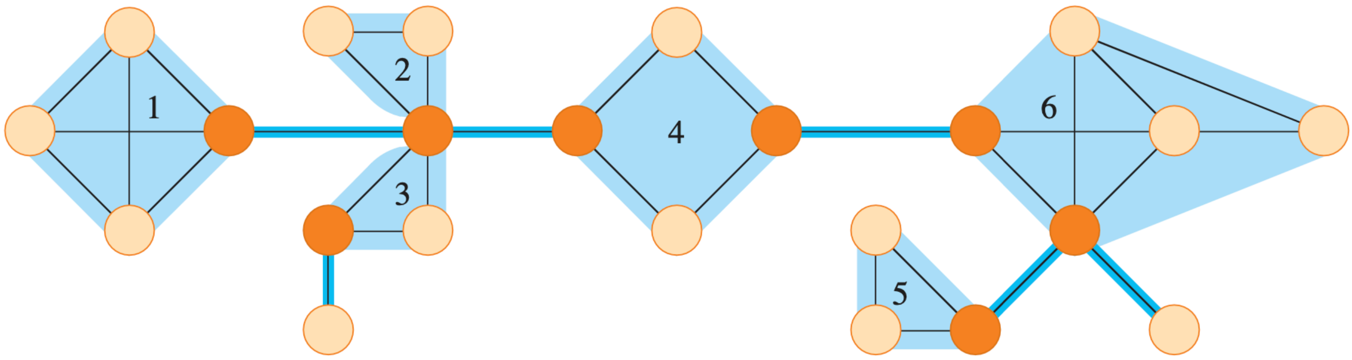
A7参考题解

1 CLRS 20-2



在图论的证明中会反复地用到反证法，这一点大家不难从课上讲的各种证明看出。

1. Prove that the root of G_π is an articulation point of G if and only if it has at least two children in G_π.

   我们记G_π的根为r。注意，G和G_π的区别在于G_π为有向图，且G_π中每一条边都带有标记(tree edge和back edge，没有另外两种edge是因为G为连通无向图)：

   (a) $\Rightarrow$
   我们使用反证法，已知r是articulation point，且G_π中r只有一个children（记为c）。此时，我们把G和G_π中的r去掉，分别得到图G'和G'_π。由于G'_π是一棵树，而把G'_π中所有边都变成无向边就能得到G'，故G'为连通图，这与r是articulation point矛盾。

   (b) $\Leftarrow$
   已知G_π中的r有多个children。此时，我们把G和G_π中的r去掉，分别得到图G'和G'_π。G_π中的r有多个children，故G'_π为多棵树的森林。同样地，由于G'_π是一棵树，而把G'_π中所有边都变成无向边就能得到G'。故只要G'_π中不存在cross edge，那么G'为非连通图。而由课上所学知识，我们知道无向连通图中没有cross edge。故r是articulation point。

2. Let v be a nonroot vertex of G_π. Prove that v is an articulation point of G if and only if v has a child s such that there is no back edge from s or any descendant of s to a proper ancestor of v.

   在第一题讨论的基础上证明这一题很简单，我们只需要利用好G和G_π的关系即可。

3. Let

$$v.low = \min \begin{cases} v.d, \\ w.d : (u,w) \text{ is a back edge for some descendant } u \text{ of } v \end{cases}$$

   Show how to compute $v.low$ for all vertices $v \in V$ in $O(E)$ time.

   我们可以利用一个递归算法来计算：

$$\text{computeLow}(v) = \min\{w.d : (u,w) \text{ is a back edge for some descendant } u \text{ of } v\}$$

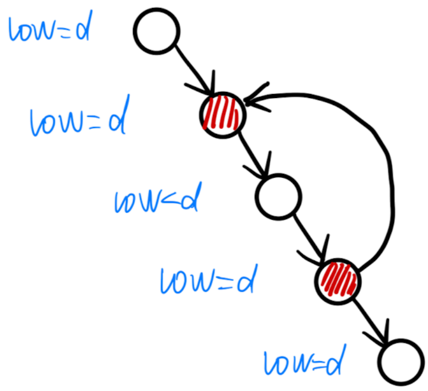
   算法如下：

$$
\begin{aligned}
&\text{computeLow}(v):\\
&\quad tmp := \text{MAX}\\
&\quad \text{for } u \text{ such that } u \text{ is } v\text{'s neighbor}\\
&\quad\quad tmp := \min(tmp, \text{computeLow}(u))\\
&\quad\quad \text{for all back edges } uw \text{ of } u\\
&\quad\quad\quad tmp := \min(tmp, w.d)\\
&\quad v.low = \min(v.d, tmp)\\
&\quad \text{return } tmp
\end{aligned}
$$

然后我们直接调用computeLow(r)，其中r为深度优先树的根，便可得到所有节点v的low值。算法复杂度的上界为$O(V+E)=O(E)$（因为G为无向连通图，故$V \le E+1$）。

4. Show how to compute all articulation points in $O(E)$ time.

我们可以通过画图来直观感受articulation point的特点：



红色节点为articulation point

我们用第三题的算法计算出每个节点的low值，然后我们对所有的节点v进行判断：

$$
\begin{aligned}
&\text{computeArticulation}(G_\pi):\\
&\quad V := \varnothing \qquad {\color{red} V \text{ is the set of articulation points}}\\
&\quad \text{computeLow}(r) \qquad {\color{red} r \text{ is the root of } G_\pi}\\
&\quad \text{for } v \in V \setminus \{r\}\\
&\quad\quad \text{if } v.low = v.d \text{ and } v \text{ is not a leaf node}:\\
&\quad\quad V := V \cup \{v\}\\
&\quad \text{return } V
\end{aligned}
$$

从而得到所有的articulation point。

原理很简单，如果$v.low < v.d$，说明存在u为v的一个descendant，使得uw为back edge且w为v的proper ancestor。而如果$v.low = v.d$，说明不存在u为v的一个descendant，使得uw为back edge且w为v的proper ancestor。由第二题结论知$v.low = v.d$且v不是叶子结点（即v存在children）时v为articulation point。

这道题比较容易犯错的地方在于把v需要有descendant和v不是根结点的条件忽略掉，例如：

4. 方法：首先用3中的方法对所有$v \in V$，计算出$v.low$，然后比较$v.low$与$v.d$，若$v.low = v.d$，那么说明v及其后代不存在通向v的祖先的回边，由2中的证明可知这种点v是衔接点，总计算时间为$O(E)$。

某位同学的解答

5. Prove that an edge of G is a bridge if and only if it does not lie on any simple cycle of G.

   (a) $\Rightarrow$

   已知uv为bridge，那么$G' = (V, E \setminus uv)$不连通，且有两个连通分量$G_1(V_1, E_1), G_2(V_2, E_2)$。我们不是一般性地假设$u \in V_1, v \in V_2$。

   假如uv刚好在G中的某个simple cycle，那么存在一条从u到v的路径不经过uv，故该条路径仍存在于G'中。此时在G'中，对于任意$s \in V_1, t \in V_2$，s到u存在路径，u到v存在路径，v到t存在路径，故s到t存在路径。这与G'不连通矛盾。故uv不在在G中的任意simple cycle。

   (b) $\Leftarrow$

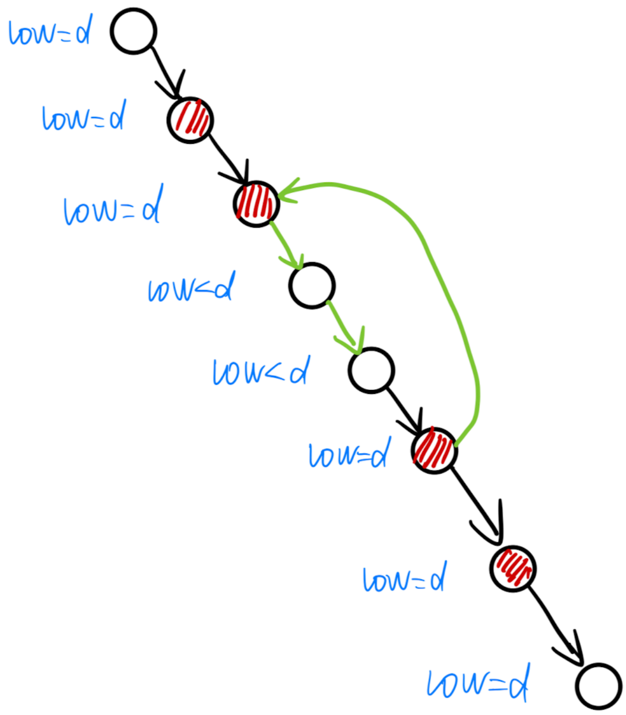
   已知uv不在在G中的任意simple cycle，那意味着u到v到唯一路径是uv。不是一般性地假设u还有邻居$w, w \neq v$。我们证明，在$G' = (V, E \setminus uv)$中，w到v没有路径。

   我们使用反证法，假如w到v有路径，那么由于u是w的邻居，那么u到v也有路径，这与uv不在在G中的任意simple cycle矛盾。故$G' = (V, E \setminus uv)$中，w到v没有路径，即G'不连通，uv为bridge。

6. Show how to compute all the bridges of G in $O(E)$ time.

   这里需要利用到一个观察：如果一条边(u, v)处于某个cycle上，那么满足以下三个条件之一：

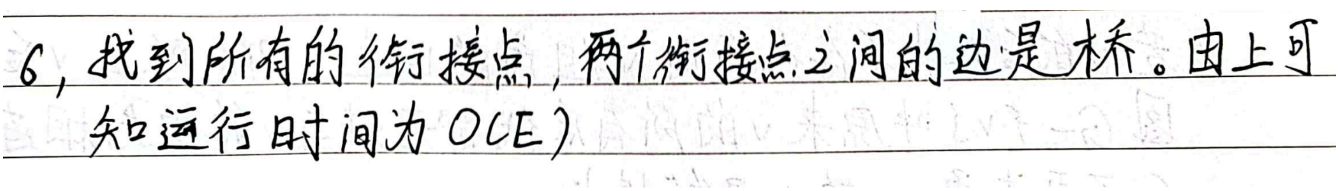
   - $u.low < u.d$ 或者 $v.low < v.d$
   - $u.d > v.d$，即(u, v)为back edge



红色节点为articulation point，绿色边为simple cycle上的边

我们只需要对每一条边，判断其是否在simple cycle上即可。如果不在simple cycle上，那么由第5题结论可得该边为bridge。

有同学误以为任意两个articulation point之间的边均为bridge：


6，找到所有的衔接点，两个衔接点之间的边是桥。由上可知运行时间为O(E)

某位同学的解答

7. Prove that the biconnected components of G partition the nonbridge edges of G.

我们记

- $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$：$G$的biconnected components的集合
- $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$：$G$的nonbridge edges的集合

我们要证明对于任意$e \in \mathcal{E}$，存在唯一的$i \in \{1, 2, \ldots, k\}$使得$e$为$C_i$中的边。

- 唯一性

  我们只需要证明，任意两个biconnected components没有共同边。我们使用反证法，不失一般性地假设C_1和C_2之间存在一条共同边uv。我们从C_1中任选一个不同于u的点w，从C_2中任选一个不同于v的点t，我们知道w到u存在至少两条路径（记为p_1, p_2），v到t也存在至少两条路径（记为p_3, p_4），那么从w到t也至少存在四条路径：

- p_1, uv, p_3
- p_2, uv, p_3
- p_1, uv, p_4
- p_1, uv, p_4

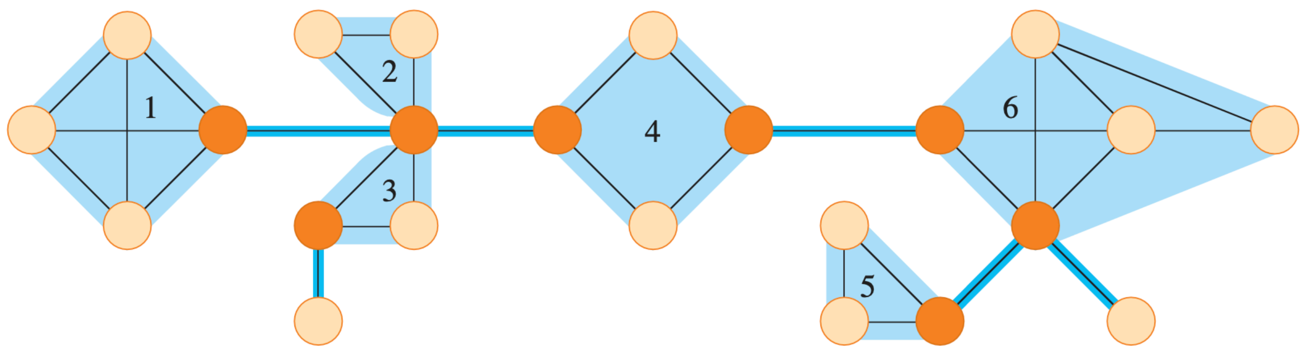
故C_1和C_2构成了更大的biconnected components，这与biconnected components定义中的maximal是矛盾的。故任意两个biconnected components没有共同边。

- 存在性

由于e为nonbridge edge，故e存在于某个simple cycle中。由biconnected components的定义知，e一定存在于某个biconnected components中。

8. Give an $O(E)$-time algorithm to label each edge e of G with a positive integer $e.bcc$ such that $e.bcc = e'.bcc$ if and only if e and e' belong to the same biconnected component.

观察题目给出的这个图，计算biconnected component可以先把所有的bridge去掉，然后对整个图做DFS或者BFS即可。



算法如下：

$$
\begin{aligned}
&\text{computeBCC}(G):\\
&\quad index := 0\\
&\quad \text{for } v \in V\\
&\quad\quad v.visit := \textbf{false}\\
&\quad \text{for } e \text{ that is bridge of } G\\
&\quad\quad E := E \setminus \{e\}\\
&\quad \text{for } v \in V\\
&\quad\quad \text{if } v.visit = \textbf{false}\\
&\quad\quad\quad \text{dfs}(v, index)\\
&\quad\quad\quad index := index + 1
\end{aligned}
$$

$$
\begin{aligned}
&\text{dfs}(v, index):\\
&\quad v.visit := \textbf{true}\\
&\quad \text{for } u \text{ that is } v\text{'s neighbor}\\
&\quad\quad vu.bcc = index\\
&\quad\quad \text{if } u.visit = \textbf{fasle}\\
&\quad\quad\quad \text{dfs}(u, index)
\end{aligned}
$$

2 Analyze topological sort

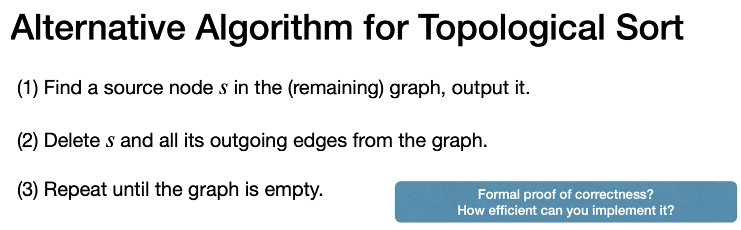


Figure 2 The alternative algorithm for topological sort in slide 14 *Application of DFS*

- 正确性证明:

  我们使用反证法。假设该算法得到的拓扑排序序列为

  $$v_1, v_2, \ldots, v_n$$

  如果该拓扑排序序列有误，那么存在$i < j$使得$v_j v_i \in E$，我们假设这一点成立。那么当我们在选择并删除v_i的时候，v_j和$v_j v_i$都存在，此时v_i并不是source，这与算法矛盾。故该拓扑排序算法正确。

- 复杂度分析:

  这里仅分析平凡的复杂度上界，而不去分析该算法可以通过优化达到的最优复杂度上界。算法第i次（从1开始计起）迭代时，图中仍有$|V| + 1 - i$个节点，此时找到一个source的最坏需要$|V| + 1 - i$次，即遍历所有节点。算法需要做n次迭代，故这部分复杂度上界为:

  $$\sum_{i=1}^{n} |V| + 1 - i = O(|V|^2)$$

  而删除部分总共删除$|V|$个节点和$|E|$条边，故这部分复杂度上界为$O(|V| + |E|)$。

  两者加起来得到算法的复杂度上界:

  $$O(|V|^2 + |E|)$$

3 *Implement Prim's Algorithm with Fibonacci Heap

这道题其实就是老师想让大家自学一下Fibonacci Heap，故么得答案