22.4-2 拓扑排序计算有向无环图简单路径的条数

请给出一个线性时间的算法，算法的输入为一个有向无环图 $G=(V,E)$ 以及两个结点 s 和 t，算法的输出是从结点 s 到结点 t 之间的简单路径的数量。例如，对于图 22-8 所示的有向无环图，从结点 p 到结点 v 一共有 4 条简单路径，分别是 pov、$poryv$、$posryv$ 和 $psryv$。（本题仅要求计数简单路径的条数，而不要求将简单路径本身列举出来。）

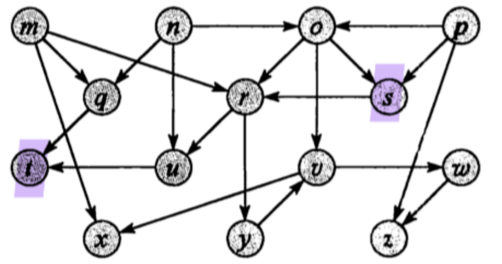

图 22-8 一个用于拓扑排序的有向无环图

算法的实现思路为：

令u.path_count属性表示u结点到v节点简单路径的条数

- 如果u结点和v结点相同，则只有一条路径；
- 如果u结点和v结点不相同，并且u.path_count=NIL(这表示递归还未结束，递归结束时，这个值应该是一个Integer，而不是一个NIL)时，遍历u的邻接链表中的所有结点，递归调用SIMPLE-PATH-NUMBER函数，累加path_count，因为每一条从u到v的路径一定会经过u的邻接结点；
- 如果u结点和v结点不相同并且u.path_count!=NIL,表示递归已经调用结束，路径条数已经求出，可以return。
- 该算法的时间复杂度是$O(V+E)$，是线性时间算法。

```
SIMPLE-PATH-NUMBER(u,v)
    if(u==v)
        return 1
    else if(u.path_count!=NIL)
        return u.path_count
    else
        for each w ∈ Adj[u] do
            u.path_count=u.path_count+SIMPLE-PATH-NUMBER(w,v)
        return u.path_count
```

22.4-5 寻找入度为0的结点的方法实现拓扑排序

在有向无环图 $G=(V,E)$ 上执行拓扑排序还有一种办法，就是重复寻找入度为 0 的结点，输出该结点，将该结点及从其发出的边从图中删除。请解释如何在 $O(V+E)$ 的时间内实现这种思想。如果图 G 包含环路，将会发生什么情况？

找出入度为0的节点，删除该点并删除该点所有的出边，删除的顺序就是拓扑排序的顺序

实现思想：

1. 遍历所有邻接链表，统计每个结点的入度，时间复杂度为O(E)
2. 找出一个入度为0的结点u，将u的所有出边(u,v)的目的点v的入读减一，时间复杂度为O(V+E)
3. 循环步骤1

如果**G**包含环路：

1. 若G中包含环路，则会找不到入度为0的结点，且G中仍然有结点不能删除（剩下的就是G的回路）。

23.1-11

***23.1-11　给定图 G 和一棵最小生成树 T，假设减小了位于 T 之外的某条边的权重。请给出一个在修改后的图中寻找最小生成树的算法。**

Consider edge E is the edge whose weight is decreased by one unit.

createUpdatedMST(MST T, edge E)

Step 1 : Sort the edges in the MST T in decreasing order by weight.

Step 2 : Select the **maximum weight edge** from the above sorted list which is not yet selected. Let this edge be **e.**

Step 3 : If weight of edge **E** is more than weight of edge **e**, exit the algorithm as **given MST T is the actual MST of the graph.**(不做任何改变)

Step 4 : Else if weight of edge E is less than weight of edge e selected from the list, check if adding **E** to the MST and removing **e** from MST results in a cycle.

Step 5 : If a cycle is not formed return the new updated MST with edge **E.**

Step 6 : If the cycle is formed, go to step 2 and check the next highest weight edge from the sorted list.

23.2-7

*23.2-7 假定图 G 的一棵最小生成树已经被计算出来。如果在图中加入一个新结点及其相关的新边，我们需要多少时间来对最小生成树进行更新？

线性时间

1. The greedy method works on the basis of this selection policy: **choose the minimum-weight remaining edge.** If that edge does not create a cycle in the evolving tree, add it to the tree.

2. For finding and deleting the min-weight edge, use a minheap where its nodes are the labels+weights of the graph edges.

3. For **cycle detection**, note that

- T is a forest at any given time,
- adding an edge eliminates two trees from the forest and replaces them by a new tree containg the union of the nodes of the two old trees, and
- and edge $e = (x, y)$ creates a cycle if both x and y belong to the same tree in the forest.

Complexity:

- $O(|E|)$ to build the heap
- up to |E| calls to U and F, taking $O(|E|logn)$ time

therefore, the total time is $O(|E|log|E|)$.