Design and Analysis of Algorithms

Part IV: Graph Algorithms

Lecture 23: DFS on Directed Graphs

童咏昕

北京航空航天大学
计算机学院

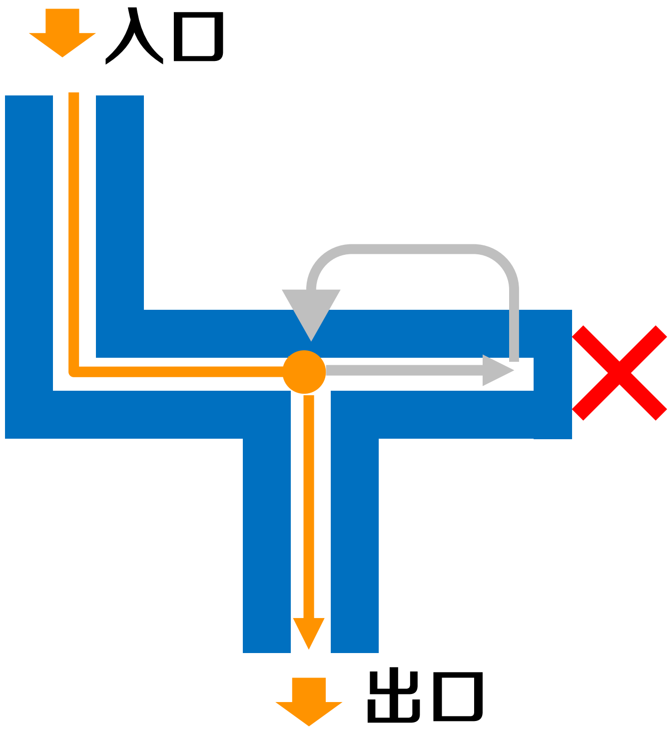
图算法篇概述

- 在算法课程第四部分“图算法”主题中，我们将主要聚焦于如下经典问题：
  - Basic Concepts in Graph Algorithms（图算法的基本概念）
  - Breadth-First Search (BFS，广度优先搜索）
  - Depth-First Search (DFS，深度优先搜索）
  - Cycle Detection（环路检测）
  - Topological Sort （拓扑排序）
  - Strongly Connected Components（强连通分量）
  - Minimum Spanning Trees (最小生成树)
  - Single Source Shortest Path (单源最短路径)
  - All-Pairs Shortest Paths (所有点对最短路径)
  - Bipartite Graph Matching（二分图匹配）
  - Maximum/Network Flows (最大流/网络流)

深度优先搜索回顾：算法思想

- 算法步骤
  - 分叉时，任选一条边深入
  - 无边时，后退一步找新边
  - 找到边，从新边继续深入

- 辅助数组
  - $color$：表示顶点状态
    - $White$：白色顶点尚未被发现
    - $Black$：黑色顶点已被处理
    - $Gray$：正在处理，尚未完成
  - $pred$：顶点u由$pred[u]$发现
  - d：顶点发现时刻（变成灰色的时刻）
  - f：顶点完成时刻（变成黑色的时刻）



伪代码

- **DFS(G)**

输入: 图G
输出: 祖先数组$pred$，发现时刻d，结束时刻f
新建数组 $color[1..V], pred[1..V], d[1..V], f[1..V]$
//初始化
for $v \in V$ **do**
    $pred[v] \leftarrow NULL$
    $color[v] \leftarrow WHITE$
end
$time \leftarrow 0$
for $v \in V$ **do**
    if $color[v] = WHITE$ **then**
        DFS-Visit(G, v)
    end
end
return $pred, d, f$

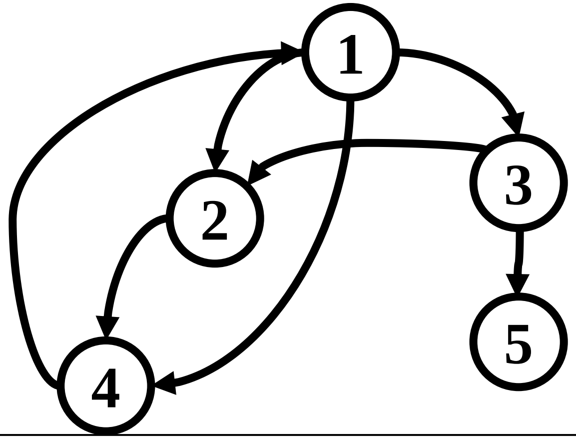
伪代码

- **DFS-Visit(G, v)**

输入: 图G, 顶点v

$color[v] \leftarrow GRAY$
$time \leftarrow time + 1$
$d[v] \leftarrow time$
for $w \in G.Adj[v]$ **do**
    if $color[w] = WHITE$ **then**
        $pred[w] \leftarrow v$
        DFS-Visit(G, w)
    end
end
$color[v] \leftarrow BLACK$
$time \leftarrow time + 1$
$f[v] \leftarrow time$

深度优先搜索回顾：有向图算法实例



$color[v] \leftarrow GRAY$
$time \leftarrow time + 1$
$d[v] \leftarrow time$
for $w \in Adj[v]$ **do**
  if $color[w] = WHITE$ **then**
    $pred[w] \leftarrow v$
    DFS-Visit(w)
  end
end
$color[v] \leftarrow BLACK$
$time \leftarrow time + 1$
$f[v] \leftarrow time$

$time = 0$

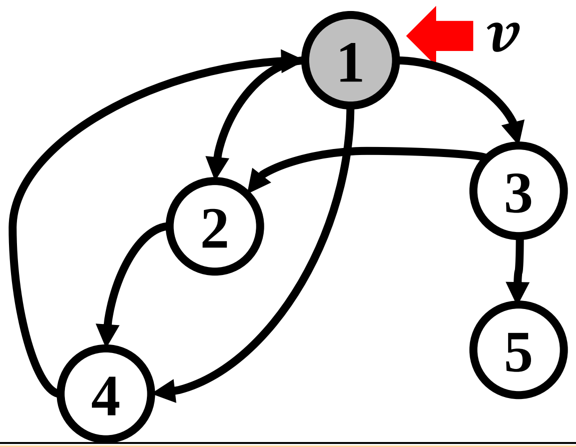
v	1	2	3	4	5
$pred$	N	N	N	N	N

v	1	2	3	4	5
$color$	W	W	W	W	W

v	1	2	3	4	5
d					

v	1	2	3	4	5
f					

深度优先搜索回顾：有向图算法实例




```
color[v] ← GRAY
time ← time + 1
d[v] ← time
for w ∈ Adj[v] do
    if color[w] = WHITE then
        pred[w] ← v
        DFS-Visit(w)
    end
end
color[v] ← BLACK
time ← time + 1
f[v] ← time
```

$time = 0$

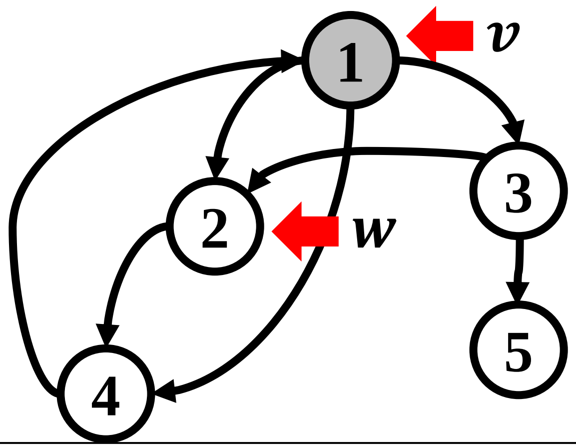
v	1	2	3	4	5
$pred$	N	N	N	N	N

v	1	2	3	4	5
$color$	G	W	W	W	W

v	1	2	3	4	5
d					

v	1	2	3	4	5
f					

深度优先搜索回顾：有向图算法实例



$color[v] \leftarrow GRAY$
$time \leftarrow time + 1$
$d[v] \leftarrow time$
for $w \in Adj[v]$ **do**
  if $color[w] = WHITE$ **then**
    $pred[w] \leftarrow v$
    DFS-Visit(w)
  end
end
$color[v] \leftarrow BLACK$
$time \leftarrow time + 1$
$f[v] \leftarrow time$

$time = 1$

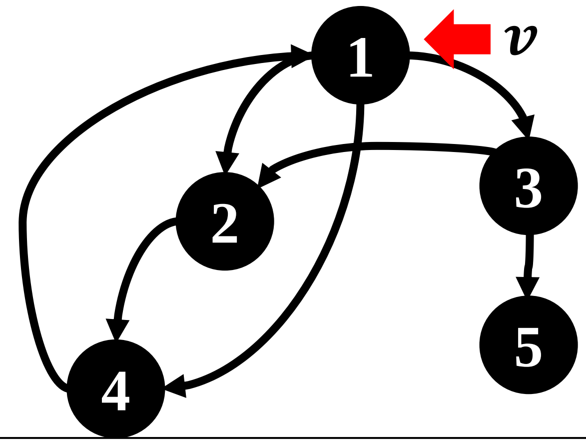
v	1	2	3	4	5
pred	N	N	N	N	N

v	1	2	3	4	5
color	G	W	W	W	W

v	1	2	3	4	5
d	1				

v	1	2	3	4	5
f					

深度优先搜索回顾：有向图算法实例



$color[v] \leftarrow GRAY$
$time \leftarrow time + 1$
$d[v] \leftarrow time$
for $w \in Adj[v]$ **do**
  if $color[w] = WHITE$ **then**
    $pred[w] \leftarrow v$
    DFS-Visit(w)
  end
end
$color[v] \leftarrow BLACK$
$time \leftarrow time + 1$
$f[v] \leftarrow time$

$time = 10$

v	1	2	3	4	5
$pred$	N	1	1	2	3

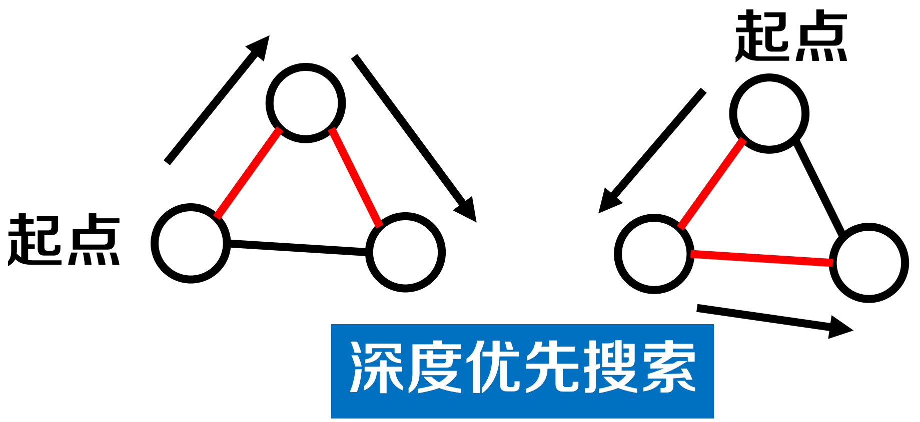
v	1	2	3	4	5
$color$	B	B	B	B	B

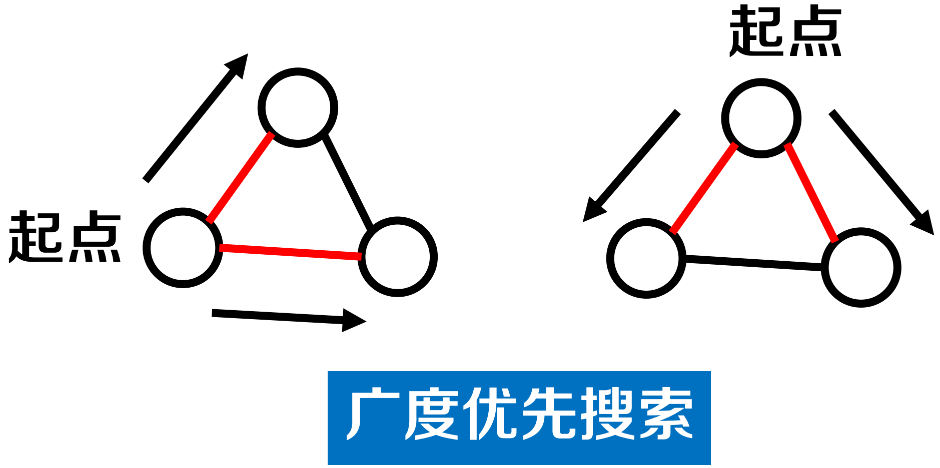
v	1	2	3	4	5
d	1	2	6	3	7

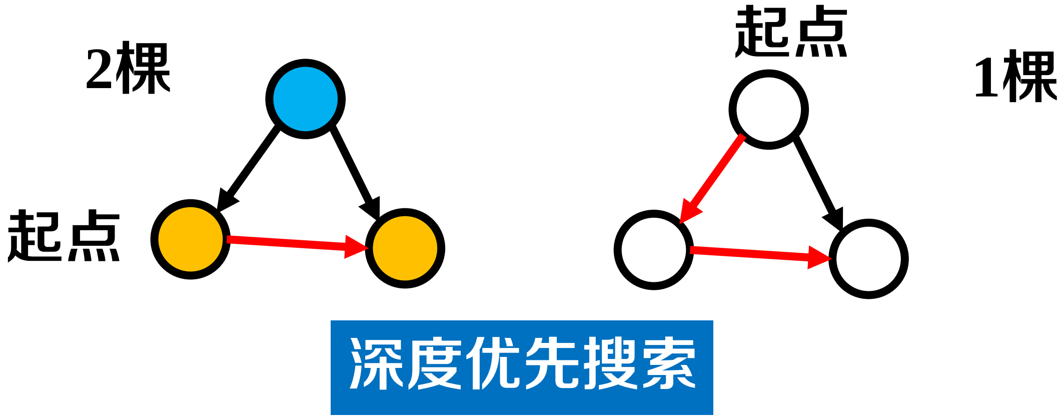
v	1	2	3	4	5
f	10	5	9	4	8

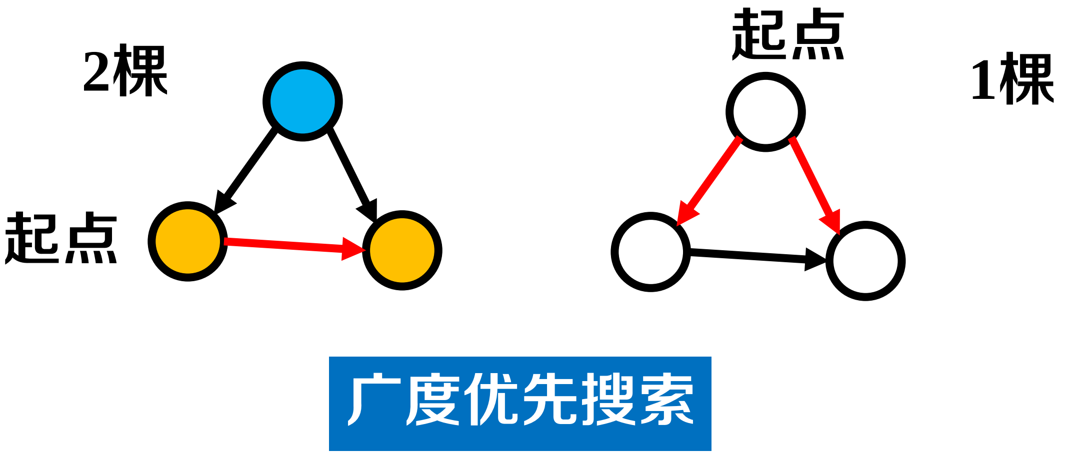
连通无向图的优先树与连通有向图的优先森林

- 无向图
  - 树的形状：取决于搜索顺序
  - 树的数量：确定1棵优先树
- 有向图
  - 树的形状：取决于搜索顺序
  - 树的数量：取决于搜索顺序

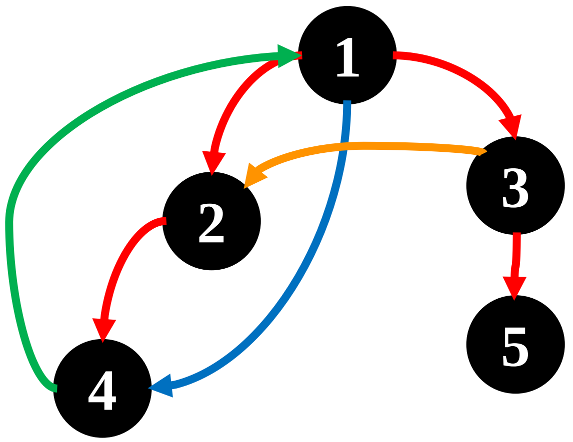








有向图深度优先森林



$time = 10$

v	1	2	3	4	5
$pred$	N	1	1	2	3

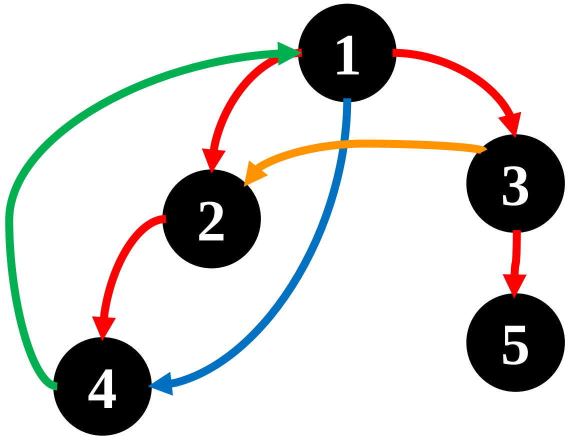
区别1：祖先指向后代？还是相反？

- 回顾深度优先搜索边的性质
  - 后向边：不是树边，但两顶点在深度优先树中是祖先后代关系
  - 对于无向图，非树边一定是后向边

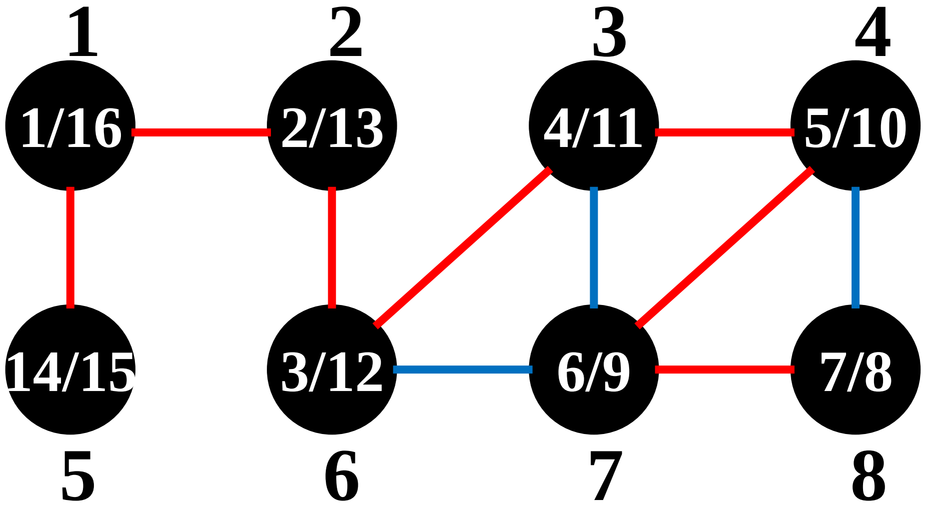
区别2：非树边出现在兄弟顶点之间

深度优先搜索边的分类

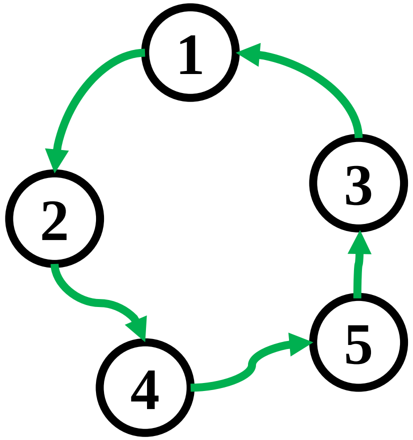
- 有向图，深度优先搜索有4类边
  - 树边：在深度优先树中的边
  - 前向边：不在深度优先树中，从祖先指向后代的边
  - 后向边：从后代指向祖先的边
  - 横向边：顶点不具有祖先后代关系的边



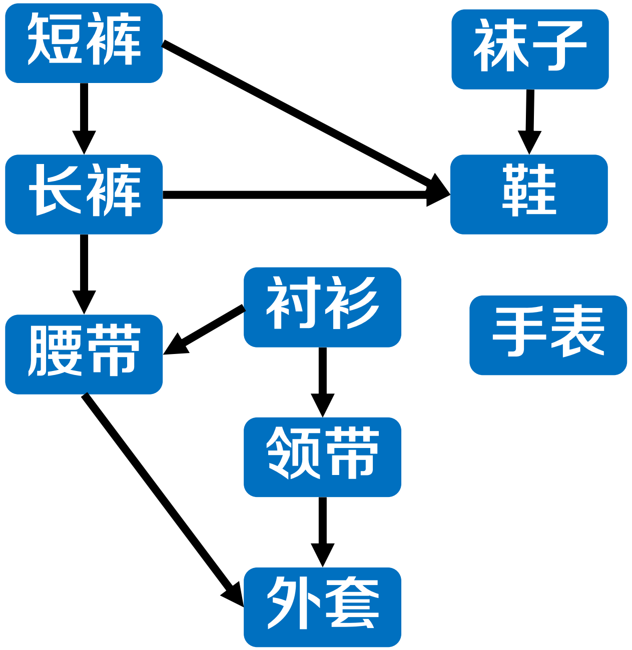
- 无向图，深度优先搜索有2类边
  - 树边：在深度优先树中的边
  - 后向边：两顶点有祖先后代关系的非树边



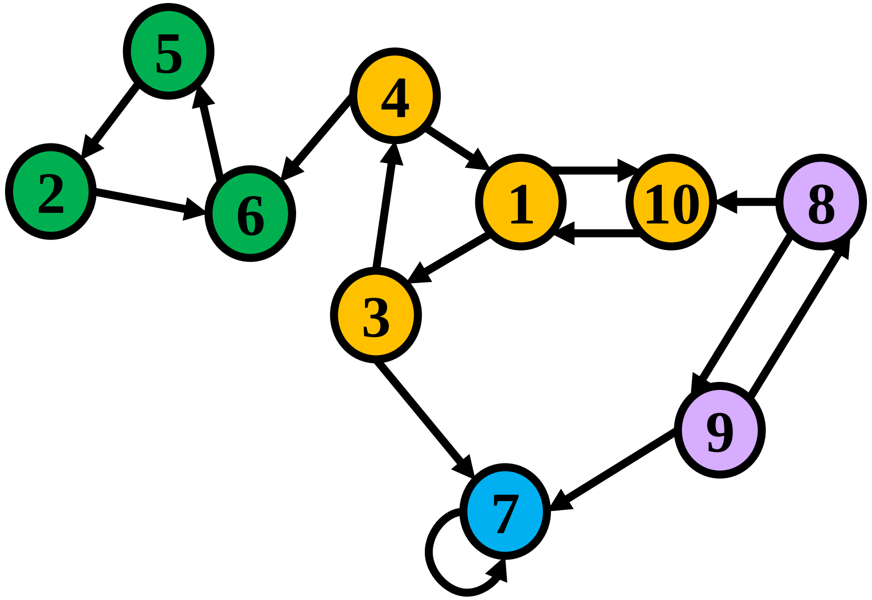
深度优先搜索应用



环路的存在性判断



拓扑排序



强连通分量

谢谢

BDA
Beihang University