算法设计与分析 Week 5

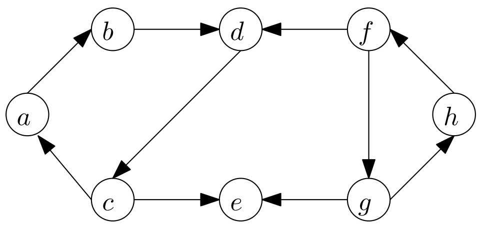
第五次作业

Lecturer: 杨启哲 Last modified: 2025 年 11 月 3 日

截止日期 2025 年 10 月 27 日晚 24：00

重要提醒： 请邮件提交的同学严格按照第一个课件上的要求，平时作业请在邮件提交一个名为"学号-姓名-算法第 5 次平时作业"的 pdf 文件作为你的作业，编程作业请在邮件提交一个名为"学号-姓名-算法第 5 次编程作业"的 pdf 文件作为你的作业，邮件标题也请写成"学号-姓名-算法第 5 次平时（编程）作业提交"。

1. 请在下图所示的有向图上应用强连通分支的算法。



解答.

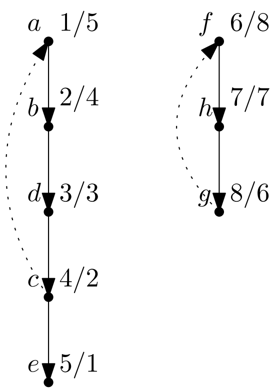
(1) 先在原图上进行 DFS，下图左侧是按字母序访问得到的 DFS 树。其 post 的逆序为：

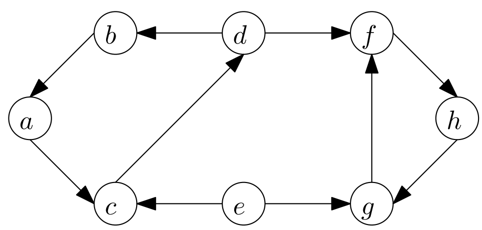
$$f,\ h,\ g,\ a,\ b,\ d,\ c,\ e$$

(2) 依照上述顺序在原图的逆图 (下图右) 中进行 DFS，每个点出发的子树即为一个强连通分支。

- 从 f 出发能访问到的点为 $\{h, g\}$，即 $\{f,\ g,\ h\}$ 是一个强连通分支。
- 去除上述点后，还剩的点为 a, b, d, c, e。从 a 出发能访问到的点为 $\{c, d, b\}$，即 $\{a, c, d, b\}$ 是一个强连通分支。
- 剩下的点为 e，即 $\{e\}$ 是一个强连通分支。

即强连通分支一共有 3 个，为 $\{f,\ g,\ h\}$, $\{a, c, d, b\}$, $\{e\}$。





$\square$

2. 给出一个线性时间的算法，找出有向图中一条路径长度为奇数的环。

   Hint: 先考虑强连通的情况。

解答. 如果是在无向图中，直接 DFS 并交替染色即可。这一方法在有向图中不适用，因为边的方向可能不一致。我们先在强连通的有向图中考虑，注意到：

> **Lemma 0.1**
>
> 在强连通的 n 个点的有向图中，一定有一条长度为 n 的圈经过所有的顶点。

因此，如果该图有奇数个顶点，那么一定有一条长度为奇数的圈。否则如果从 u 到 v 有一条偶数长的路径，则从 v 到 u 也有一条偶数长的路径。我们可以通过这个思路结合无向图中的处理方法，得到一个奇圈：

我们可以从任一点出发对图进行 DFS，并交替染黑和白色。不妨令初始点为 s，并且标记为白色，则：

- 被标记为黑色的点意味着从初始点出发有一条奇数长的路径。
- 被标记为白色的点意味着从初始点出发有一条偶数长的路径。

我们宣称，如果当前顶点访问到一个同色的顶点，则存在一个奇数圈。不妨假设这两个点为 u, v，则构造圈如下：

$$(s \to^* u) + (u \to v) + (v \to^* s)$$

其中 $s \to^* t$ 表示 s 到 t 的一条路径。注意到 u, v 都是同色的，因此有前面的讨论，存在一条都是奇数或者都是偶数的 s 到 u 和 v 到 s 的路径，因此上述圈是奇数的。同时我们注意到，如果图中存在一个奇圈，无论从哪个点开始 DFS 交替染色，一定会访问到相同染色的顶点。因此上述过程一定能在有奇圈的强连通有向图中找出一个奇圈。

现在考虑一般的情况，这只需注意到奇圈一定出现在有向图中的某个强连通分量上，因此我们只需利用强连通分量的算法求出其所有的强连通分量，再分别在强连通分量上执行上述算法即可。算法流程如下：

(1) 执行强连通分量算法得到其所有的强连通分量。

(2) 对每个强连通分量，若其只有奇数个点，则返回 Yes。

(3) 对其进行 DFS 并交替染色，若访问到同色的顶点，返回 Yes.

(4) 返回 No.

注意到上述每个过程都至多是线性的，因而这是一个线性时间算法。 $\square$

Remark 0.2

(1) 事实上，上述过程也给出了具体的环的求法。

(2) 在考虑有向图上的问题中，先将图简化为强连通分量考虑是个常用手段。

3. (DAG 中的路径) 请给出一个算法计算给定一个有向无环图中所有的路径数量，并分析你的算法。

解答. 基本思路是先将 DAG 输出其拓扑序列，假设其拓扑序为 $v_1, \ldots, v_k$，定义 $P(i,j)$ 为 v_i 到 v_j 的路径数量，则我们有：

$$P(i,j) = \begin{cases} 1 & i \geqslant j \\ \sum_{v_k \in G.adj[v_i]} P(k,j) & i < j \end{cases}$$

全部加起来即可获得图中所有路径的数量。算法流程如下：

(1) 计算 DAG 的拓扑序列 $v_1, \ldots, v_k$（对图进行 DFS 再根据 post 序逆序输出即可）

(2) 对每个点 v_i，根据上述公式依次计算 $P(i,j)$，其中 $j \in \{i, \ldots, k\}$。

(3) 输出所有的 $P(i,j)$ 的和。

第一步的时间复杂度为 $O(|V|+|E|)$，第二步的时间复杂度为 $O(|V| \cdot |E|)$，第三步的时间复杂度为 $O(|V|^2)$，因此总的时间复杂度为 $O(|V| \cdot |E|)$. □

4. 给定一个有向图 $G = (V, E)$，其反转 $G^R = (V^R, E^R)$ 是将 G 中所有边的方向反转得到的图，即满足：

$$V^R = V, \quad E^R = \{(u,v) | (v,u) \in E\}$$

请给出一个线性时间算法，在 G 用邻接表表示的情况下，给出 G^R 的邻接表表示。

解答. 算法的核心思路是遍历 G 的邻接表，对于每一条边 (u,v)，在 G^R 的邻接表中加入一条边 (v,u)。算法流程如下：

构造反转图的邻接表

输入： 有向图 G 的邻接表表示
输出： 有向图 G^R 的邻接表表示

初始化 G^R 的邻接表为空
for 每个顶点 $u \in V$ **do**
    for 每个顶点 $v \in G.adj[u]$ **do**
        在 $G^R.adj[v]$ 中加入顶点 u
    end for
end for
return G^R 的邻接表

注意到上述算法中每条边被访问一次，因此时间复杂度为 $O(|V|+|E|)$，即线性时间。 □

5. 给出一个线性时间算法，判断在一个有向无环图中，是否存在一条路径，恰好访问每个顶点一次。

Remark 0.3

这其实就是哈密顿路径问题在有向无环图中的特例。哈密顿路径问题是 NP 完全问题，但在有向无环图中，我们可以在线性时间内解决它。

解答. 核心思路还是依旧拓扑序进行回答，事实上假设 $v_1, \ldots v_n$ 是图的一个拓扑序列，则我们有：

Lemma 0.4

图中存在一条恰好访问每个顶点一次的路径，当且仅当对于任意的 $i \in \{1, \ldots, n-1\}$，都有 $(v_i, v_{i+1}) \in E$。

引理证明： 充分性是显然的，下证必要性。反设存在一条恰好访问每个顶点一次的路径 P，但存在某个 i 使得 $(v_i, v_{i+1}) \notin E$。现在考虑 v_i 和 v_{i+1} 在路径 P 的位置，有如下两种情况：

(1) v_{i+1} 在 v_i 之前，不妨令为：$v_{i+1}, v_{i_1}, \ldots, v_{i_k}, v_i$，则由拓扑序的定义，$v_{i+1}$ 必然在 v_i 之前，与上述序列矛盾。

(2) v_{i+1} 在 v_i 之后，注意到 $(v_i, v_i + 1) \notin E$，从而必然存在 $v_j \in V$ 在 v_i 和 v_{i+1} 之间，此时在拓扑序里，v_j 必然在 v_i 之后 v_{i+1} 之前，与给定的序列矛盾。

□

从而我们可以设计如下算法：

判断有向无环图中是否存在哈密顿路径

输入： 有向无环图 G
输出： G 中是否存在一条恰好访问每个顶点一次的路径

```
1: 计算 G 的一个拓扑序列 v_1, ..., v_n
2: for i = 1 to n − 1 do
3:     if (v_i, v_{i+1}) ∉ E then
4:         return false
5:     end if
6: end for
7: return true
```

注意到计算拓扑序列的时间加上检查边的时间都是线性的，所以该算法是一个线行时间的算法，即复杂度为 $O(|V| + |E|)$. □