图

大连理工大学

刘 馨 月

主要内容

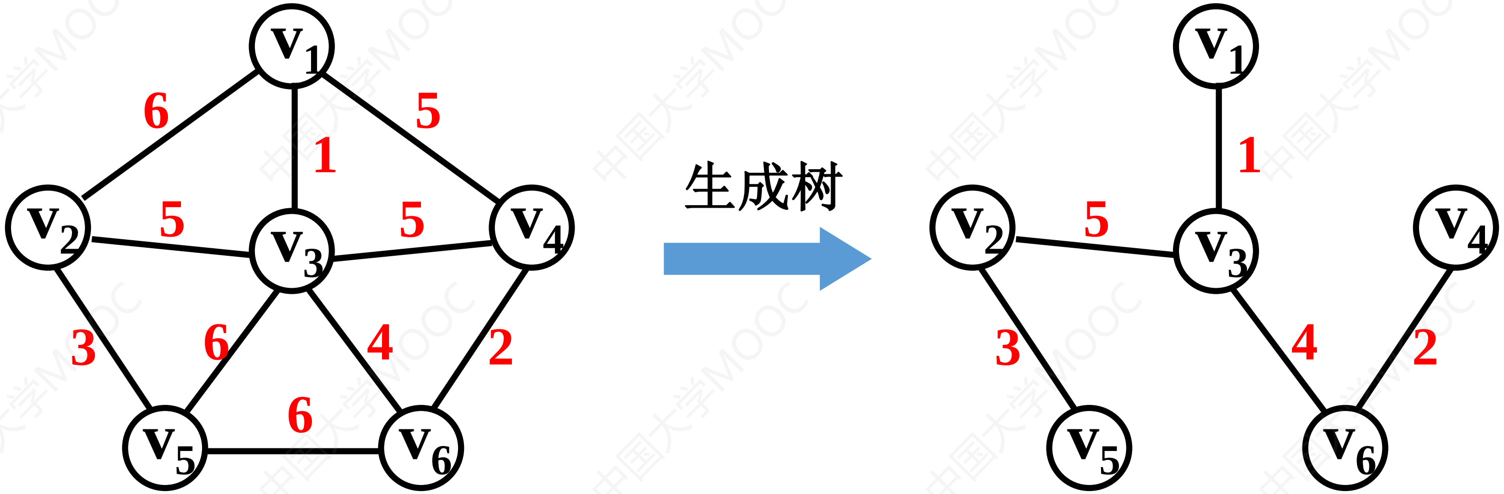
- 图的基本概念
- 图的存储
- 图的遍历
- 最小生成树
- 最短路径
- 关键路径

最小生成树

最小生成树

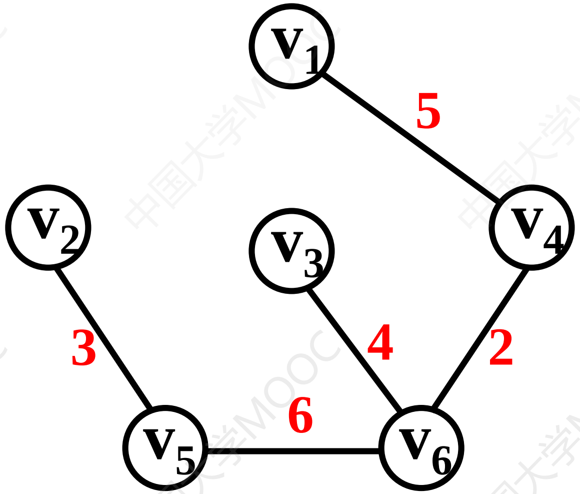
- 最小生成树的概念
  - 连通图的最小生成树
  - 应用举例
- 最小生成树的构造算法
  - **Prim算法**
  - **Kruskal算法**

最小生成树的概念

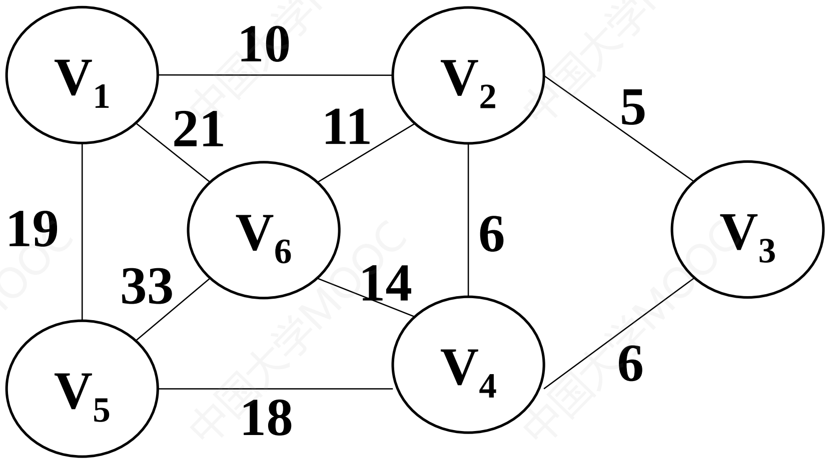


生成树的代价:为树上各边的权之总和。

所有的生成树中，代价最小的生成树称为图G的最小生成树(minimum-cost spanning tree，简称MST)。

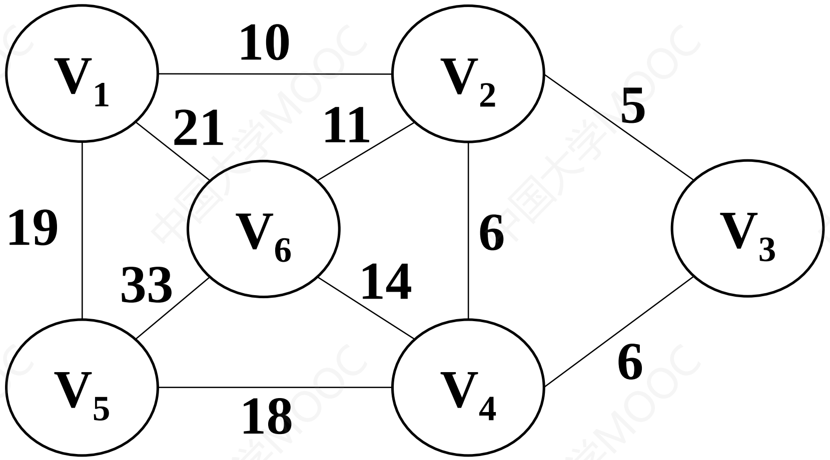


最小生成树应用举例

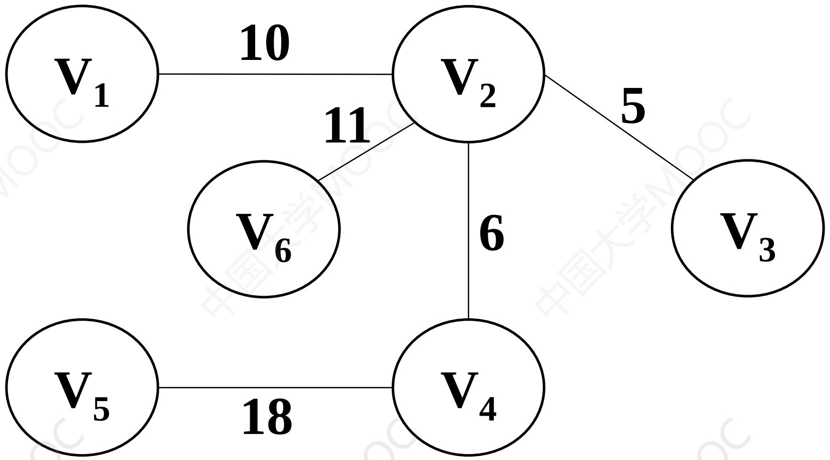


- 上图代表6个城市间的交通网，边上的权表示公路的造价
- 现在要用公路把6个城市连接起来（这至少要修5条公路）
- 如何设计使得这5条公路的总造价最少呢？

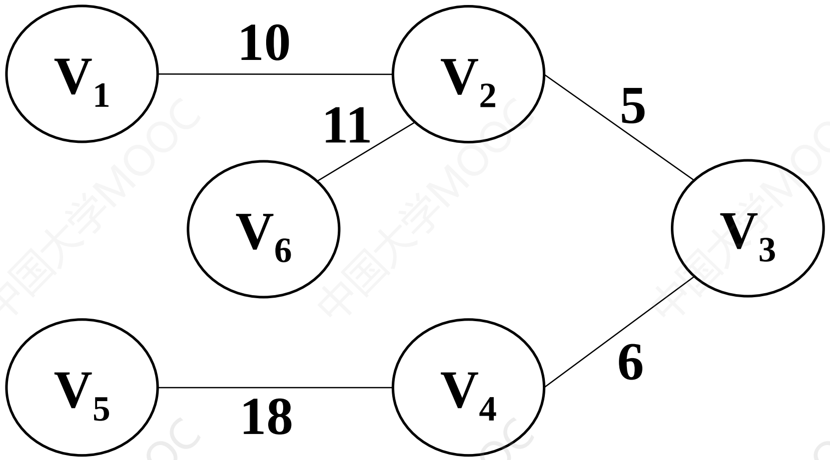
最小生成树应用举例



(a)



(a)的MST



(a)的MST

最小生成树的构造方法

- 普里姆（Prim）算法
- 克鲁斯卡尔（Kruskal）算法

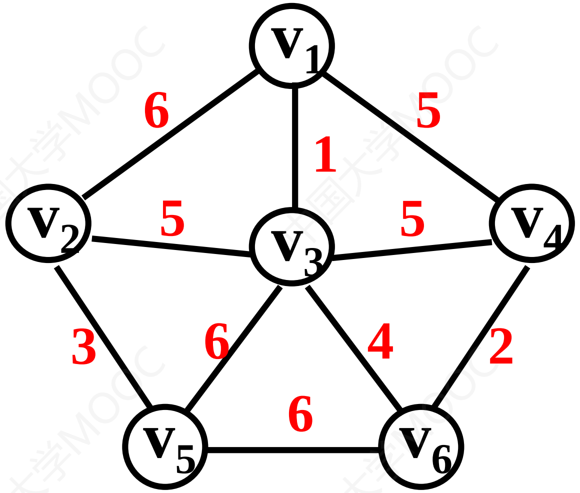
Prim算法

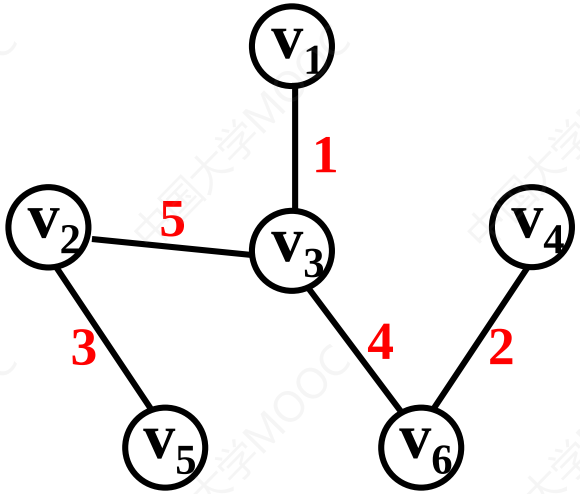
思想：

- G = (V, E) 是具有 n 个顶点的连通图，设 U 是最小生成树中顶点的集合，TE 是最小生成树中边的集合；
- 初始，U = { u_1 } ，TE = { } ，
- 重复执行:
  - 在所有 u∈U，v∈V-U 的边（ u ， v ）中寻找代价最小的边(u’ ， v’) ，并纳入集合 TE 中；
  - 同时将 v’ 纳入集合 U 中；
- 直至 U = V 为止。

集合 TE 中必有 n-1 条边。

Prim算法　例，





初始： $U = \{ v_1 \}$， $V\text{-}U = \{ v_2 , v_3 , v_4 , v_5 , v_6 \}$　　$TE = \{ \}$

$U = \{ v_1 , v_3 \}$， $V\text{-}U = \{ v_2 , v_4 , v_5 , v_6 \}$　　$< v_1 , v_3 >$

$U = \{ v_1 , v_3 , v_6 \}$， $V\text{-}U = \{ v_2 , v_4 , v_5 \}$　　$< v_3 , v_6 >$

$U = \{ v_1 , v_3 , v_4 , v_6 \}$， $V\text{-}U = \{ v_2 , v_5 \}$　　$< v_6 , v_4 >$

$U = \{ v_1 , v_2 , v_3 , v_4 , v_6 \}$， $V\text{-}U = \{ v_5 \}$　　$< v_3 , v_2 >$

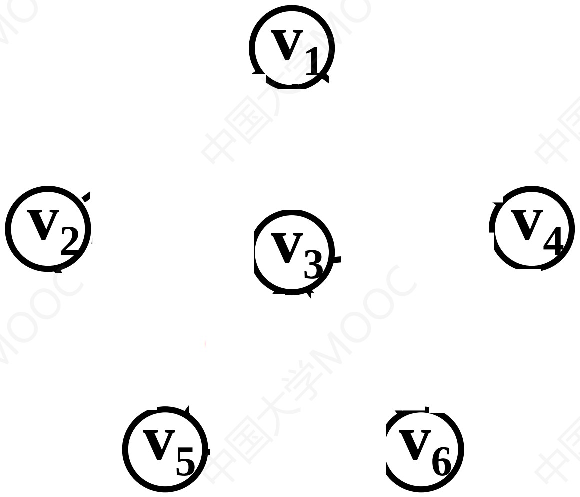
$U = \{ v_1 , v_2 , v_3 , v_4 , v_5 , v_6 \}$， $V\text{-}U = \{ \}$　　$< v_2 , v_5 >$

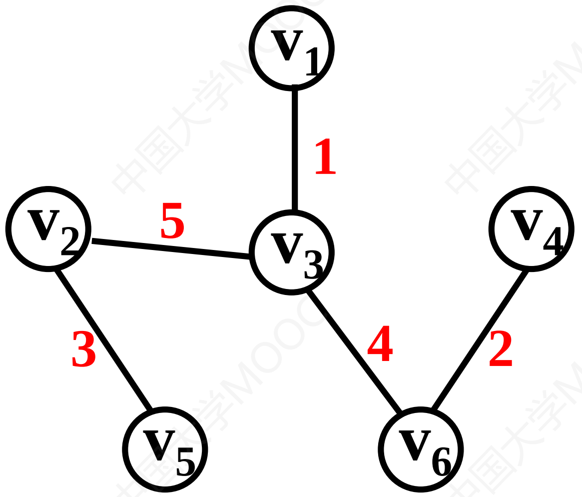
重点：　边一定存在U 与 V-U 之间(MST性质)。

Kruskal算法

- 使用的贪心准则，从剩下的边中选择具有最小权值且不会产生环路的边加入到生成树的边集中。

Kruskal算法





初始 TE = { }

当前权值最小边 (v_5 , v_6)

< v_1 , v_3 >

< v_4 , v_6 >

< v_2 , v_5 >

< v_3 , v_6 >

< v_2 , v_3 >

Kruskal算法

- 基本操作：

（1）确定权值最小的边；

（2）判定一条边所关联的两个顶点是否在一个连通分量中；

（3）如果不是则合并两个顶点所属的连通分量。

Prim 算法

- 算法时间复杂度：$O(n^2)$
- 与边的个数无关；
- 适合于求边稠密的图的最小生成树。

Kruskal 算法

- 算法时间复杂度：$O(e \log e)$

  e为图的边的数目
- 适合于求边稀疏的图的最小生成树。

图

大连理工大学

刘 馨 月