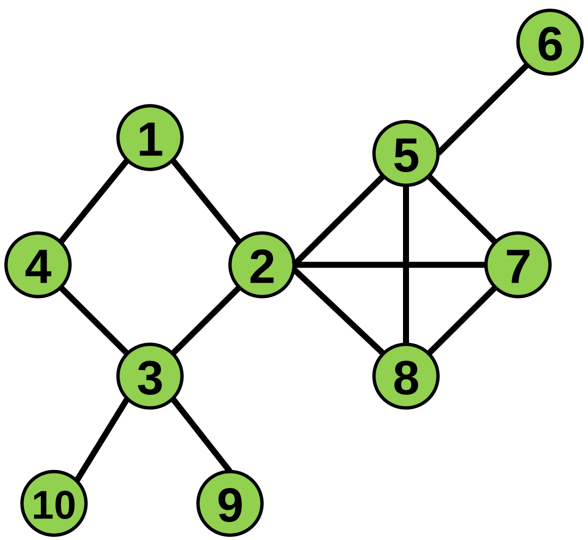
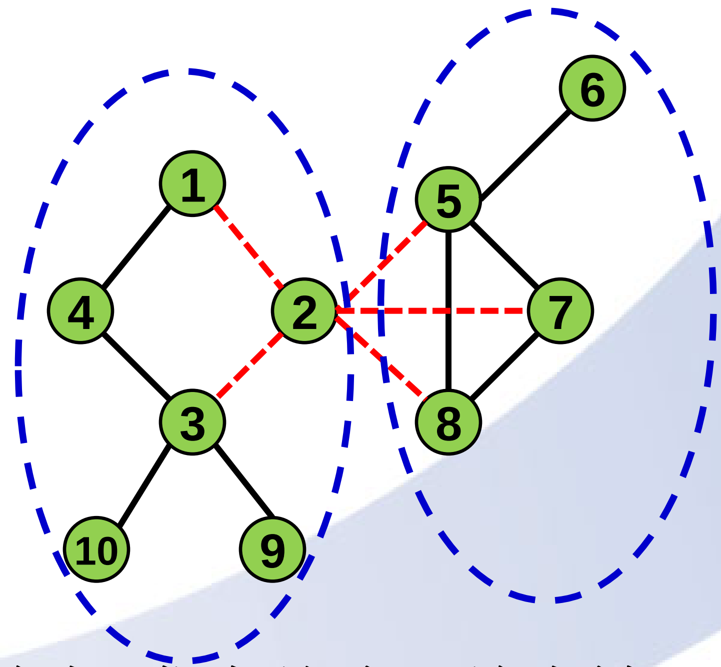


图2.11一个连通图



结点2发生故障，结点被“隔离”

2.3 双联通图与网络可靠性

■ 1. 问题描述

□通信网

➢图中结点表示通信站，边表示通信线路。

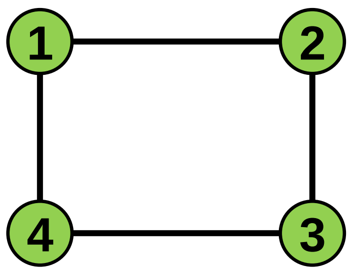
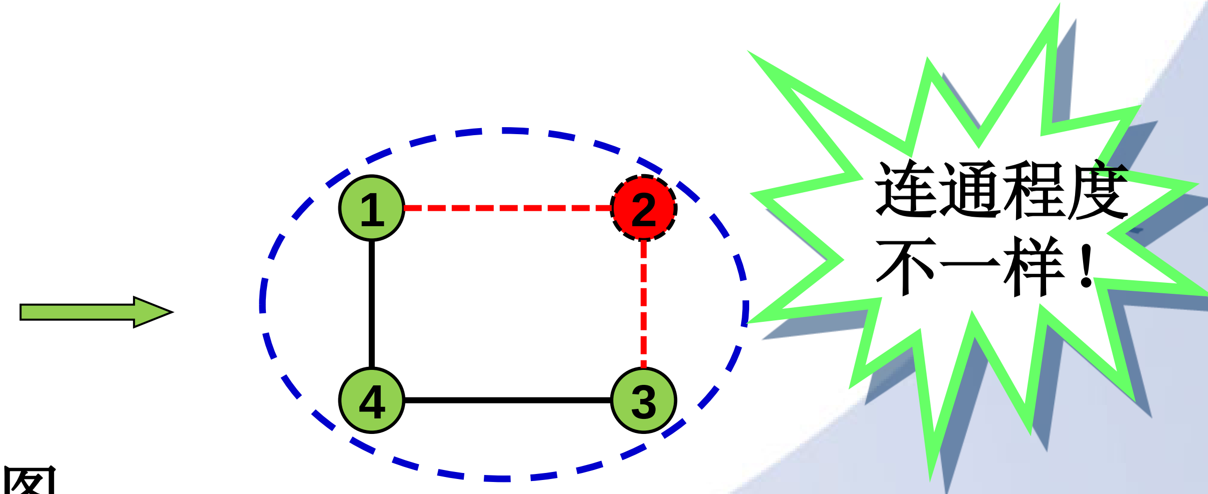


图2.12一个双连通图

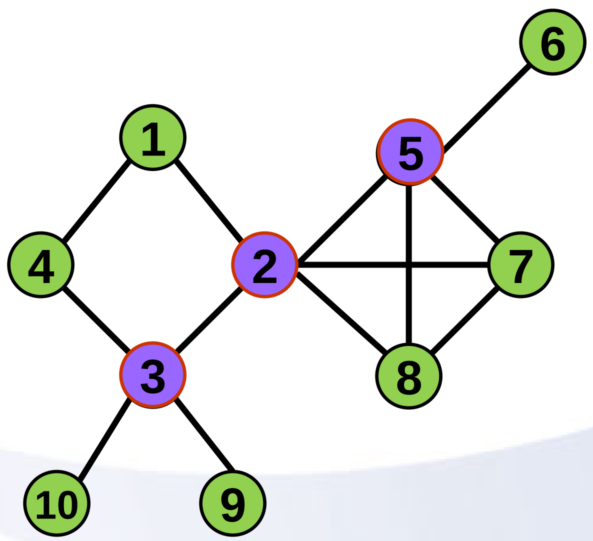


结点发生故障不影响
其它结点间的通信

2.3 双联通图与网络可靠性

■ 1. 问题描述

- 关节点：
  - 无向连通图中某结点a以及a相关联的所有边删除，得到两个或两个以上的非空分图，则a称为G的关节点。
  - 关节点影响到无向连通图的连通性能。



结点2, 3, 5都是关节点

2.3 双联通图与网络可靠性

■ 1. 问题描述

□ 双连通图：

- ➢如果无向连通图G不含关节点，则称G为双连通图。
- ➢可靠的通信网应该是双连通的。

□ 问题：

- ➢对一个无向连通图，如何判断它是不是双连通的？
- ➢如果存在关节点，如何识别关节点，并将非双连通的图改造成为双连通的？

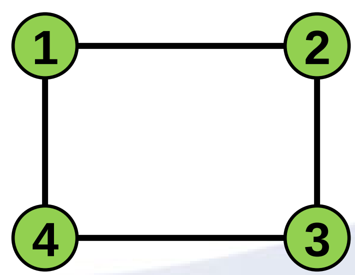


图2.12一个双连通图

2.3 双联通图与网络可靠性

- **1. 问题描述**
  - 目标:
    ① 设计一个算法测试某个连通图是否双连通
    ② 不是双连通的，找出所有的关节点
    ③ 在找出所有关节点的基础上，确定一个适当的边集加到G上，将其变为一个双连通图

2.3 双联通图与网络可靠性

■ 2. 双连通分图

- □最大双连通子图称为双连通分图。G’=(V’, E’)是G的最大双连通子图是指G中再没有这样的双连通子图G’’=(V’’, E’’)存在，使得 $E' \subseteq E''$ 且 $V' \subseteq V''$

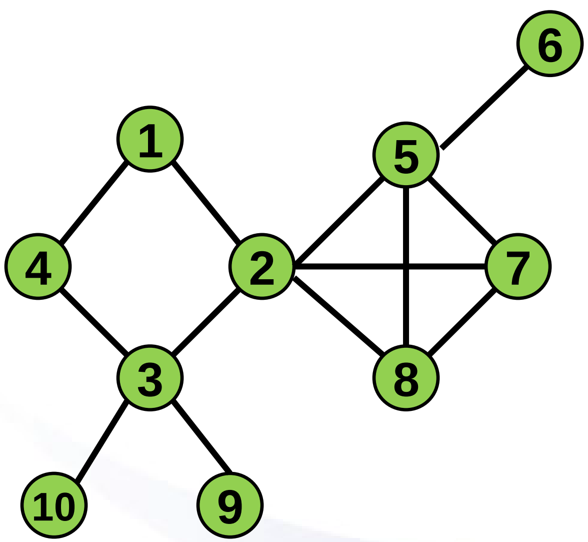
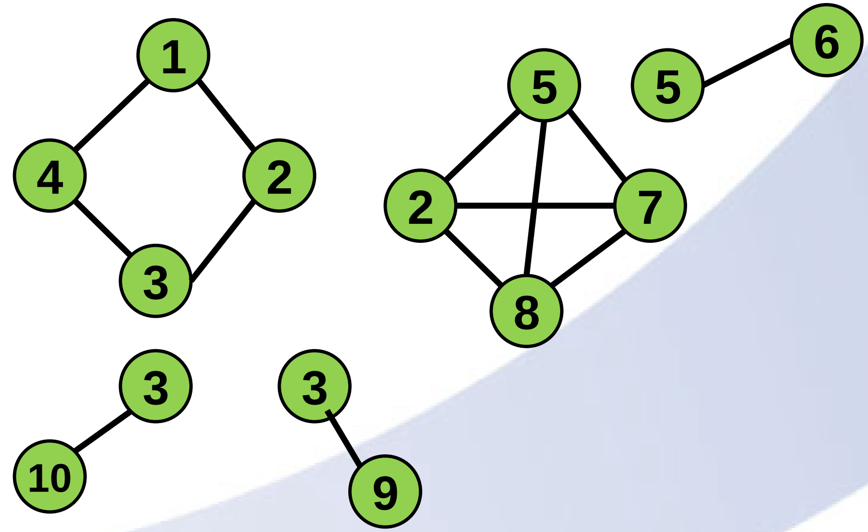


图2.11一个连通图



2.3 双联通图与网络可靠性

■ 2. 双连通分图

□双连通分图性质：

- ➢双连通图只有一个双连通分图，就是它自身。
- ➢两个双连通分图至多有一个公共结点，且这个结点是关节点。
- ➢任何一条边不可能同时在两个不同的连通分图中（因为这需要两个公共结点）。

2.3 双联通图与网络可靠性

■ 2. 双连通分图

□ 问题一：如何把一个非双连通的无向连通图变成双连通图？

➢通过加边，使双连通分图间有其它的边相连接。

for 每一个关节点a **do**

设$B_1, B_2, B_3, \ldots, B_k$是包含结点a的双连通分图

设v_i是B_i的一个结点，且$v_i \neq a$, $1 \leq i \leq k$

将(v_i, v_{i+1}), $1 \leq i < k$, 加到G

repeat

2.3 双联通图与网络可靠性

■ 2. 双连通分图

□ 问题一：如何把一个非双连通的无向连通图变成双连通图？

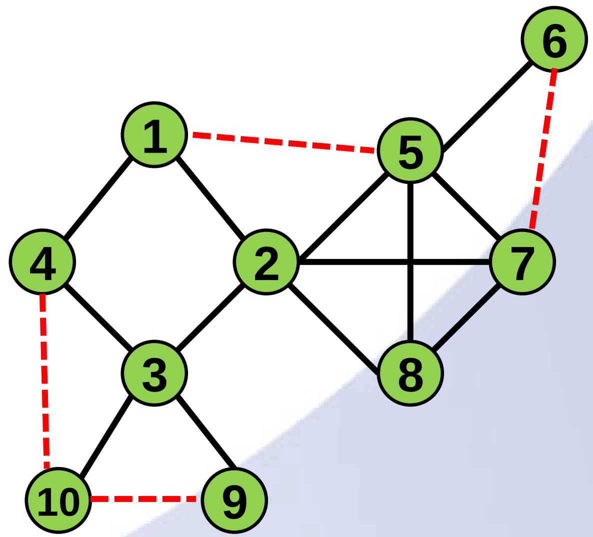
将G变为双连通图

图2.11中

关节点3：增加边 (4, 10) (10, 9)

关节点2：增加边 (1, 5）

关节点5：增加边 (6, 7）



在图2.11中增加边

2.3 双联通图与网络可靠性

- **2. 双连通分图**
  - 问题一：如何把一个非双连通的无向连通图变成双连通图？
    - 说明：
      - ✓设G有p个关节点，而与每个关节点对应的双连通分图数为k_i个，$1 \le i \le p$。则增加的边数总共为

$$\sum_{1 \le i \le p}(k_i - 1)$$

      - ✓此方法增加的总边数比将G变成双连通图所需要的最小边数大

2.3 双联通图与网络可靠性

■ 3. 深度优先生成树

- □ 问题二：如何识别图的关节点和双连通分图？
  - ➢策略：利用深度优先检索和深度优先生成树解决图的关节点与双连通分图的识别问题

深度优先树(DFN)：

按深度优先检索时访问结点的次序号。

例: **DFN(1)=1, DFN(2)=6**

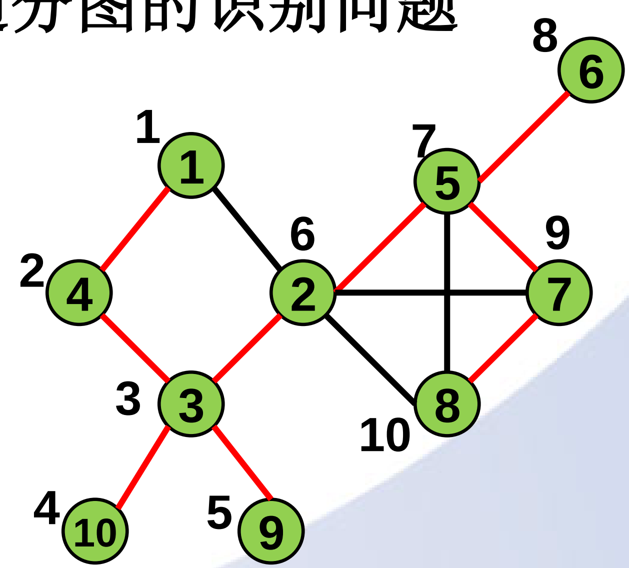


图2.11各结点的DFN

2.3 双联通图与网络可靠性

■ 3. 深度优先生成树

□ 问题二：如何识别图的关节点和双连通分图?

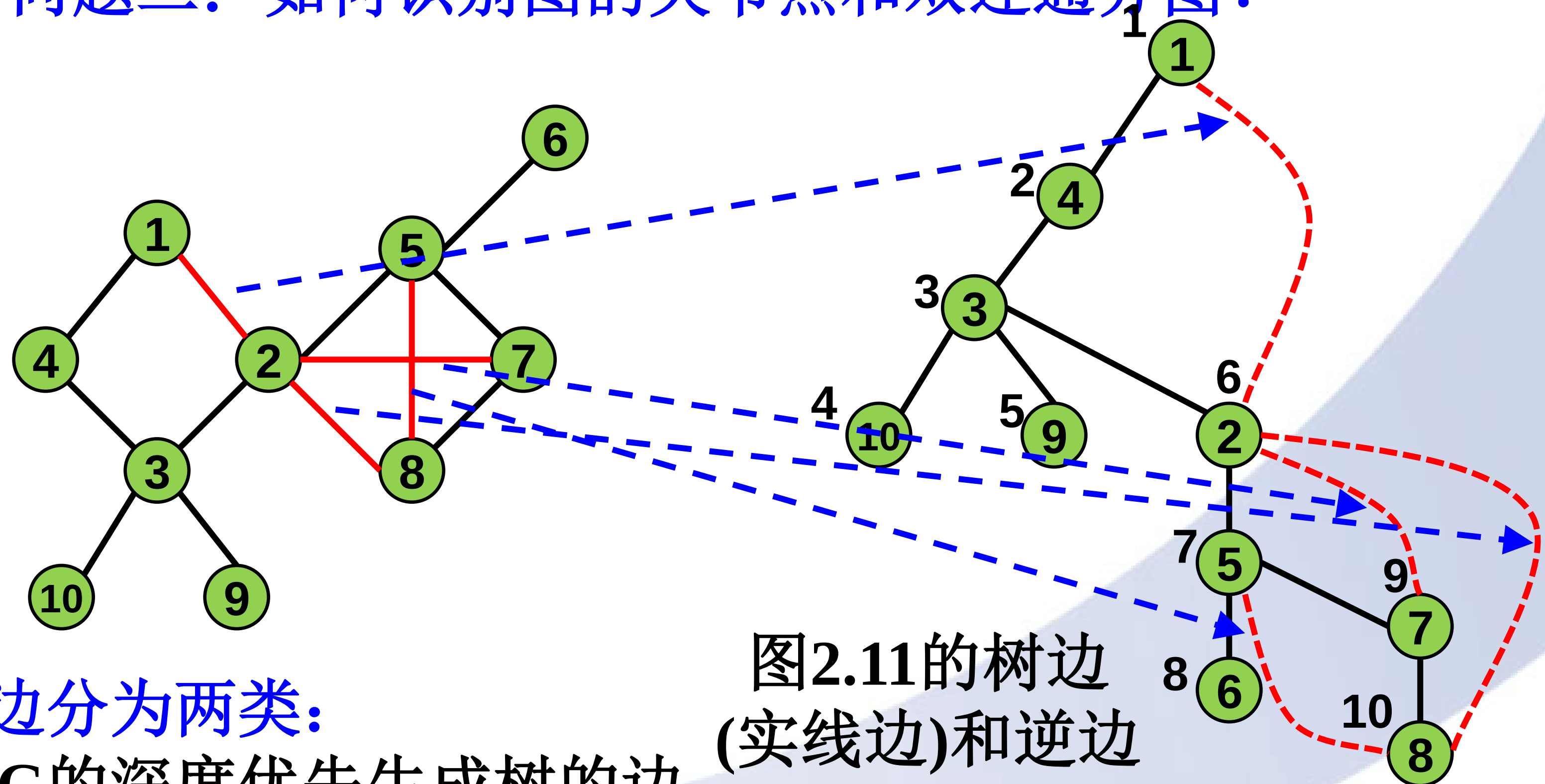


图2.11的树边
(实线边)和逆边
(虚线边)

将图G中的边分为两类:

树边：构成G的深度优先生成树的边。

逆边：不在生成树中的边

2.3 双联通图与网络可靠性

- 3. 深度优先生成树
  - 深度优先生成树的性质：
    - 性质1：若(u, v)是G中任一条边，则相对于深度优先生成树T，或者u是v的祖先，或者v是u的祖先。

中国科学院大学
University of Chinese Academy of Sciences

2.3 双联通图与网络可靠性

- **3. 深度优先生成树**
  - 深度优先生成树的性质：
    - 性质2：
      - ✓当且仅当一棵深度优先生成树的根结点至少有两个儿子时，此根结点是关节点；
      - ✓如果u是除根外的任一结点，那么，当且仅当由u的每一个儿子w出发，若只通过w的子孙组成的一条路径和一条逆边就可到达u的某个祖先时，则u就不是关节点。
      - ✓换言之，若对于u的某个儿子w不能做到这一点，则u为关节点。因为删除u至少会产生两个非空分图：一个包含根，一个包含结点w。

2.3 双联通图与网络可靠性

■ 3. 深度优先生成树

□深度优先生成树的性质：

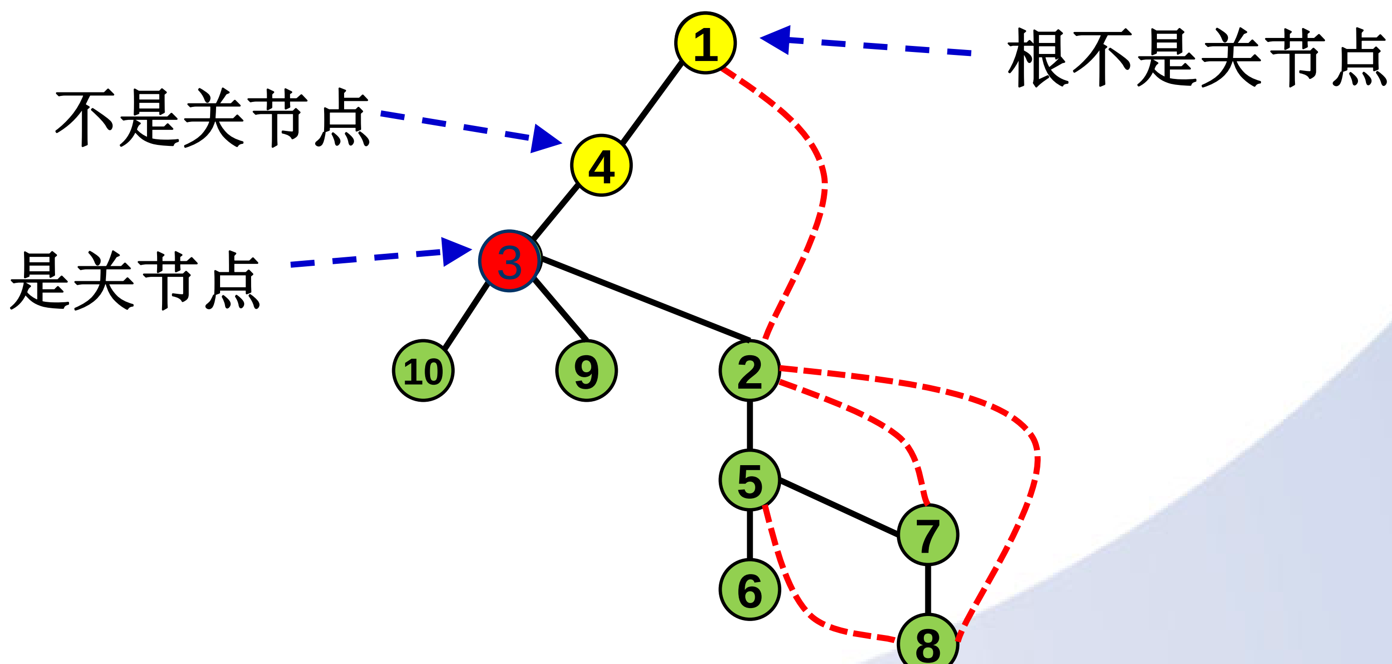


图2.11的树边(实线边)和逆边(虚线边)

2.3 双联通图与网络可靠性

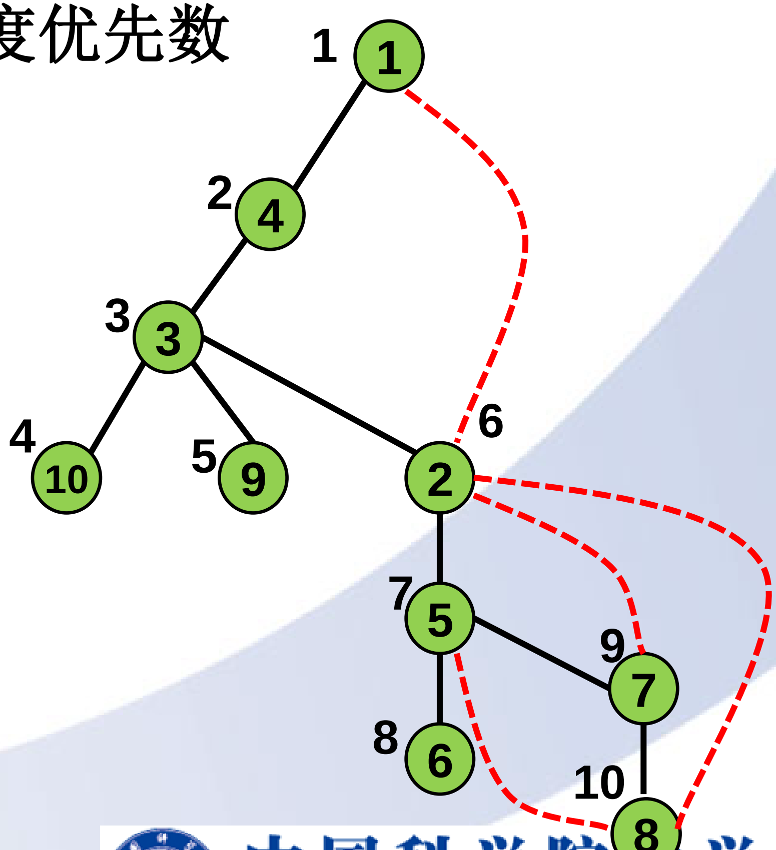
- ■ 4. 识别关节点和双连通分图
  - □ 最低深度优先数L(u):
    - ➢ 定义:
      L(u)=min{DFN(u), min{L(w)|w是u的儿子},
      min{DFN(w)|(u, w)是一条逆边} }
    - ➢ 注：L(u)是u通过一条子孙路径且至多后随一条逆边所可能到达的最低深度优先数。
    - ➢ 计算L(u)的方法:
      - ✓ 按后根次序访问深度优先生成树的结点，根据结点的DFN计算L。

2.3 双联通图与网络可靠性

i	1	2	3	4	5	6	7	8	9	10
L(i)	1	1	1	1	6	8	6	6	5	4

各结点的最低深度优先数

	DFN(u)	min{L(w)}	min{DFN(w)}
10	4	---	---
9	5	---	---
6	8	---	---
8	10	---	6
7	9	6	6
5	7	6	10
2	6	6	1
3	3	1	---
4	2	1	---



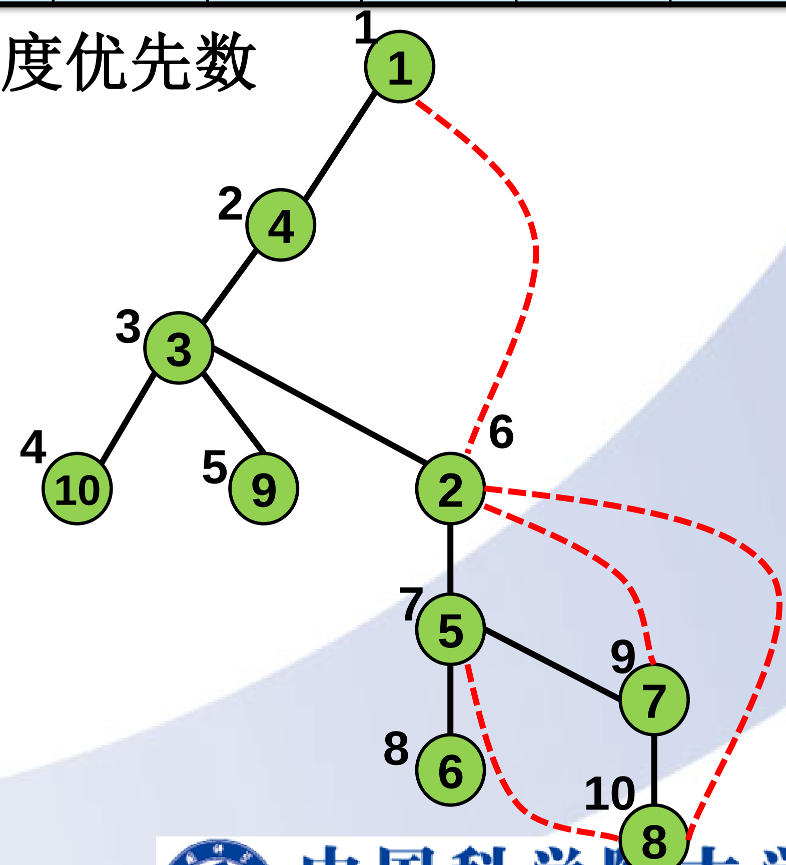
2.3 双联通图与网络可靠性

- **4. 识别关节点和双连通分图**
  - 关节点判定条件：
    - 如果**u**不是根，那么当且仅当
      - ✓**u**有一个儿子**w**，使得**L(w) ≥ DFN(u)**时，**u**是一个关节点。

2.3 双联通图与网络可靠性

i	1	2	3	4	5	6	7	8	9	10
L(i)	1	1	1	1	6	8	6	6	5	4

各结点的最低深度优先数



关节点：

结点3：它的儿子结点10有L(10)=4而DFN(3)=3。

结点2：儿子结点5有L(5)=6而DFN(2)=6。

结点5：儿子结点6有L(6)=8而DFN(5)=7。

中国科学院大学
University of Chinese Academy of Sciences

2.3 双联通图与网络可靠性

- 4. 识别关节点和双连通分图
  - 确定G的关节点的工作：
    - ① 完成对G的深度优先搜索，产生G的深度优先生成树T。
    - ② 按后根次序访问树T的结点，根据结点的DFN计算L值，然后再进行关节点的判定。
  - 对G的深度优先检索和计算DFN和L可同时进行。算法如下：

2.3 双联通图与网络可靠性

■ 4. 识别关节点和双连通分图

算法2.11计算DFN和L的算法

```
procedure  ART(u, v)
//u是开始结点。在深度优先生成树中，u若有父亲，则v是其父亲。Num=1//
  global DFN(n), L(n), num, n
  DFN(u) ← num; L(u) ← num; num ← num+1
  for 每个邻接于u的结点w do
     if DFN(w)=0  then
          call ART(w, u)    //还没访问w//
          L(u) ← min(L(u), L(w))
     else
          if w ≠ v then L(u) ← min(L(u), DFN(w)) endif
     endif
   repeat
end ART
```

深度优先树上的结点

w已经访问过，和u相邻接，又不是u的父亲v

2.3 双联通图与网络可靠性

■ 4. 识别关节点和双连通分图

□ 算法分析：

- 设图G有n个结点e条边，G由邻接表表示，那么ART的计算时间为O(n+e)。
- 因此L(1:n)可在时间O(n+e)内算出。
- 一旦算出L(1:n)，G的关节点就能在O(n)时间内识别出来。
- 因此识别关节点的总时间不超过O(n+e)

2.3 双联通图与网络可靠性

■ 4. 识别关节点和双连通分图

□判断G的双连通分图方法：

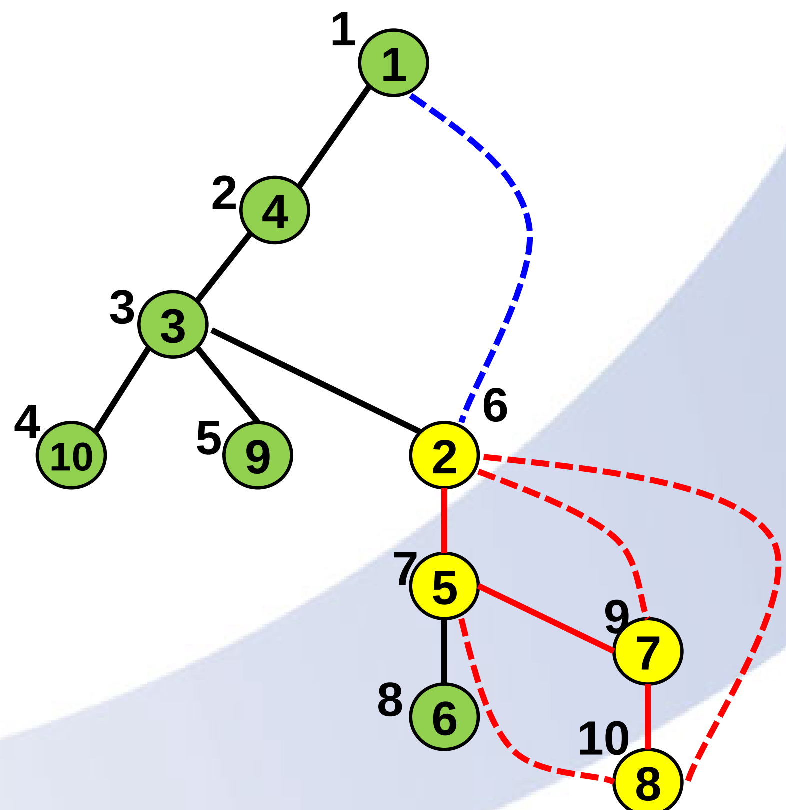
- ➢在第三行调用ART之后有L(w) ≥ DFN(u)，就可断定u或者是根，或者是关节点。
- ➢不管u是否是根，也不管u有一个或是多个儿子，将边(u, w)和对ART的这次调用期间遇到的所有树边和逆边加在一起，构成一个双连通分图。
- ➢对ART作一些修改即可生成识别双连通分图的算法，引进一个用来存放边的栈S

2.3 双联通图与网络可靠性

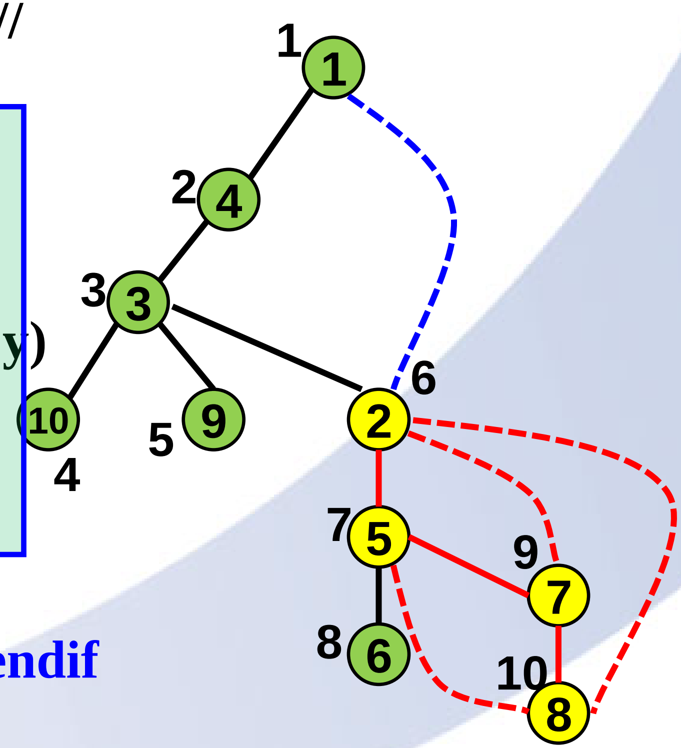
■ 4. 识别关节点和双连通分图

□判断G的双连通分图方法:

例2.11中结点2的儿子结点5有L(5)=6而DFN(2)=6



```
procedure  ART*(u, v)
    Global DFN(n), L(n), num, n
    DFN(u) ← num; L(u) ← num; num← num+1
    for 每个邻接于u的结点w do
        if v≠w and DFN(w)<DFN(u) then 将(u, w)加到S的顶部 endif
                                        // DFN(w)=0或者(u, w)是逆边//
        if DFN(w)=0  then                            //还没访问w//
            call ART(w, u)
            if L(w) ≥DFN(u) then        找到了关节点
              print('new biconnected componet')
              loop
                从栈S的顶部删去一条边,设这条边是(x, y)
                print('(',x,', ',y,')')
              until((x, y)=(u, w) or (x, y)=(w, u)) repeat
            endif
            L(u) ← min(L(u), L(w))
        else  if w ≠ v then L(u) ← min(L(u), DFN(w)) endif
        endif
    repeat
end ART*
```



2.3 双联通图与网络可靠性

■ 4. 识别关节点和双连通分图

- 定理2.10 当连通图G至少有两个结点时，增加了上述语句后，ART算法能够正确生成G的双连通分图。
- 证明：
  - 数学归纳法。
  - 证明略。
- 注意：上述算法要求，相对于生成树，所给定的图没有交叉边。而相对于宽度优先生成树，一些图可能有交叉边，此时算法ART对BFS不适用。

End