算法设计
——图遍历

陈卫东

chenwd@scnu.edu.cn

华南师范大学计算机学院

2021-09

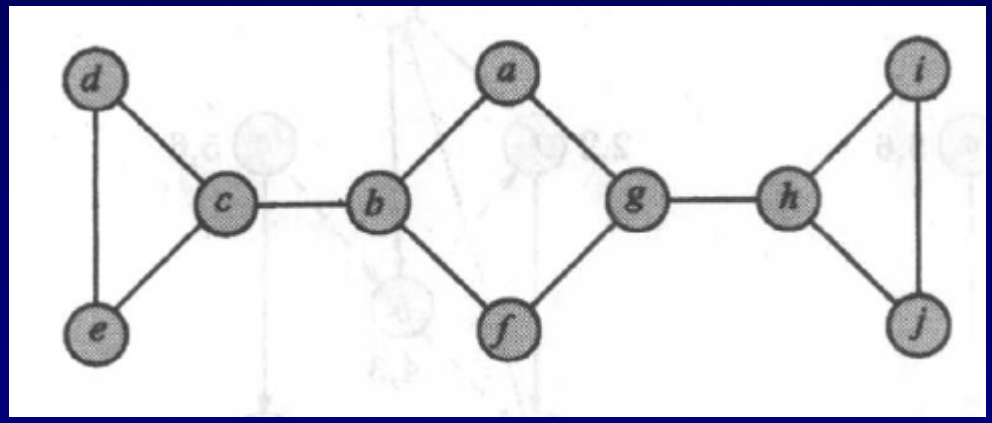
图的遍历

- 3.1 引言
- 3.2 深度优先搜索（Depth-First）
- 3.3 深度优先搜索的应用
- 3.4 宽度优先搜索（Breadth-First）
- 3.5 宽度优先搜索的应用

3.1 引言

- 图遍历的两种方法:
  - 深度优先搜索（Depth-First）
  - 宽度优先搜索（Breadth-First）

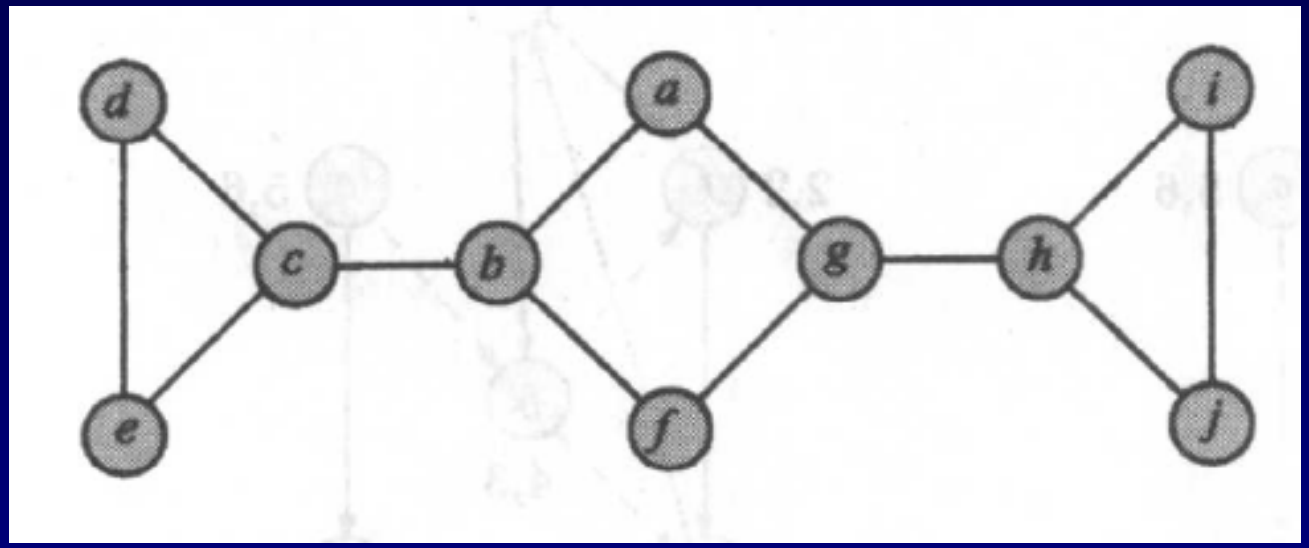
系统地访问图中所有结点的简单方法

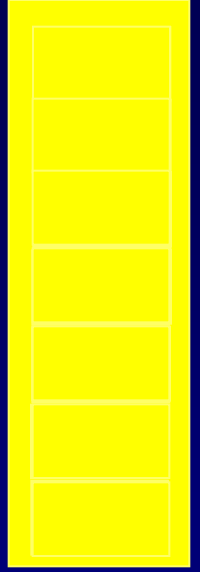


- 深度优先遍历序列:
  a, b, c, d, e, f, g, h, i, j
- 宽度优先遍历序列:
  a, b, g, c, f, h, d, e, i, j

如何描述这两种搜索方法？

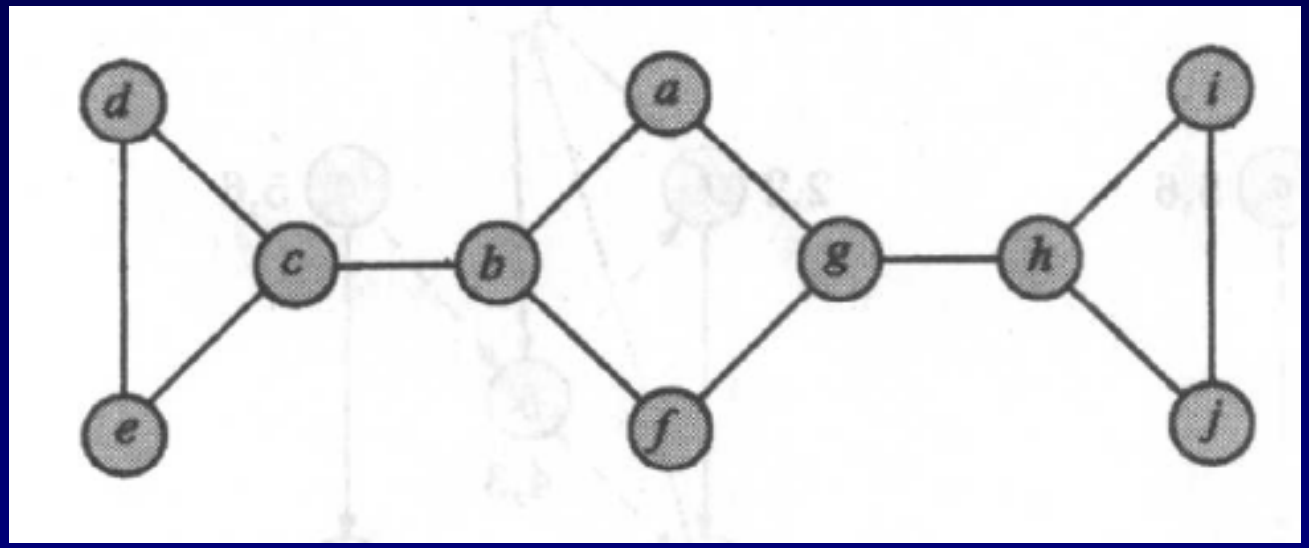
- 深度优先搜索、宽度优先搜索的基本思想
  - 被访问结点、未被访问结点
  - 被检测结点（死结点）
    ——该节点及其所有邻节点都已被访问的节点
  - 未被检测结点（活结点）
    ——该节点已被访问但尚有邻接点未被访问的节点

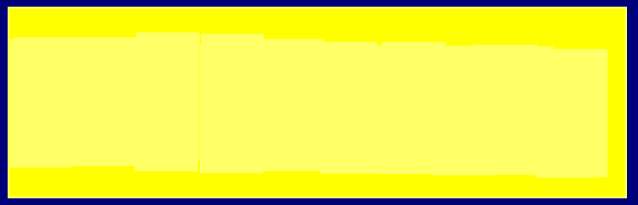




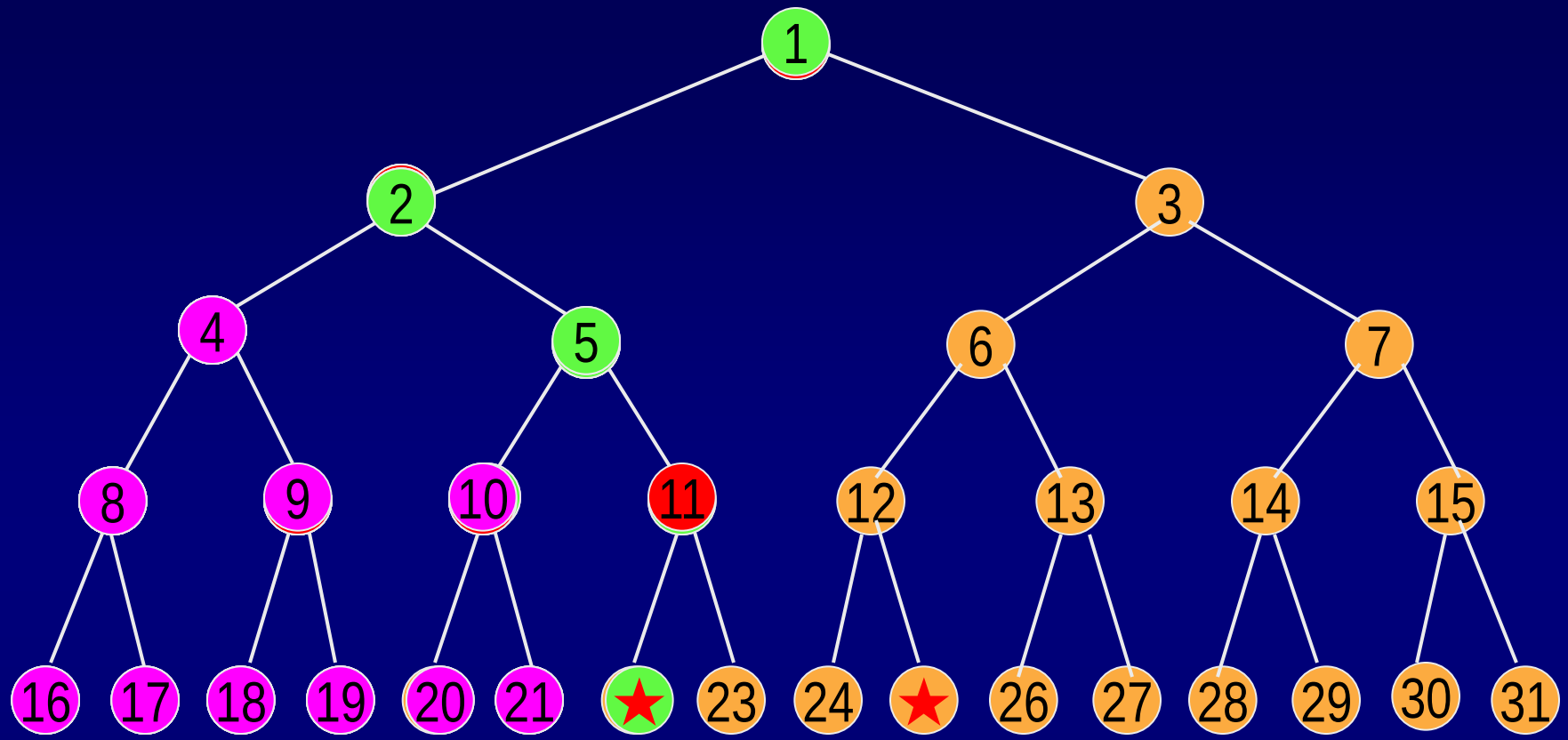
a b c d e f g h i j

- 深度优先搜索的基本思想
  - 采用栈存储活结点（活结点表）
    ——初始时栈中只有*a*，终止时栈为空
  - 每次检测栈顶元素
    ——访问它的一个未被访问的邻接点。
    如果没有这样的节点，删除栈顶元素（死结点）
  - 节点被访问立即入栈成新的栈顶元素



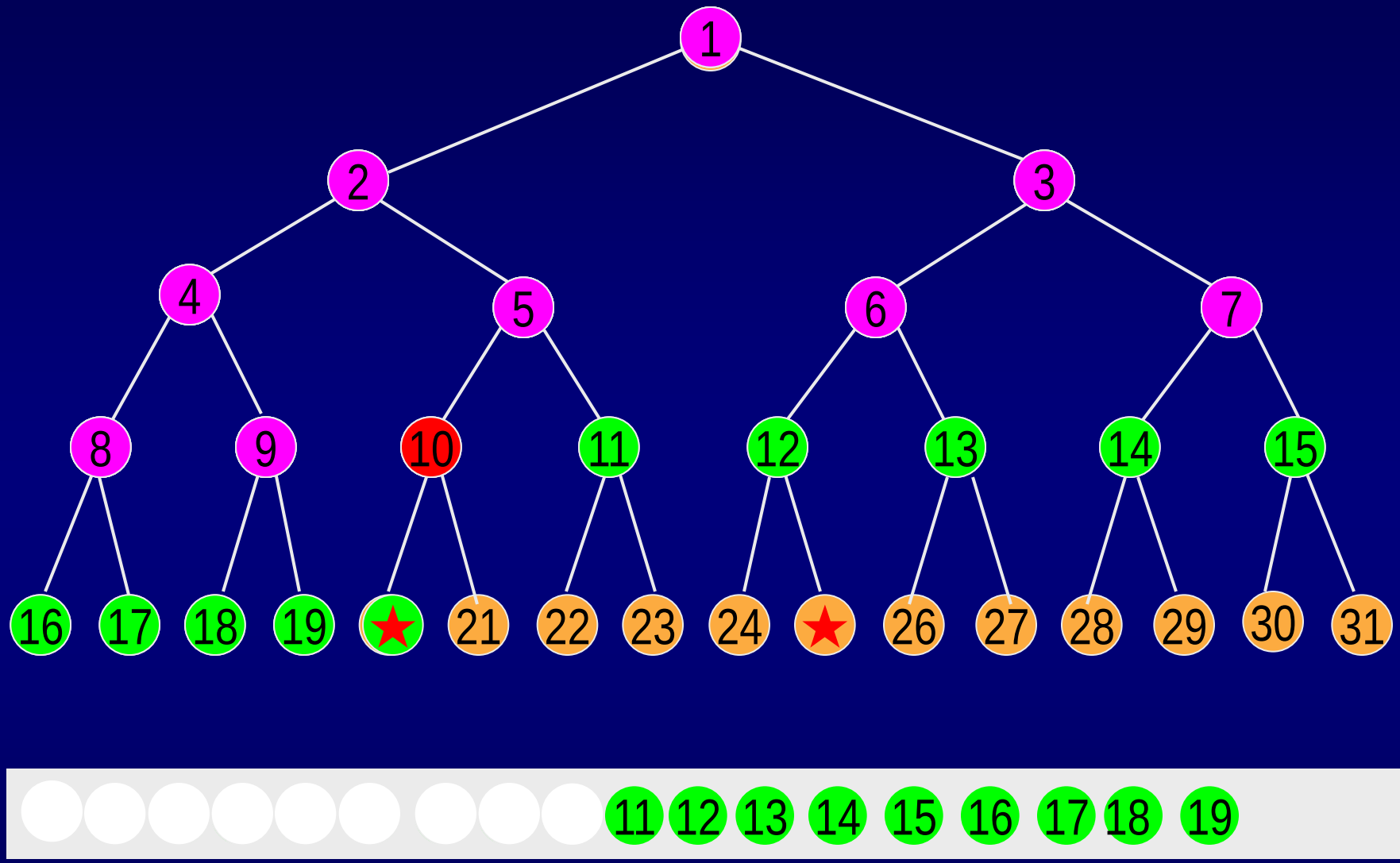


- 宽度优先搜索的基本思想
  - 采用队列存储活结点（活结点表）
    ——初始时队列中只有*a*，终止时队列为空
  - 每次检测队头元素
    ——一次性访问完它的所有未被访问的邻接点，然后删除队头元（死结点）
  - 节点被访问立即入队列

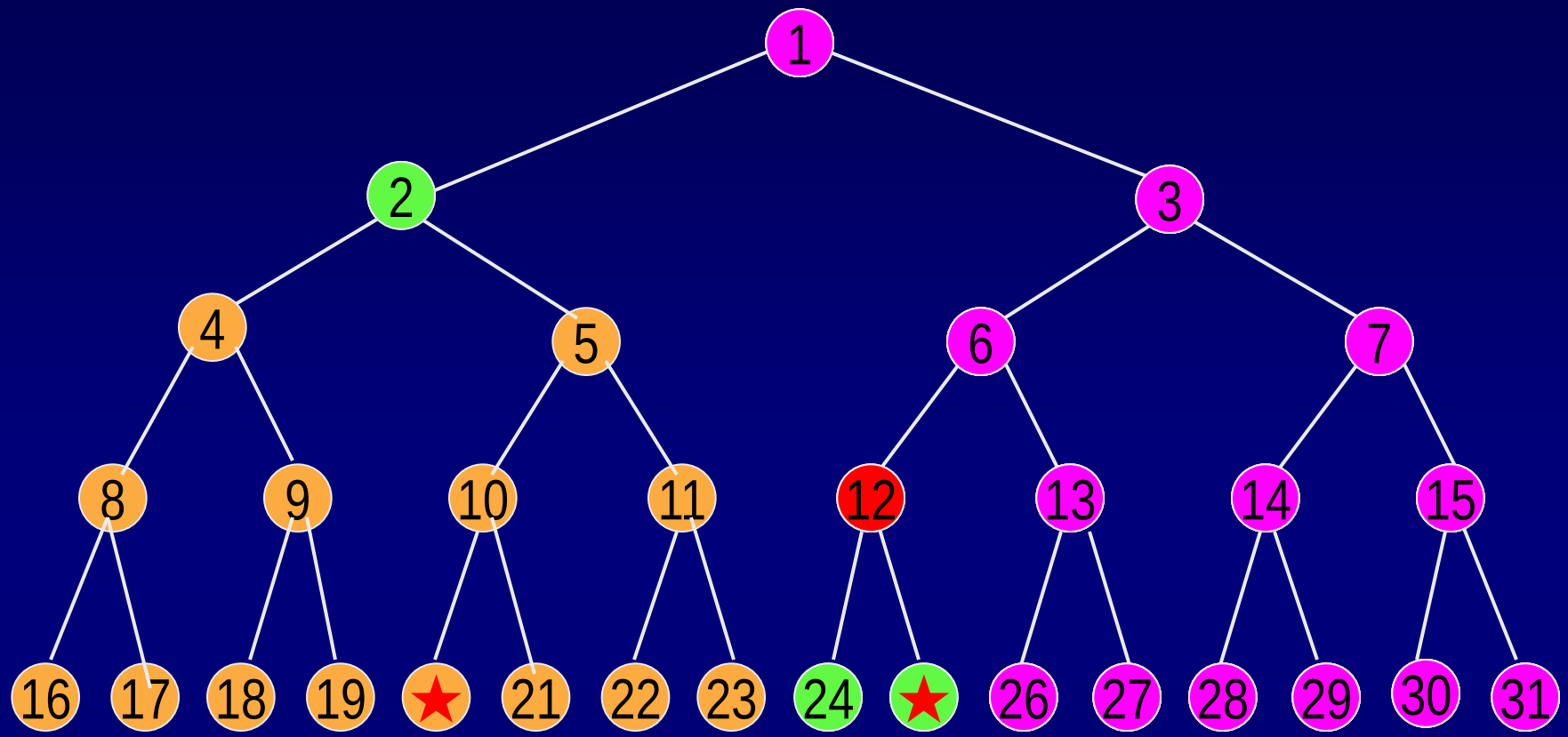


活结点表

深度优先搜索(DFS)示意图



宽度优先搜索(BFS)演示图



活结点表

D-搜索(D-Search)演示图

3.2 深度优先搜索

- 算法　DFS
- 时间复杂度
  - $\Theta(m+n)$ (邻接表)
  - $\Theta(n^2)$　(邻接矩阵)

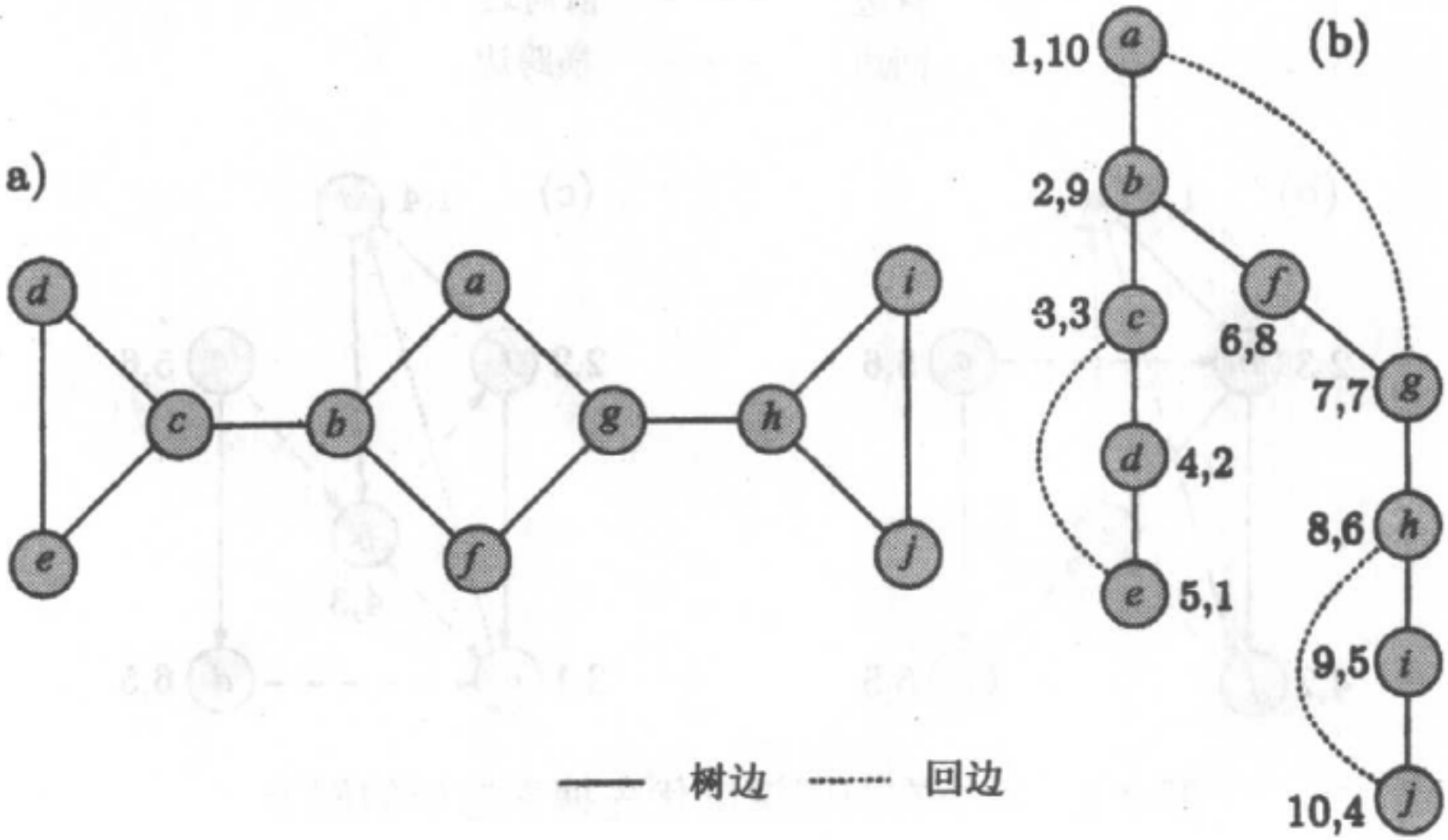
输入：有向或无向图 $G=(V,E)$。

输出：在相应的深度优先搜索树中对顶点的前序和后序。

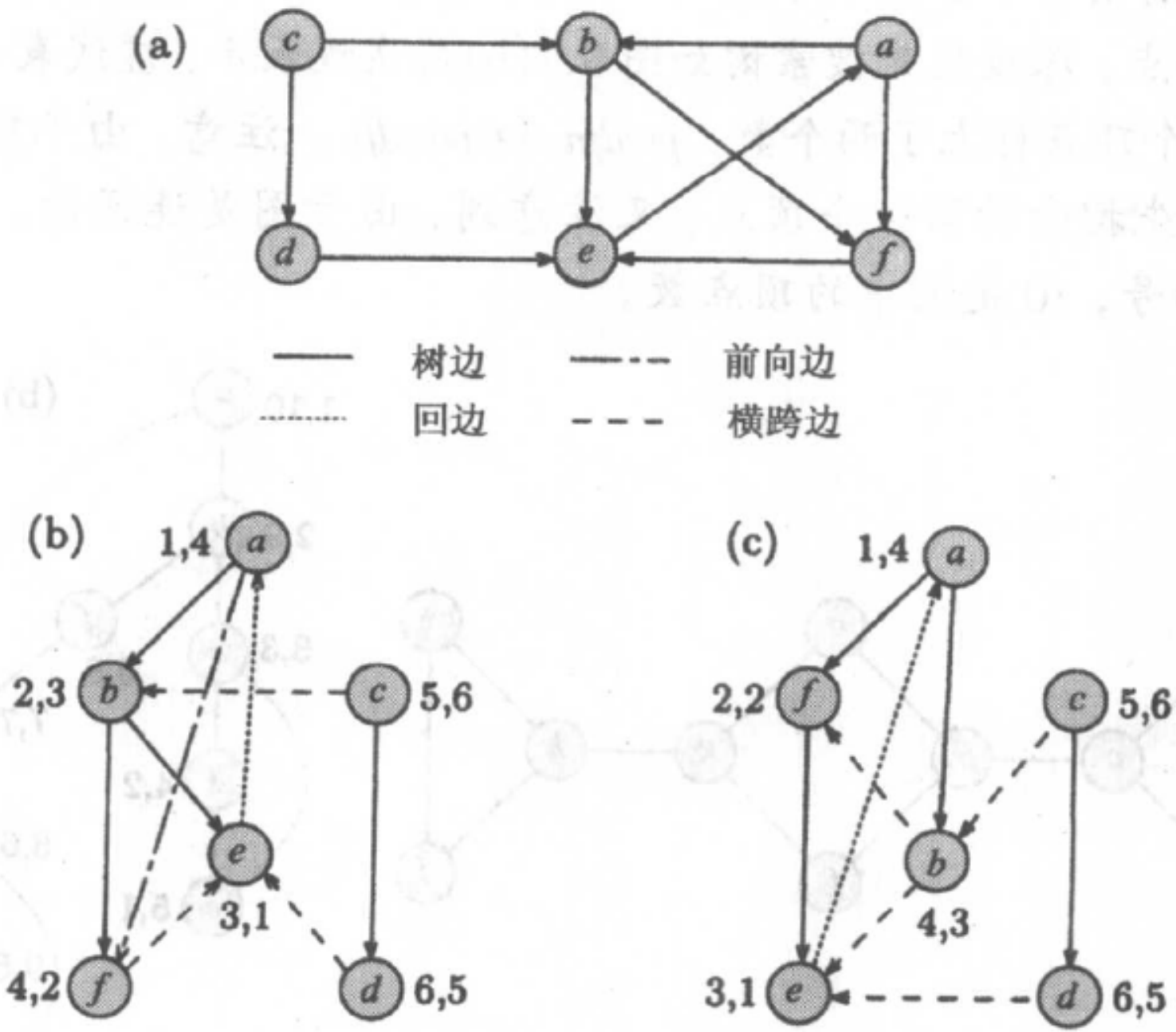
1. $predfn \leftarrow 0$; $postdfn \leftarrow 0$
2. **for** 每个顶点 $v \in V$
3. 　　标记 v 未访问
4. **end for**
5. **for** 每个顶点 $v \in V$
6. 　　**if** v 未访问 **then** $dfs(v)$
7. **end for**

过程　$dfs(v)$

1. 标记 v 已访问
2. $predfn \leftarrow predfn + 1$
3. **for** 每条边 $(v,w) \in E$
4. 　　**if** w 标记为未访问 **then** $dfs(w)$
5. **end for**
6. $postdfn \leftarrow postdfn + 1$



一个无向图深度优先搜索遍历的例子



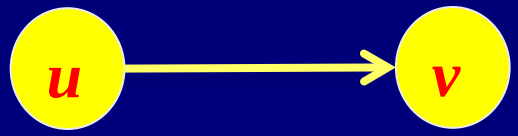
一个有向图深度优先搜索遍历的例子

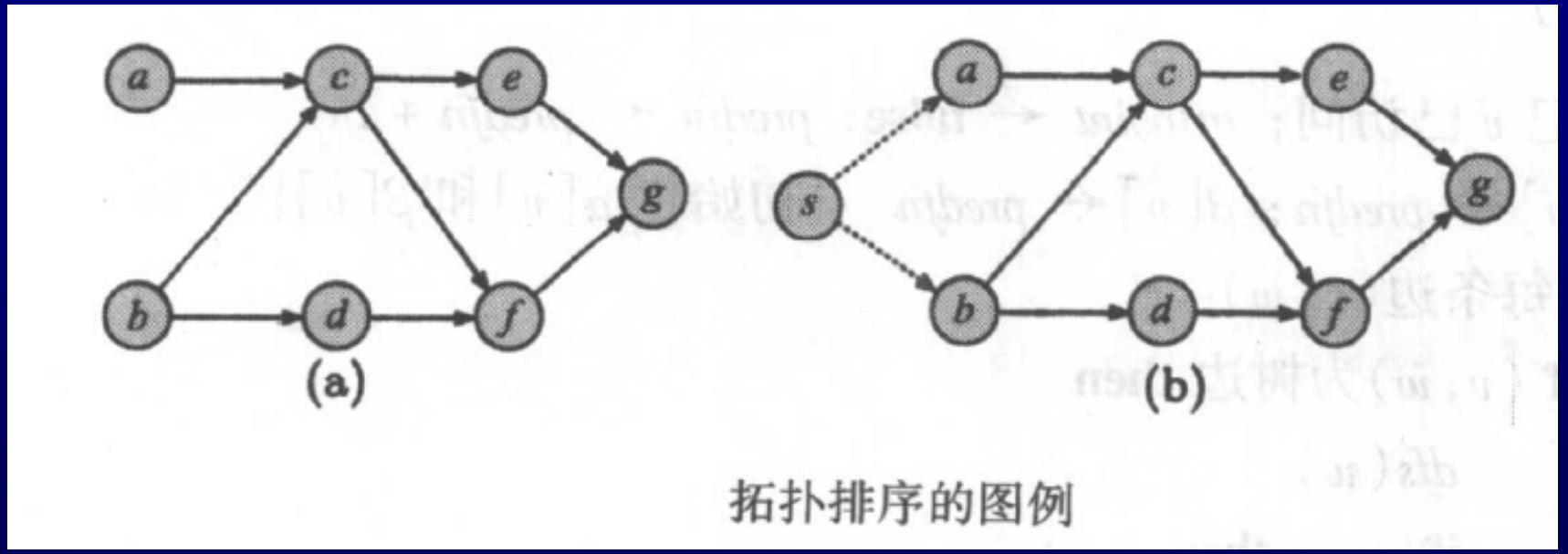
3.2 深度优先搜索

- 算法 DFS
  - 无向图的情形
    - -----树边（Tree edges）
    - -----回边（Back edges）
    - -----实例
  - 有向图的情形
    - -----树边, 回边, 向前边, 交叉边
    - -----实例

3.3 深度优先搜索的应用

- 判定图的无环性
- 拓扑排序

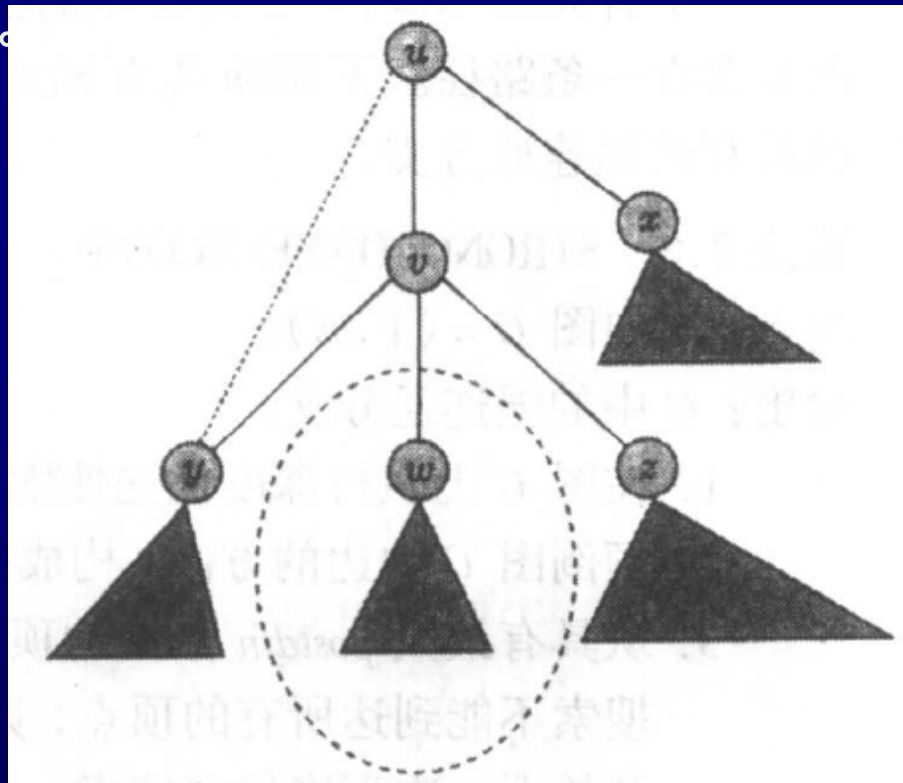




拓扑排序的图例

- 找图的拐点（关节点）
  - **α[v]=predfn:** 深度优先数。
  - **β[v]=min{α[v], min{β[w]|w is a son of v},**
    min{α[u]|(v,u) is a back edge} } :

    v通过一条子孙路径且至多后随一条回边（back edge）所能到达的最小深度优先数。

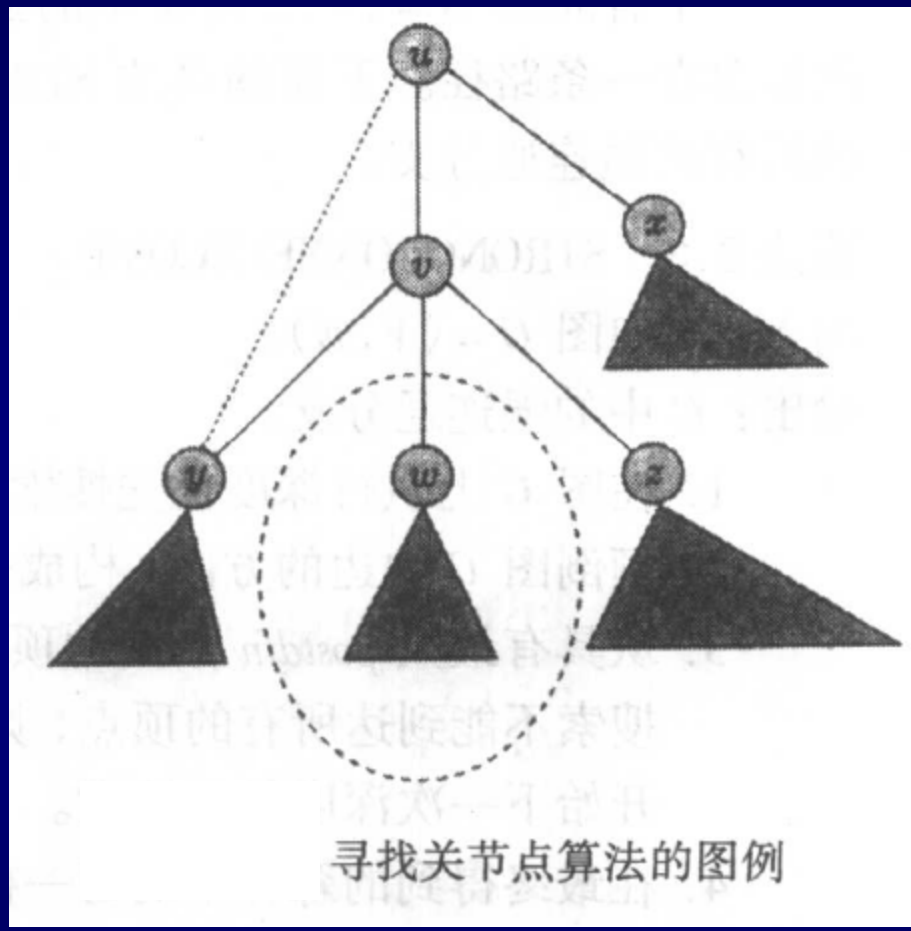


寻找关节点算法的图例

- 找图的拐点（关节点）
  - 拐点的判别方法：

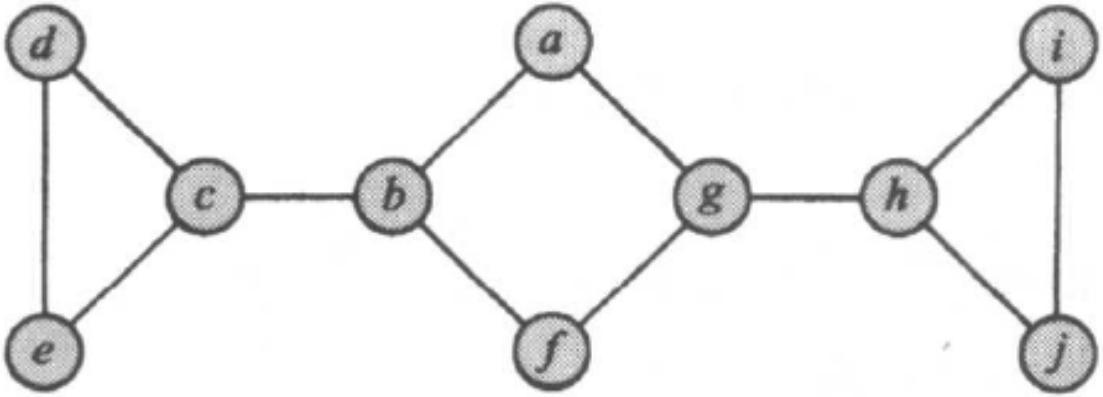
1. 当且仅当一棵深度优先搜索树的根结点至少有两个儿子时，此根结点是关节点。

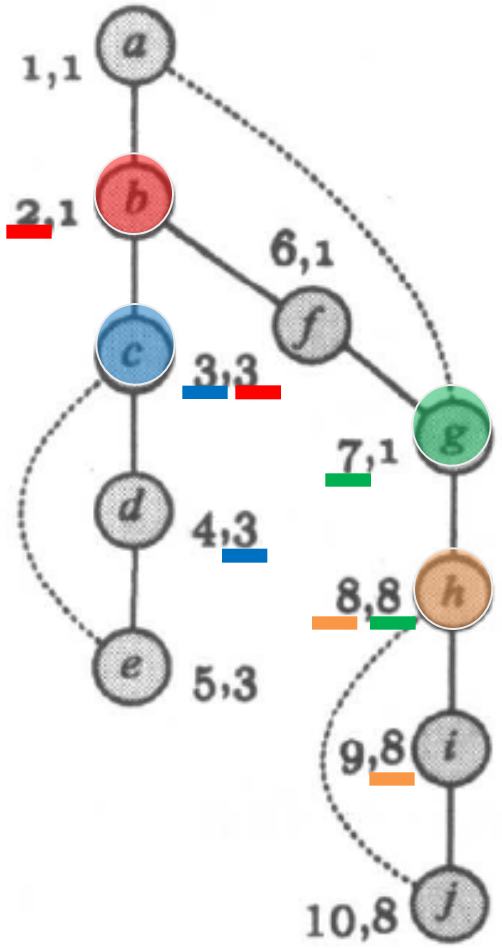
2. 如果v是除了根之外的任一结点，那么当且仅当由v的每一个儿子w出发，若只通过w的子孙组成的一条路径和一条回边就可达到v的某个祖先时，则v就不是关节点。即，当且仅当v有一个儿子w使得β[w]≥α[v]时，则v为关节点



寻找关节点算法的图例

当且仅当v有一个儿子w使得
β[w]≥α[v]时，则v为关节点





—— 树边 ····· 回边

在图中查找关节点的例子

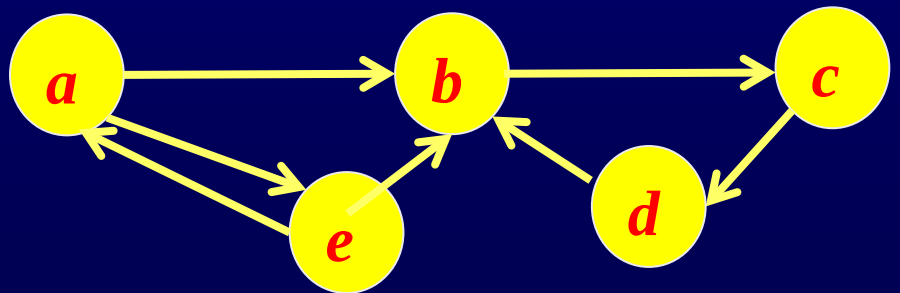
找强连通分图

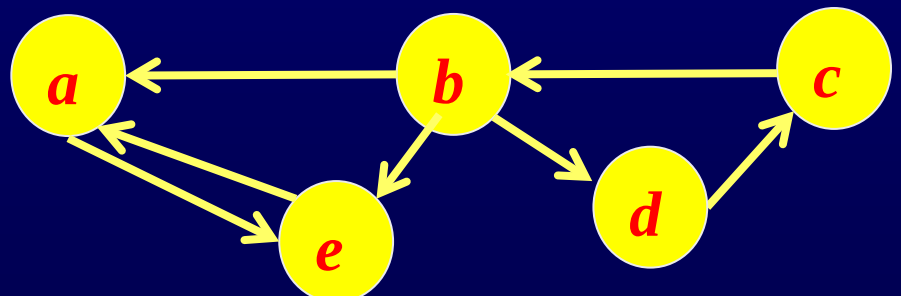
算法 STRONGCONNECTCOMP

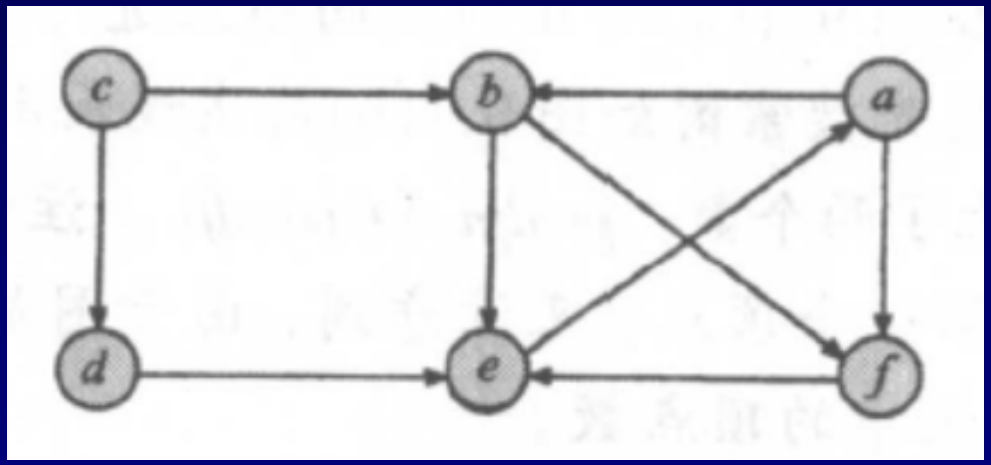
输入：有向图 $G=(V,E)$。

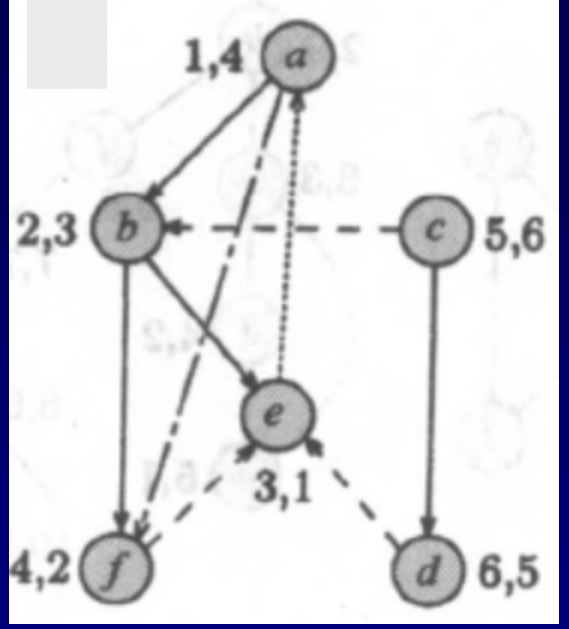
输出：G 中的强连通分支。

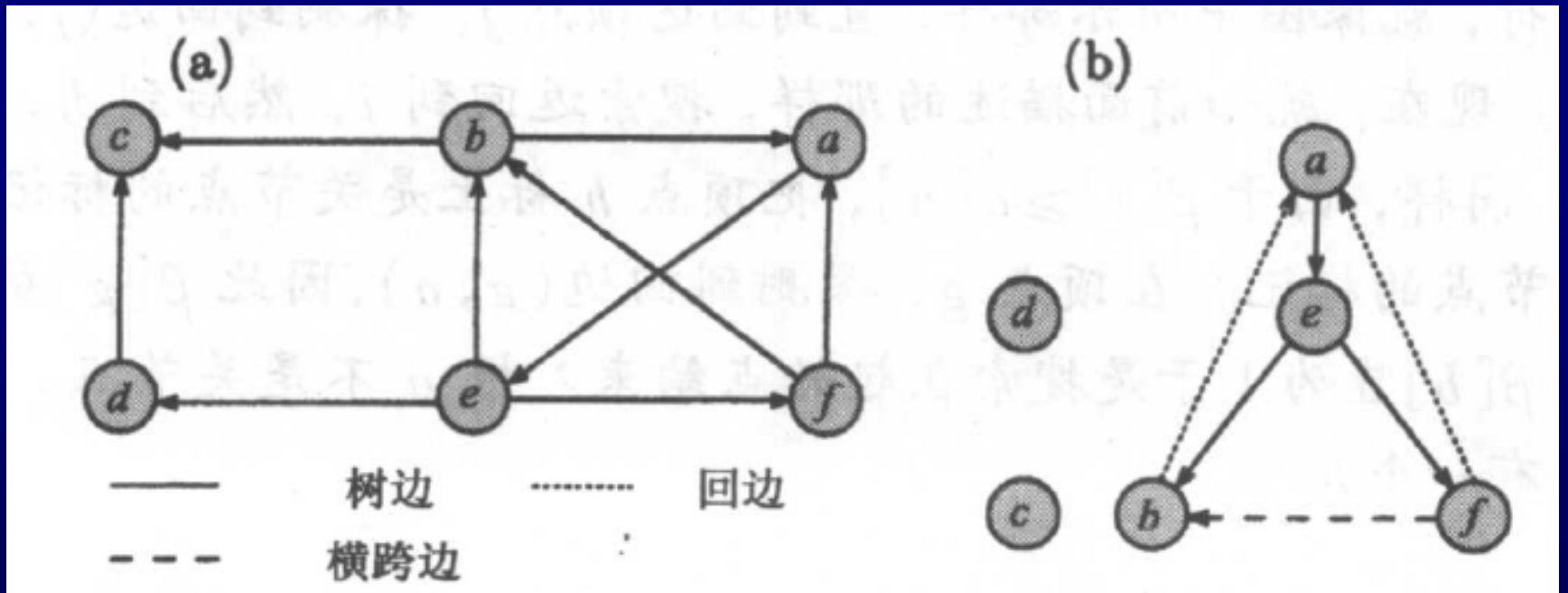
1. 在图 G 上执行深度优先搜索，对每一个顶点赋给相应的 *postdfn* 值。
2. 颠倒图 G 中边的方向，构成一个新的图 G'。
3. 从具有最大 *postdfn* 数值的顶点开始，在 G' 上执行深度优先搜索，如果深度优先搜索不能到达所有的顶点，则在余下的顶点中找一个 *postdfn* 数值最大的顶点，开始下一次深度优先搜索。
4. 在最终得到的森林中的每一棵树对应一个强连通分支。











寻找强连通分支

3.4 宽度优先搜索

- 算法 BFS
  - 实例
  - 时间复杂度

    $\Theta(m+n)$ (邻接表)

    $\Theta(n^2)$ (邻接矩阵)

算法　　BFS

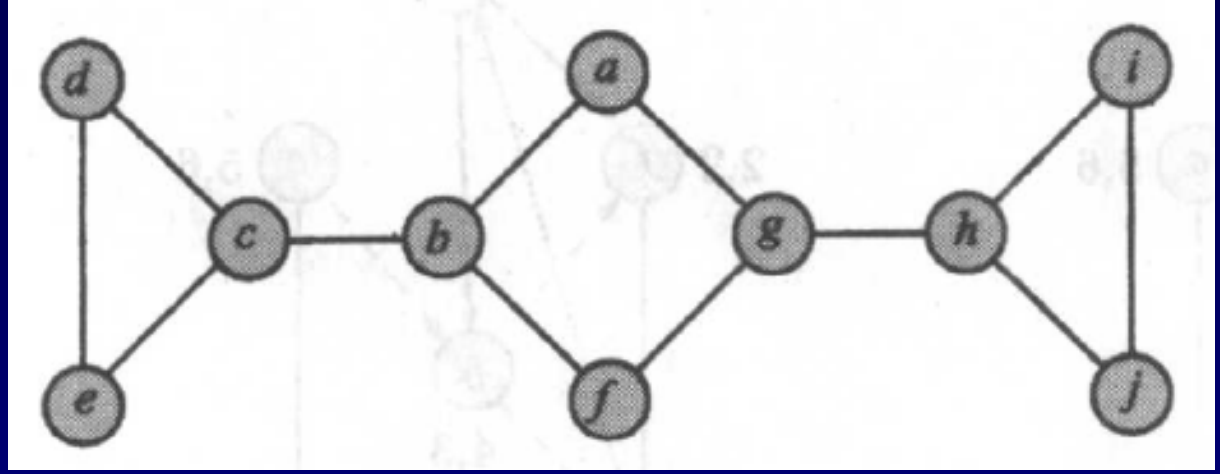
输入：有向或无向图 $G=(V,E)$。

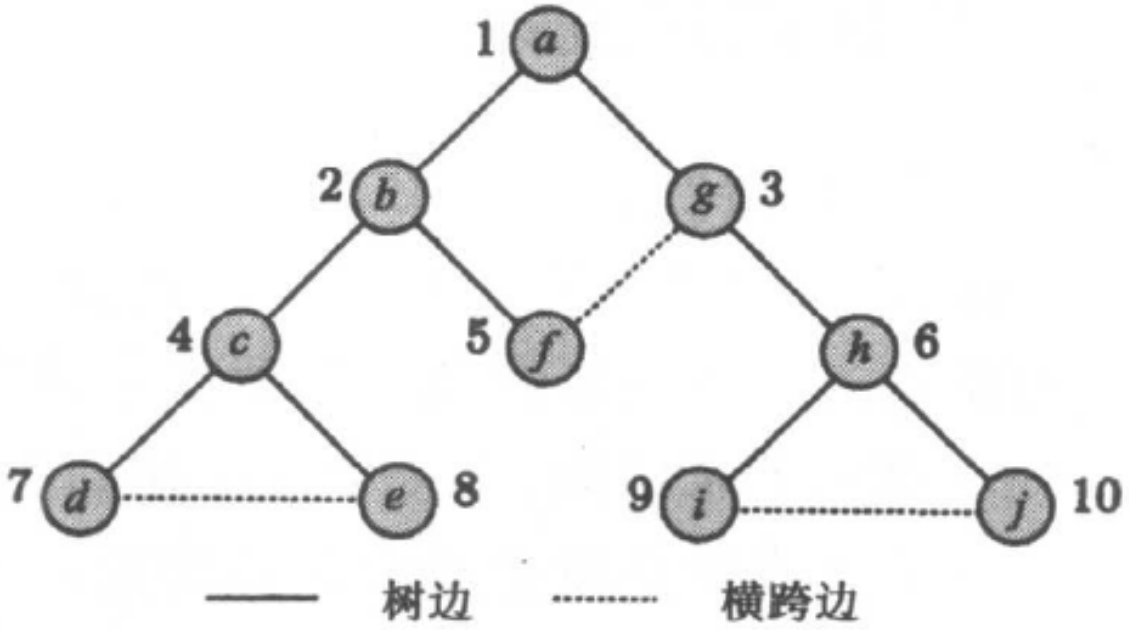
输出：广度优先搜索次序中顶点的编号。

```
1. bfn ← 0
2. for 每个顶点 v∈V
3.     标记 v 未访问
4. end for
5. for 每个顶点 v∈V
6.     if v 标记为未访问 then bfs(v)
7. end for
```

过程　$bfs(v)$

```
1. Q ← {v}
2. 标记 v 已访问
3. while Q≠{}
4.     v ← Pop(Q)
5.     bfn ← bfn + 1
6.     for 每条边(v,w)∈E
7.         if w 标记为未访问 then
8.             Push(w,Q)
9.             标记 w 已访问
10.        end if
11.    end for
12. end while
```





一个无向图广度优先搜索遍历的例子

3.5 宽度优先搜索的应用

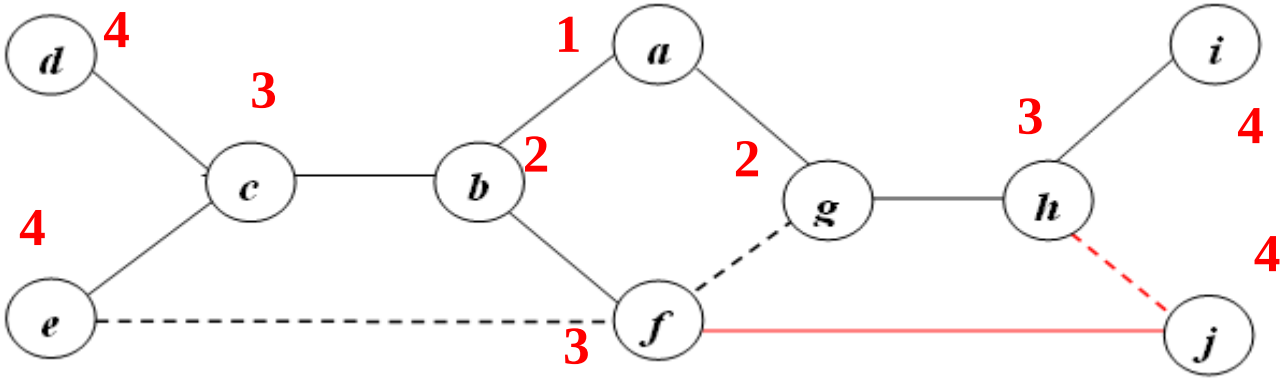
- 求图中结点*s* 到其它每个结点的距离

（*s* 到*v* 的距离被定义*s* 到 *v*的任何路径的最少边数）

当将算法 BFS 用于图*G* 中且指定起点为*s* 时，所得树（宽度优先搜索树）中*s* 到其他任何结点有最少的边数。

请设计一个有效算法来确定一个给出的图是否是二分图

- 采用宽度优先遍历图 G，得到宽度优先生成树。如果宽度优先生成树的同层节点没有边出现，则图 G 为二部图，否则图 G 不是二部图。
- 在图 G 的宽度优先生成树中，G 中任何边必跨于相邻层节点之间或同层节点之间，前者是树边，后者是横跨边。如果宽度优先生成树的同层节点没有边，则图 G 为二部图。反之，若图 G 是二部图，则必有其宽度优先生成树同层节点之间无边。算法正确性证毕。



宽度优先遍历图 G 得宽度优先生成树（实心边是树边，虚线是横跨边）

请设计一个有效算法来确定一个给出的图是否是二分图

如何采用图的深度优先遍历方法设计算法?