数据结构和算法
(Python描述)

郭 炜

微信公众号

微博：http://weibo.com/guoweiofpku

学会程序和算法，走遍天下都不怕!

讲义照片均为郭炜拍摄

拓扑排序和关键路径

拓扑排序

韩国济州岛火山口

AOV网络

- Activity on Vertex Network

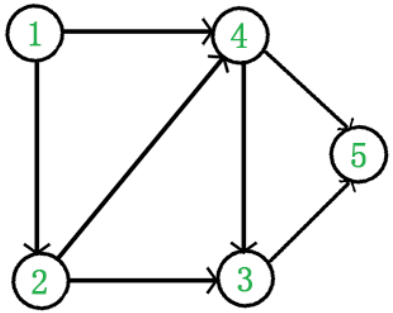
- 将有向图中的顶点看作活动，边看作活动的先后关系，A->B就意味着B活动进行前必须先进行A活动，则有向图可以看作是AOV网络

拓扑排序的概念

- 拓扑排序（Topological Sorting）：在有向图中求一个顶点的序列，使其满足以下条件:

1）每个顶点出现且只出现一次

2）若存在一条从顶点 A 到顶点 B 的路径，那么在序列中顶点 A 出现在顶点 B 的前面



拓扑序列1,2,4,3,5

- 有向图或AOV网存在拓扑排序的充要条件：
  图中无环，即图是有向无环图DAG
  (Directed Acyclic Graph)

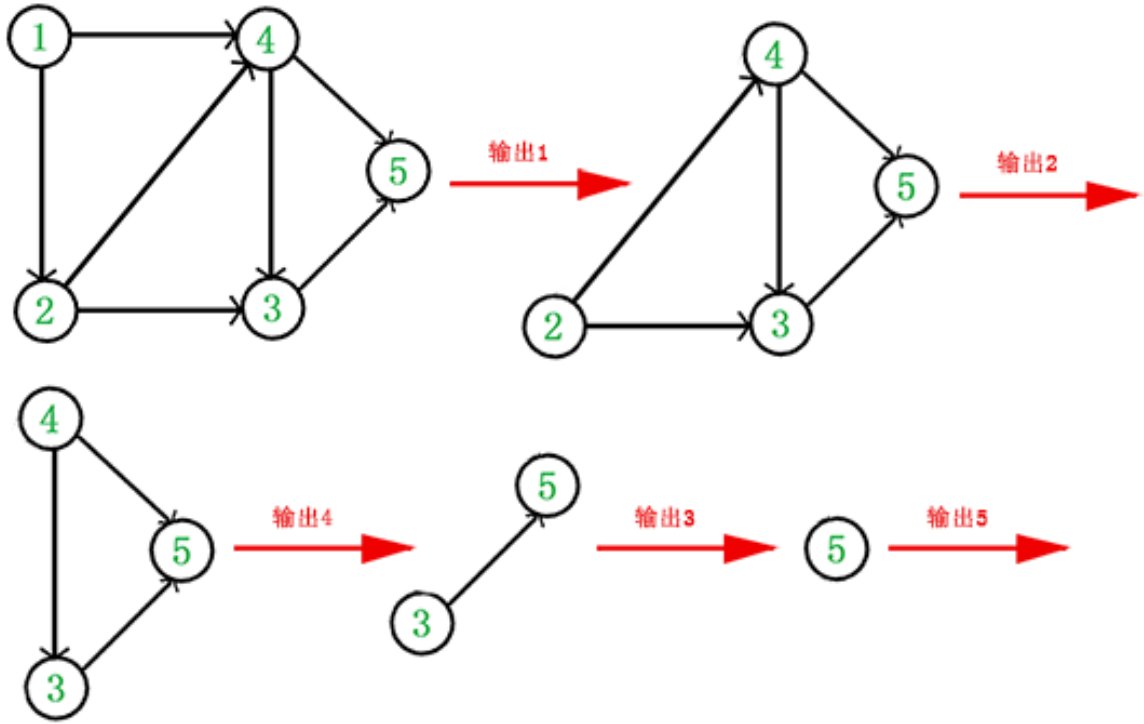
拓扑排序算法

1. 从图中任选一个没有前驱（入度为0）的顶点 x 输出

2. 从图中删除 x 和所有以它为起点的边

重复 1 和 2 直到图为空或当前图中不存在无前驱的顶点为止(后一种情况说明图中有环，无法拓扑排序)

具体实现：用队列存放入度变为0的点。每个顶点出入队列一次，每个顶点连的边都要看一次，复杂度O(E+V)



例题：Genealogical tree

给一个有向无环图，输出任一拓扑排序

样例输入
```
5          #5个点
0          #1号点没出边
4 5 1 0 #2号点有边连到 4,5, 1
1 0
5 3 0
3 0
```

样例输出
```
2 4 5 3 1
```

```
class Edge:
        def __init__(self,v,w):
                self.v,self.w = v,w   #v是顶点，w是权值
def topoSort(G): #G是邻接表，顶点从0开始编号
        #G[i][j]是Edge对象
        n = len(G)
        import queue
        inDegree = [0] * n #inDegree[i]是顶点i的入度
        q = queue.Queue()
        for i in range(n):
                for e in G[i]:
                        inDegree[e.v] += 1
        for i in range(n):
                if inDegree[i] == 0:
                        q.put(i)
        seq = []
```

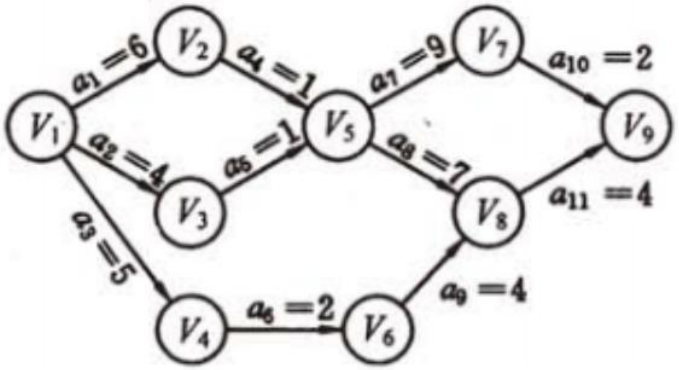
```
while not q.empty():
        k = q.get()
        seq.append(k)
        for e in G[k]:
                inDegree[e.v] -= 1   #删除边(k,v)后将v入度减1
                if inDegree[e.v] == 0:
                        q.put(e.v)
if len(seq) != n:   #如果拓扑序列长度少于点数，则说明有环
        return None
else:
        return seq
```

AOE网络

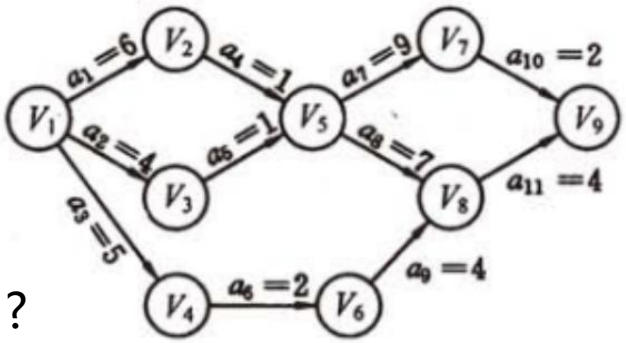
内蒙古阿斯哈图石林

AOE网络(Activity On Edge network)

- 带权有向无环图
- 顶点表示事件，事件不需要花时间
- 有向边表示活动，边权值表示活动需要花的时间
- 先后顺序无关的活动可以同时进行
- 当且仅当一个顶点的入边代表的活动都已经完成，该顶点表示的事件会发生。顶点代表的事件一旦发生，其出边代表的活动就都可以(不是必须)开始



AOE网络



- AOE网络上的问题

1） 每个事件最早可以在什么时刻发生 (时刻从0开始算）？

2） 所有活动都完成，最早是什么时刻？记最早时刻为T

T = max{ 事件i的最早发生时刻 | i = 1,2,3.....}

3） 每个事件最晚必须什么时候发生，才不会导致 T 增加?

活动的最早开始时刻 = 起点事件的最早发生时刻

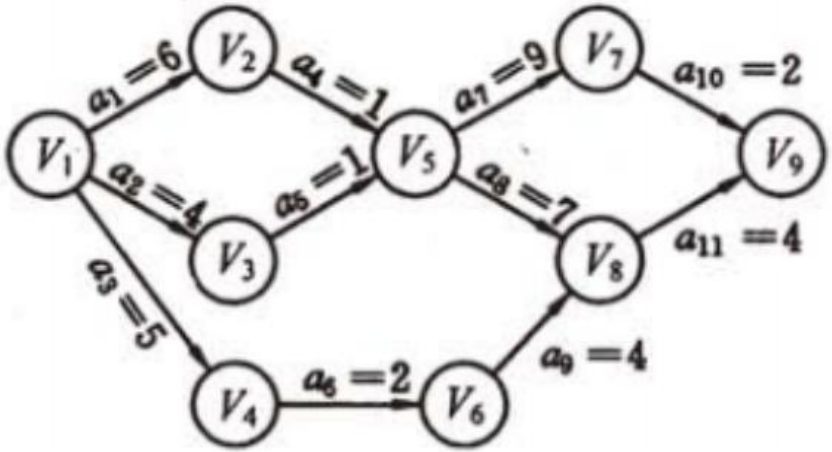
活动的最晚开始时刻= 终点事件的最晚发生时刻 – 活动时长

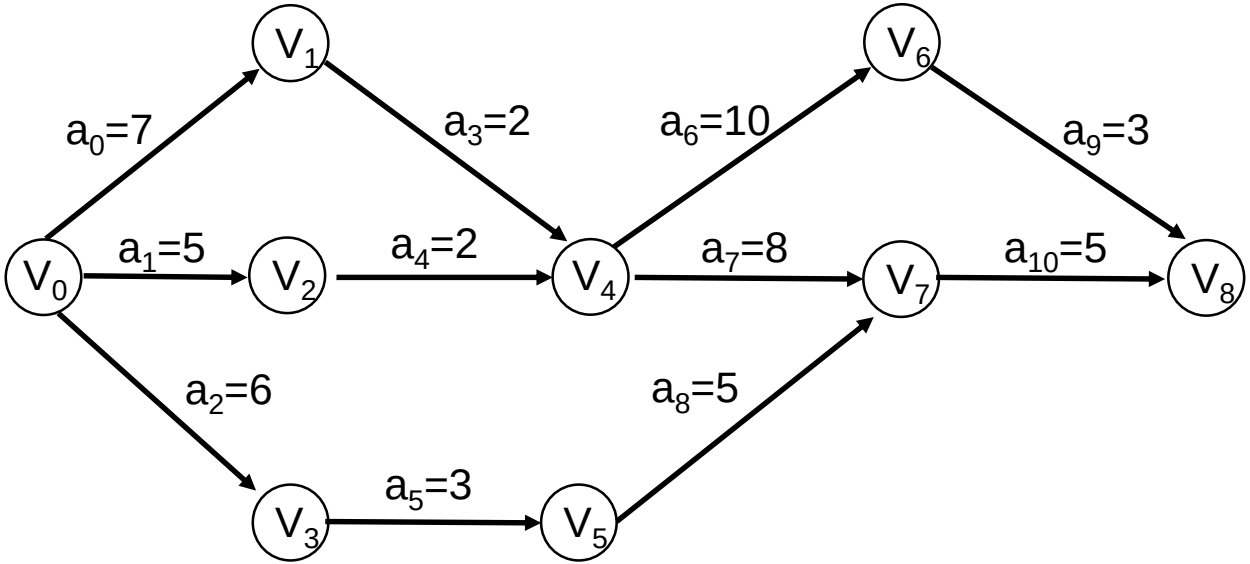
4） 求关键活动：求最早开始时刻和最晚开始时刻一致的活动

5） 求关键路径：AOE网络上权值之和最大的路径(不唯一）。关键路径一定由关键活动构成。T 就是关键路径权值之和

AOE网络

AOE网络并非只有一个入度为0的点，也并非只有一个出度为0的点。也可以为所有入度为0的点添加一个前驱点，为所有出度为0的点都添加一个后继点，边权都为0，则AOE网络变成起点终点唯一



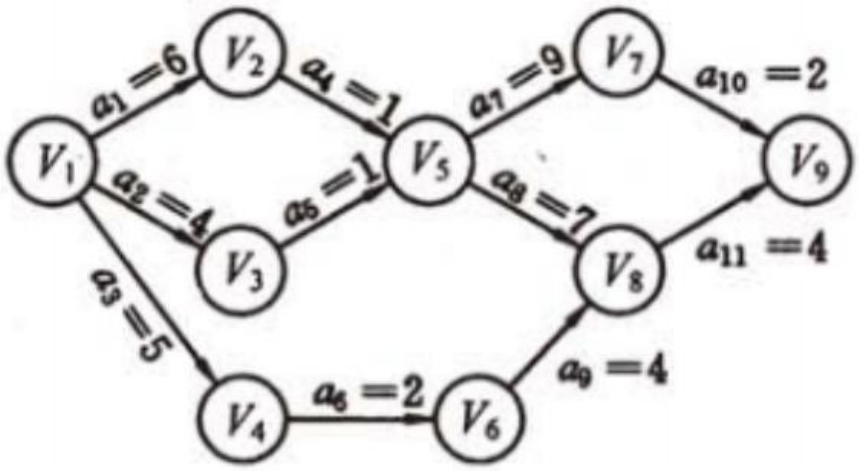

V1
V6
a0=7
a3=2
a6=10
a9=3
a1=5
a4=2
a7=8
a10=5
V0
V2
V4
V7
V8
a2=6
a8=5
a5=3
V3
V5

AOE网络

- AOE网络上的问题

1) 所有活动都完成的时刻T = 18

事件	1	2	3	4	5	6	7	8	9
最早时刻	0	6	4	5	7	7	16	14	18
最晚时刻	0	6	6	8	7	10	16	14	18



2) 关键活动：a1,a4,a7,a10,a8,a11

3) 关键路径：

a1->a4->a7->a10

a1->a4->a8->a11

AOE网络算法

- 关键是求每个事件 i 的最早发生时间 earliestTime[i] 和最晚发生时间 latestTime[i]

AOE网络算法

➢ 递推求earliestTime[i]

1) 初始条件：对每个入度为0的顶点k(事件k)，
   earlistTime[k] = 0
2) 拓扑排序
3) 按拓扑序列的顺序递推每个事件的最早开始时间：
   对拓扑序列中的顶点 i,若边<i, j>存在且权值为W_{ij} ， 则:

earliestTime[j] = max(earliestTime[j], earliestTime[i] + W_{ij})

依据：一个活动起点事件为i，终点事件为j，活动花时间W_{ij}，则:
earliestTime[j] >= earliestTime[i] + W_{ij}

AOE网络算法

- 递推求latestTime[i]

1) 求出全部活动都完成的最早时刻 T
2) 初始条件：对每个出度为0的顶点k(事件k)，
   latestTime [k] = T
2) 拓扑排序
3) 按拓扑序列的逆序递推每个事件的最晚开始时间：
   对拓扑逆序列中的顶点 j,若边<i , j>存在且权值为W_{ij} ， 则:

latestTime[i] = min(latestTime[i], latestTime[j] - W_{ij})

依据：一个活动起点事件为i，终点事件为j，活动花时间W_{ij}，则:
latestTime[i] <= latestTime[j] - W_{ij}