



数据结构与算法 (Python版)

图的应用：最小生成树

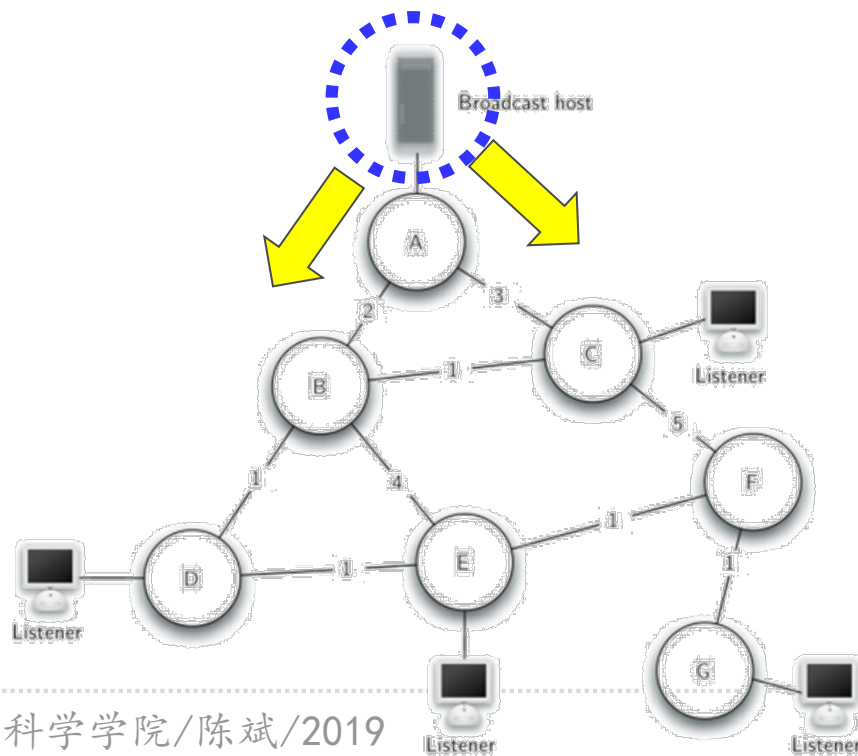
陈斌 北京大学 gischen@pku.edu.cn

最小生成树

❖ 本算法涉及到在互联网中网游设计者和网络收音机所面临的问题：信息广播问题

网游需要让所有玩家获知其他玩家所在的位置

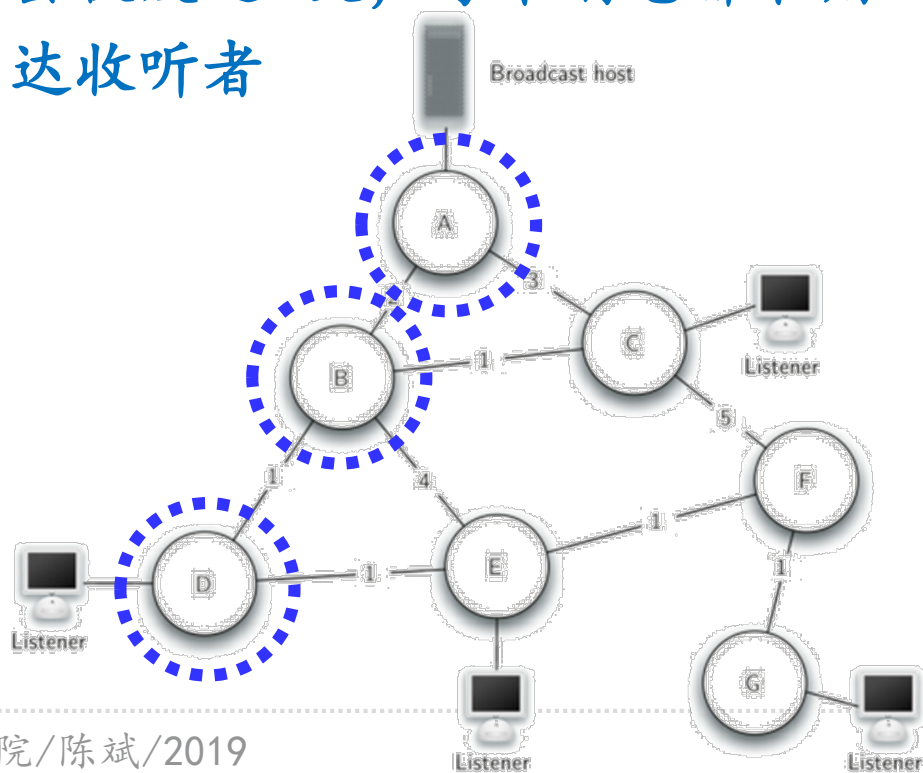
收音机则需要让所有听众获取直播的音频数据



信息广播问题：单播解法

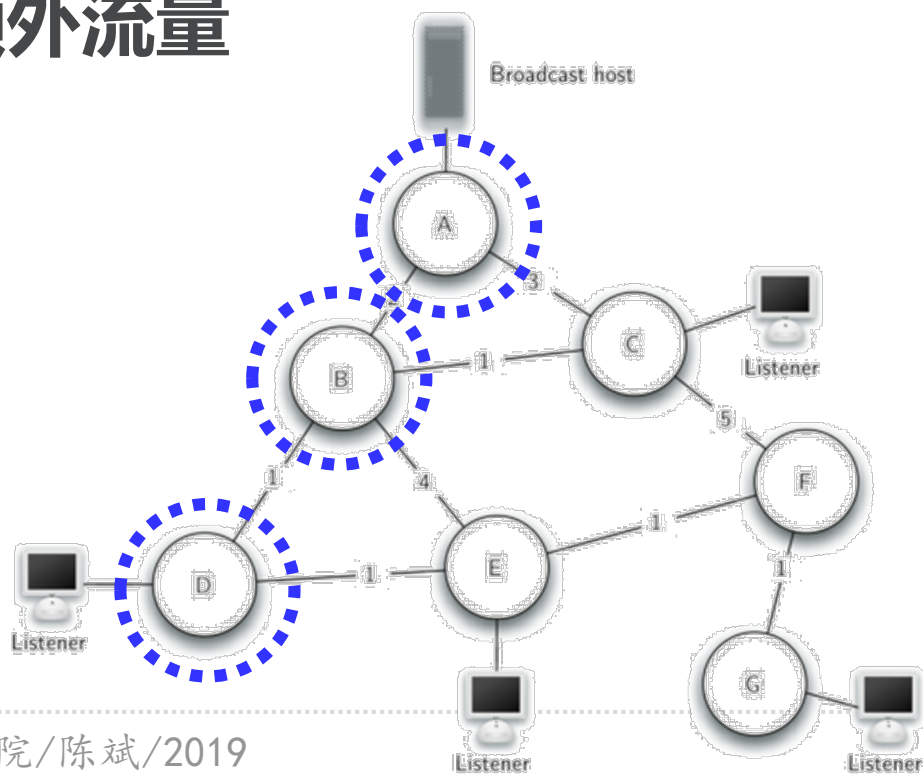
- ❖ 信息广播问题最简单的解法是由广播源维护一个收听者的列表，将每条消息向每个收听者发送一次

如图，每条消息会被发送4次，每个消息都采用最短路径算法到达收听者



信息广播问题：单播解法

- ❖ 路由器A会处理4次相同消息，C仅会处理1次；而B/D位于其它3个收听者的最短路径上，则各会处理转发3次相同消息
- ❖ 会产生许多额外流量



信息广播问题：洪水解法

- ❖ 信息广播问题的暴力解法，是将每条消息在路由器间散布出去
- ❖ 所有的路由器都将收到的消息转发到自己相邻的路由器和收听者
- ❖ 显然，如果没有任何限制，这个方法将造成网络洪水灾难
- ❖ 很多路由器和收听者会不断重复收到相同的消息，永不停止！

信息广播问题：洪水解法

- ❖ 所以，洪水解法一般会给每条消息附加一个生命值 (**TTL**:Time To Live)，初始设置为从消息源到最远的收听者的距离；
- ❖ 每个路由器收到一条消息，如果其TTL值大于0，则将TTL减少1，再转发出去
如果TTL等于0了，则就直接抛弃这个消息。
- ❖ TTL的设置防止了灾难发生，但这种洪水解法显然比前述的单播方法所产生的流量还要大。

信息广播问题：最小生成树

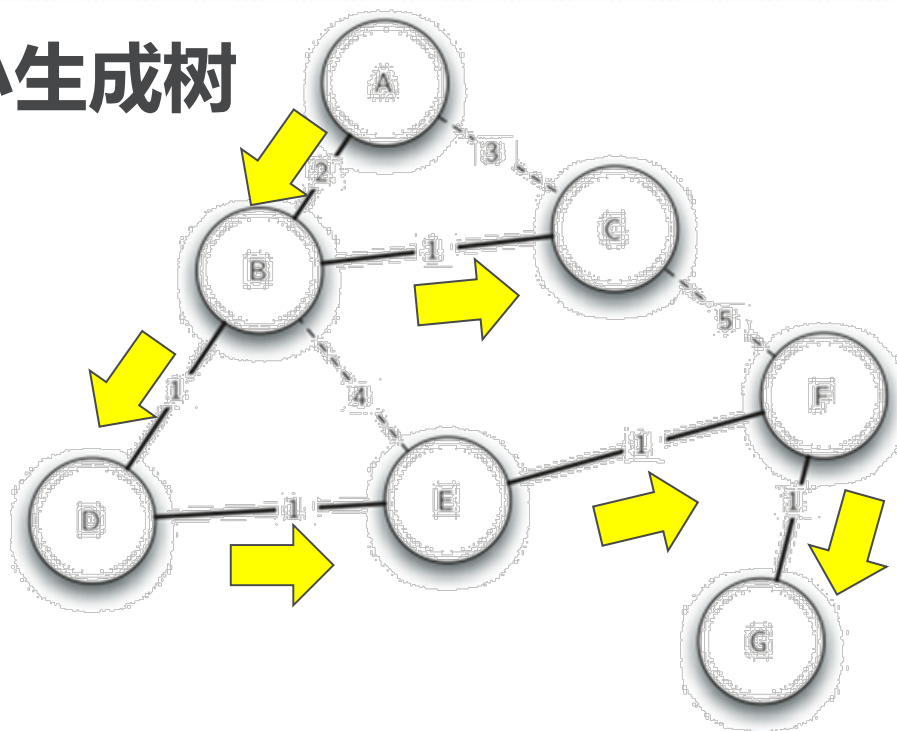
❖ 信息广播问题的最优解法，依赖于路由器关系图上选取具有**最小权重的生成树** (minimum weight spanning tree)

生成树：拥有图中**所有的顶点**和**最少数量**的边，以保持连通的子图。

❖ 图 $G(V,E)$ 的最小生成树 T ，定义为包含所有顶点 V ，以及 E 的无圈子集，并且边权重之和最小

信息广播问题：最小生成树

❖ 图为一个最小生成树



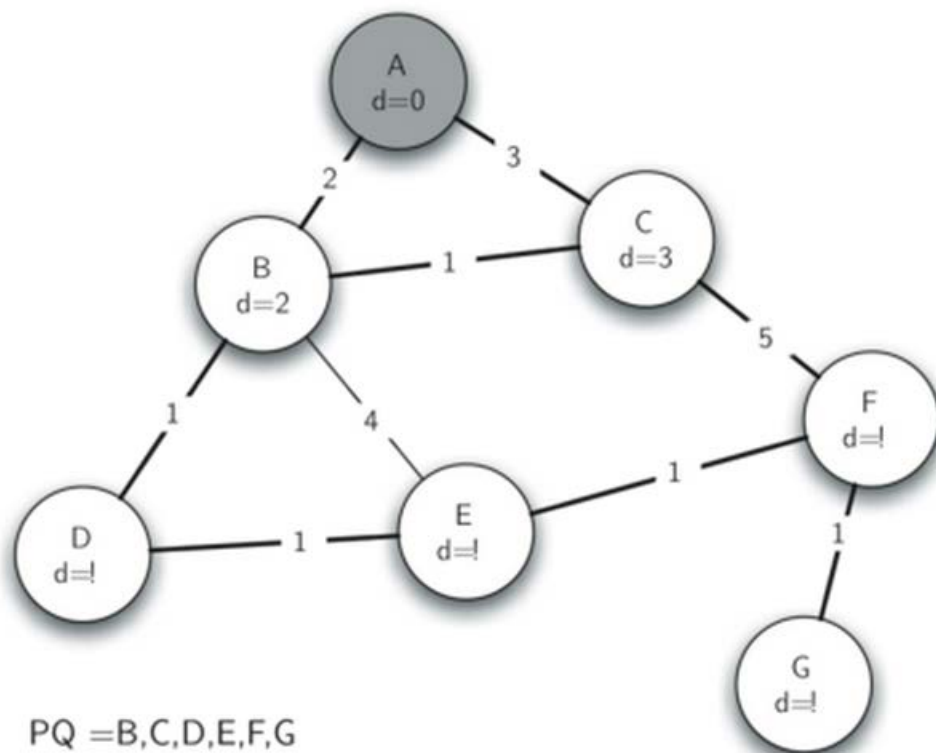
❖ 这样信息广播就只需要从A开始，沿着树的路径层次向下传播

就可以达到每个路由器只需要处理1次消息，同时总费用最小。

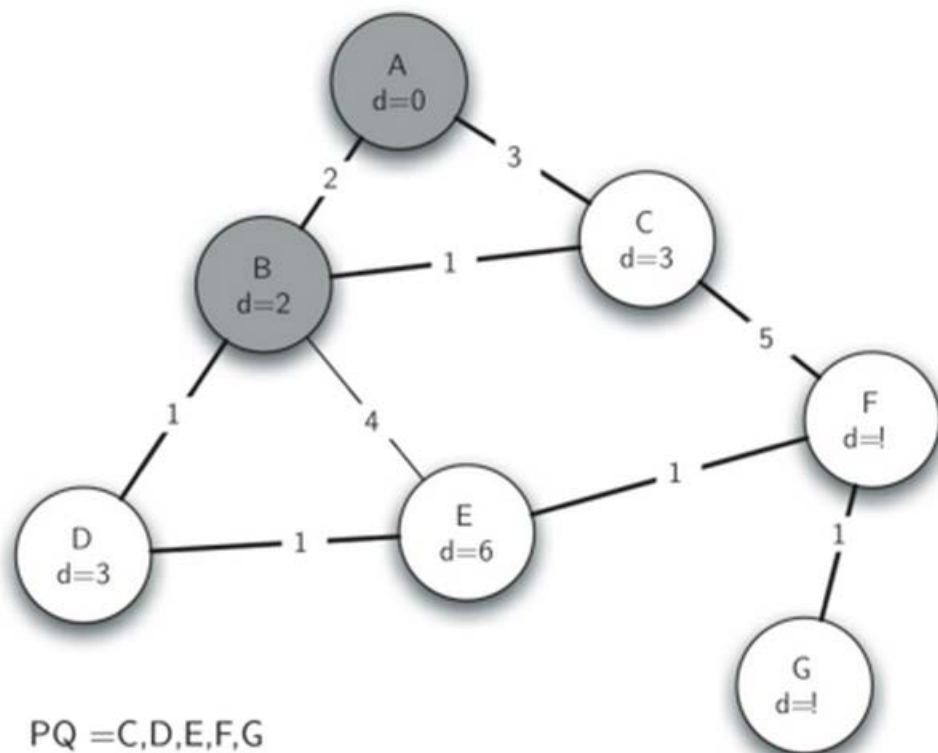
最小生成树：Prim算法

- ❖ 解决最小生成树问题的Prim算法，属于“**贪心算法**”，即每步都沿着最小权重的边向前搜索。
- ❖ 构造最小生成树的思路很简单，如果T还不是生成树，则反复做：
找到一条最小权重的可以安全添加的边，将边添加到树T
- ❖ “**可以安全添加**”的边，定义为一段顶点在树中，另一端不在树中的边，以便保持树的无圈特性

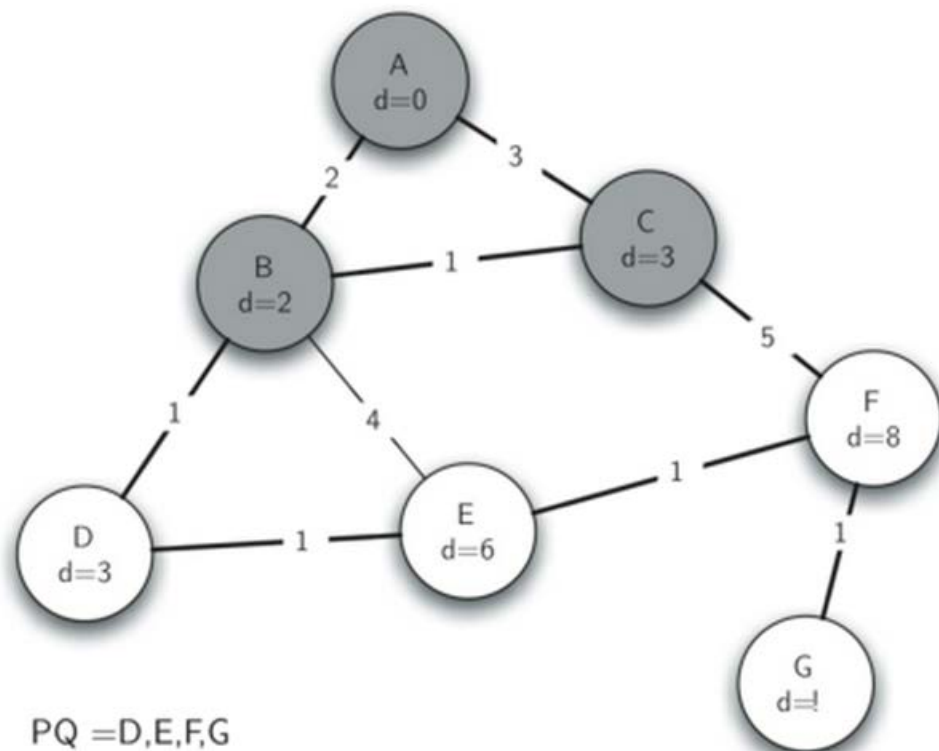
最小生成树：Prim算法示例



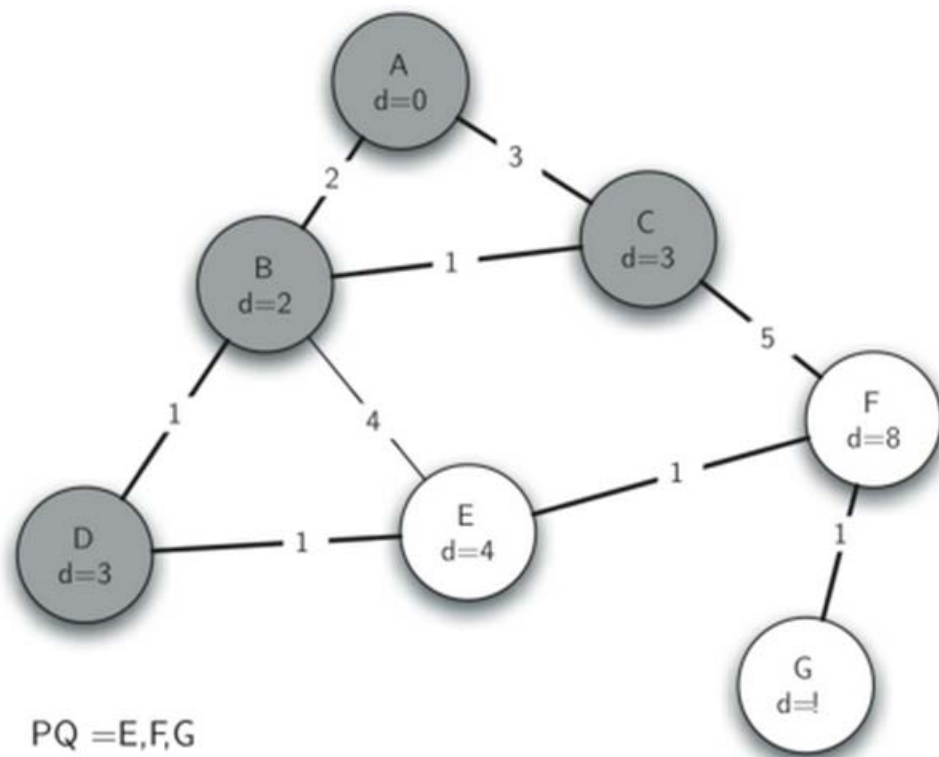
最小生成树：Prim算法示例



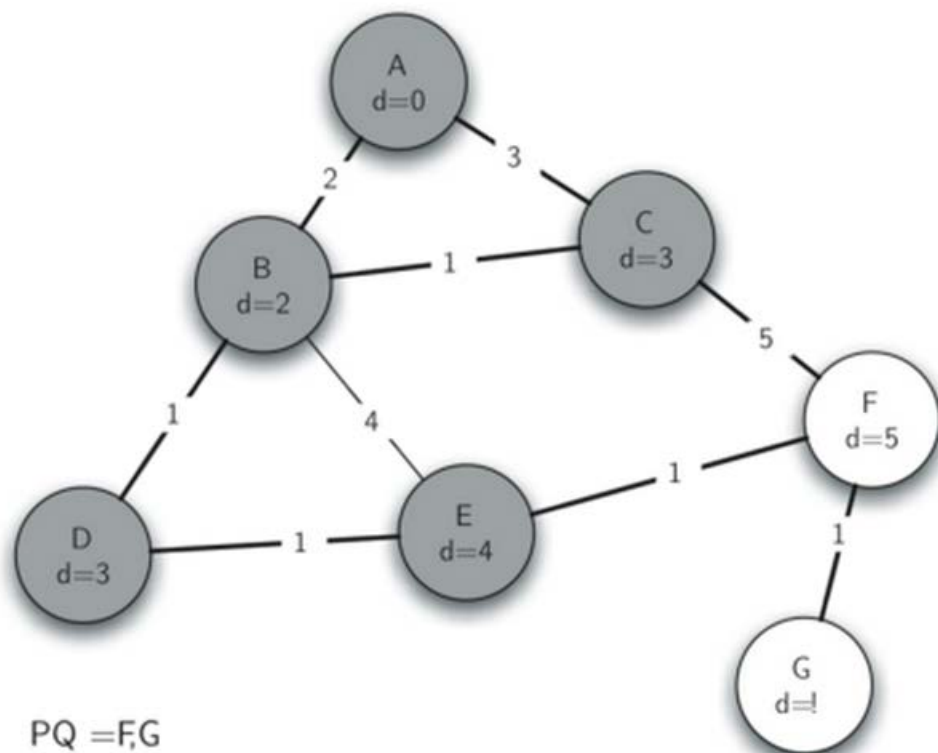
最小生成树：Prim算法示例



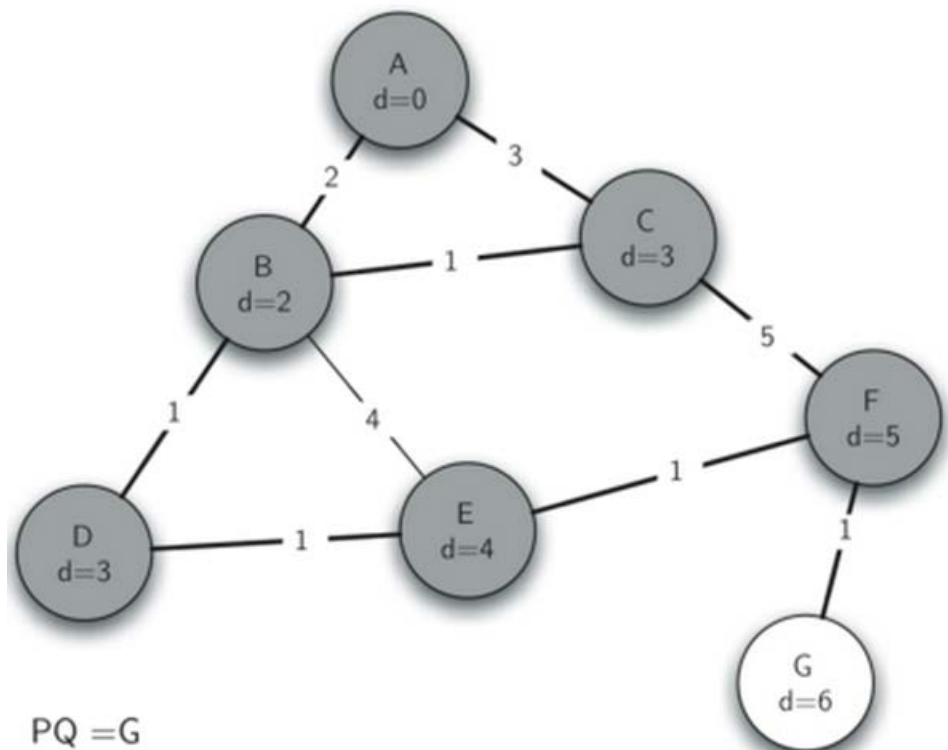
最小生成树：Prim算法示例



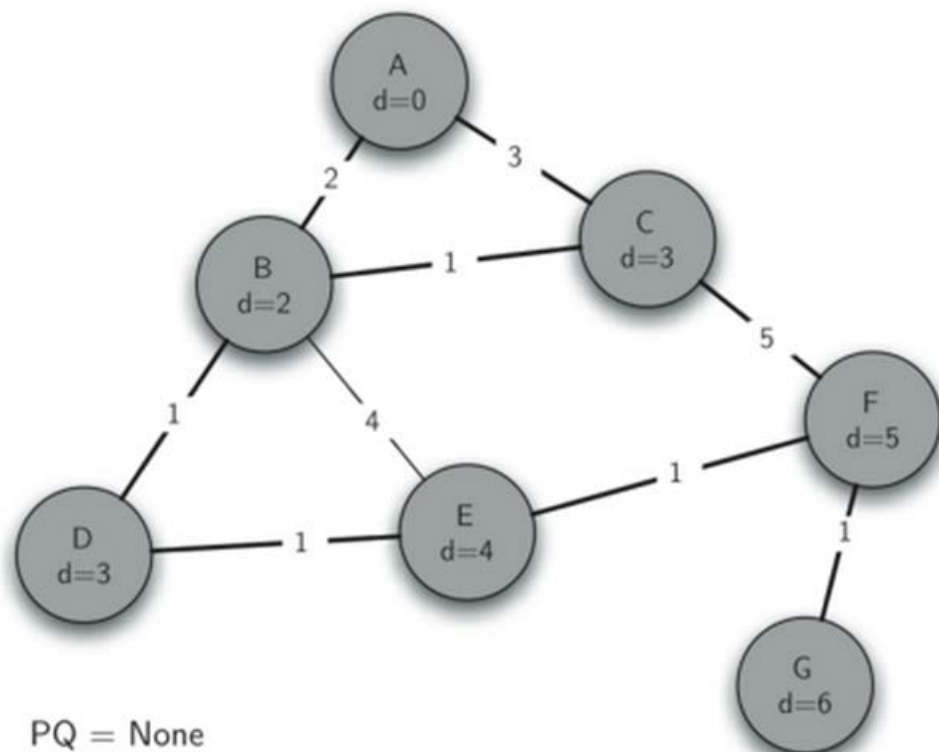
最小生成树：Prim算法示例



最小生成树：Prim算法示例



最小生成树：Prim算法示例



prim算法：最小生成树

```
1  from pythonds.graphs import PriorityQueue, Graph, Vertex
2  import sys
3
4
5  def prim(G, start):
6      pq = PriorityQueue()
7      for v in G:
8          v.setDistance(sys.maxsize)
9          v.setPred(None)
10     start.setDistance(0)
11     pq.buildHeap([(v.getDistance(), v) for v in G])
12     while not pq.isEmpty():
13         currentVert = pq.delMin()
14         for nextVert in currentVert.getConnections():
15             newCost = currentVert.getWeight(nextVert)
16             if nextVert in pq and newCost < nextVert.getDistance():
17                 nextVert.setPred(currentVert)
18                 nextVert.setDistance(newCost)
19                 pq.decreaseKey(nextVert, newCost)
```

