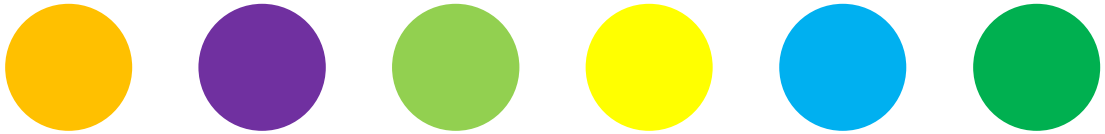
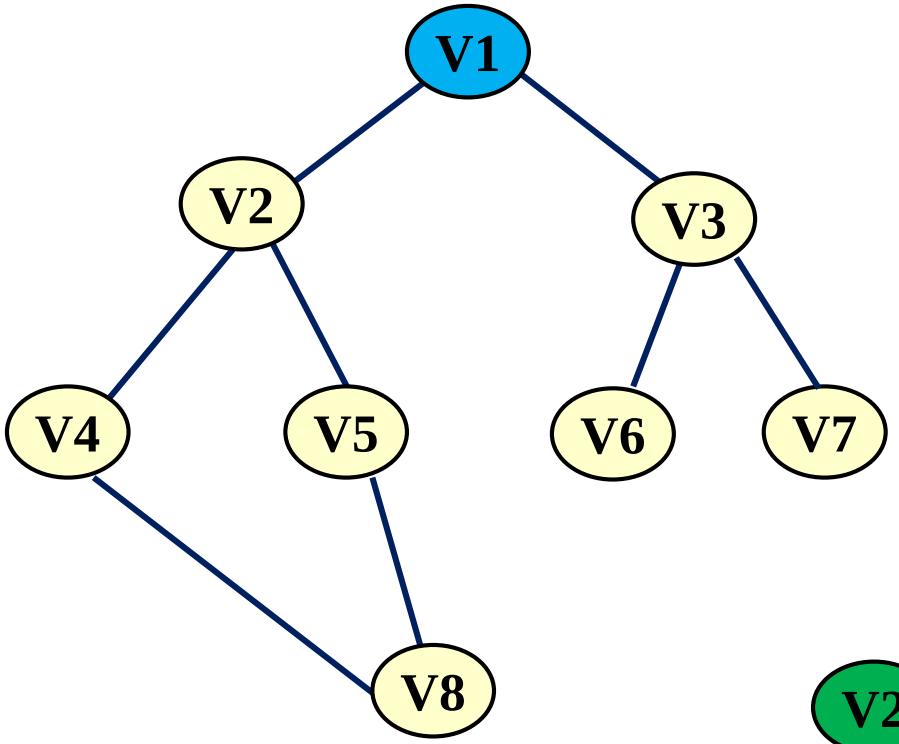
6.4最小生成树（Minimum Spanning Tree）

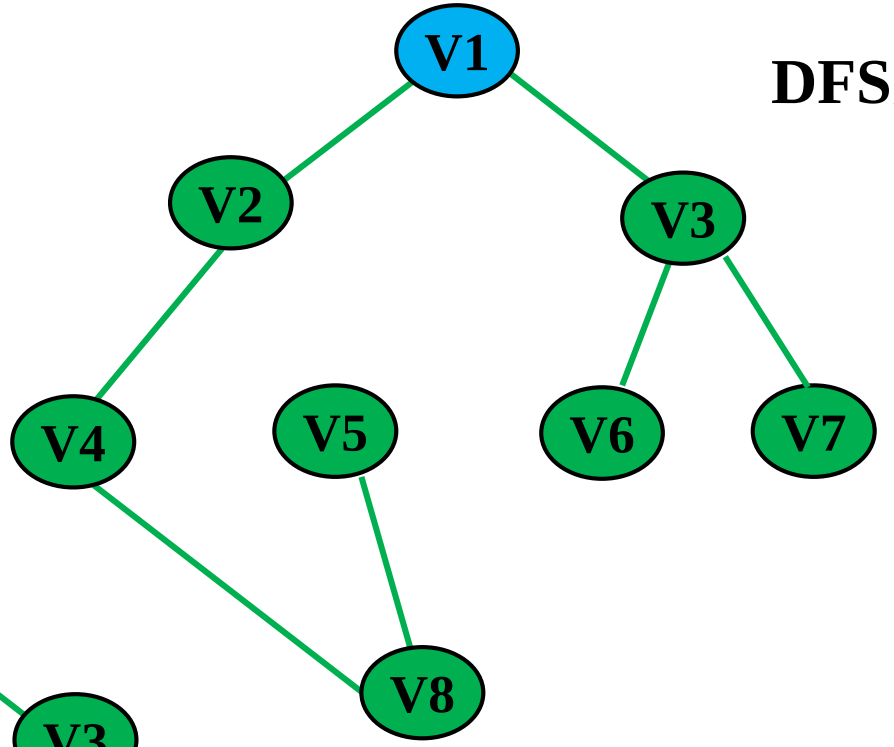
最小生成树应用场景

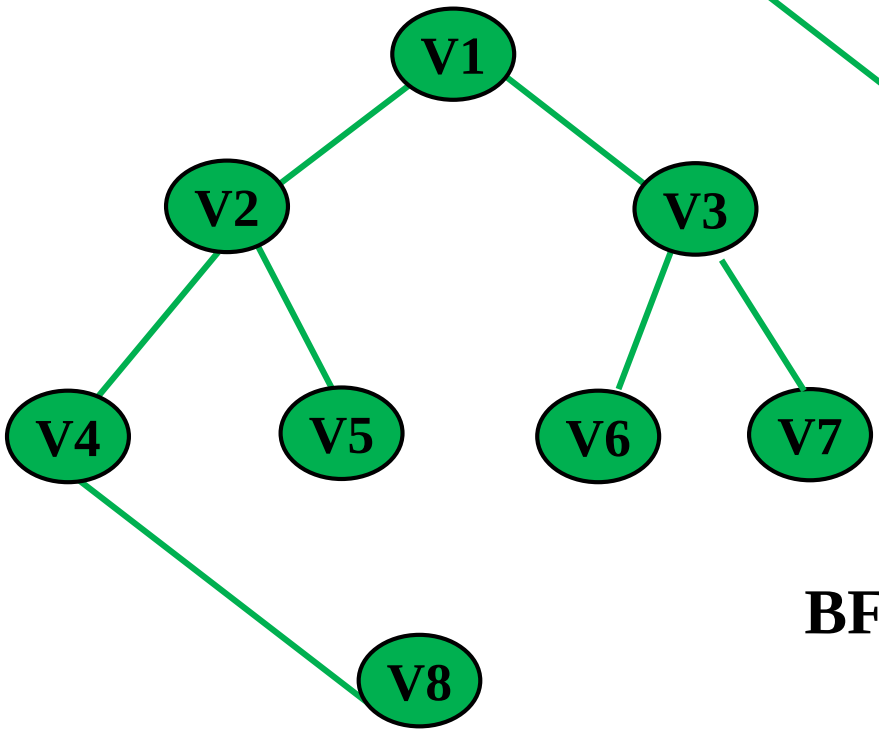
- N个城市之间建立通信网络
- M个村庄之间建立村村通公路
- ……





DFS生成树

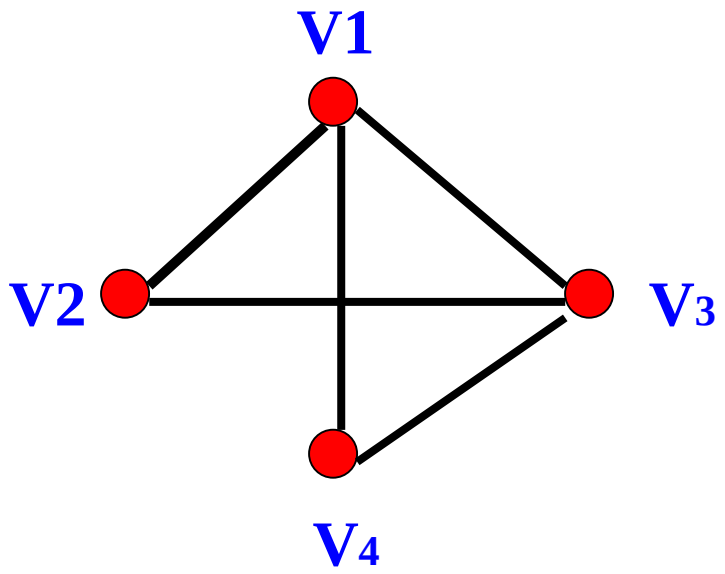


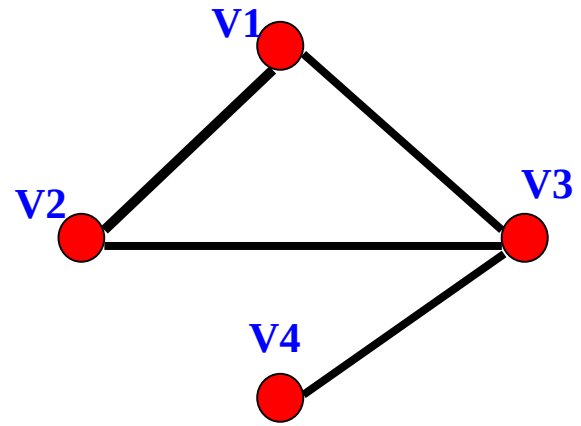


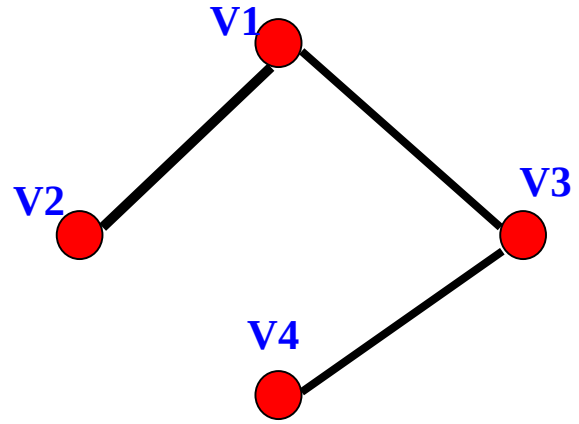
BFS生成树

什么是最小生成树（Minimum Spanning Tree）

- 是一棵生成树
  - 包含n个顶点和n-1条边的连通图；
  - 无回路；
  - DFS生成树和BFS生成树；
- 最小
  - 各边权值之和最小







6.4 Prim（普里姆）算法过程

设G=(V, E)，最小生成树$T_{mst}=(V_T, E_T)$

1. 从图G中任意顶点V_m（$V_m \in V$）开始，将V_m加入到最小生成树；

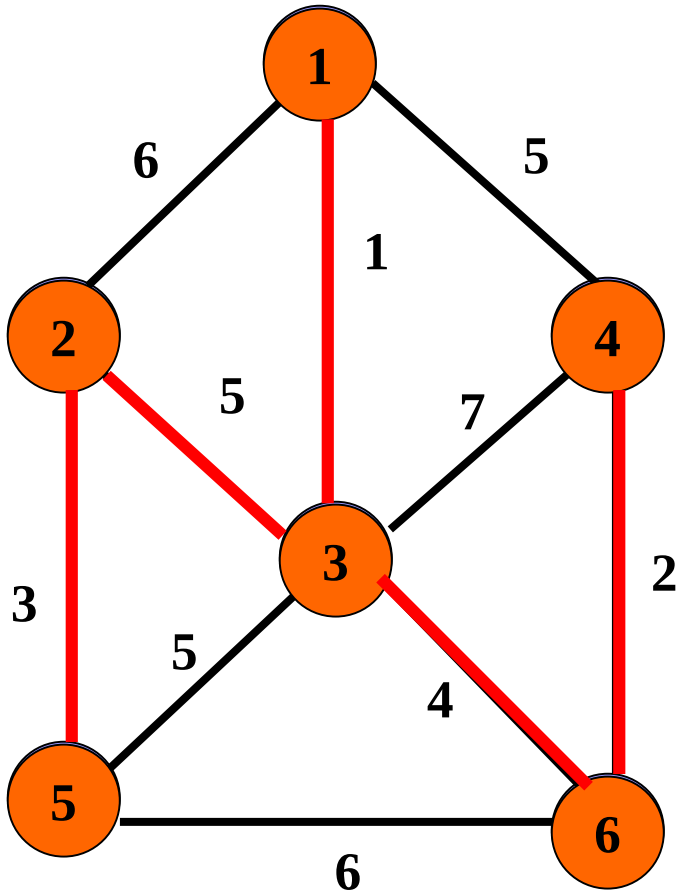
2. 选择代价最小的边（V_k, V_j）加入到最小生成树中，并将顶点V_j加入到最小生成树；

要求：

- ✓ 两个顶点属于不同的集合，$V_k \in V_T$, $V_j \in V - V_T$；
- ✓ 加入的边不能产生回路；

3. 重复这个过程，直到T_{mst}中有n-1条边为止，即$V_T = V$

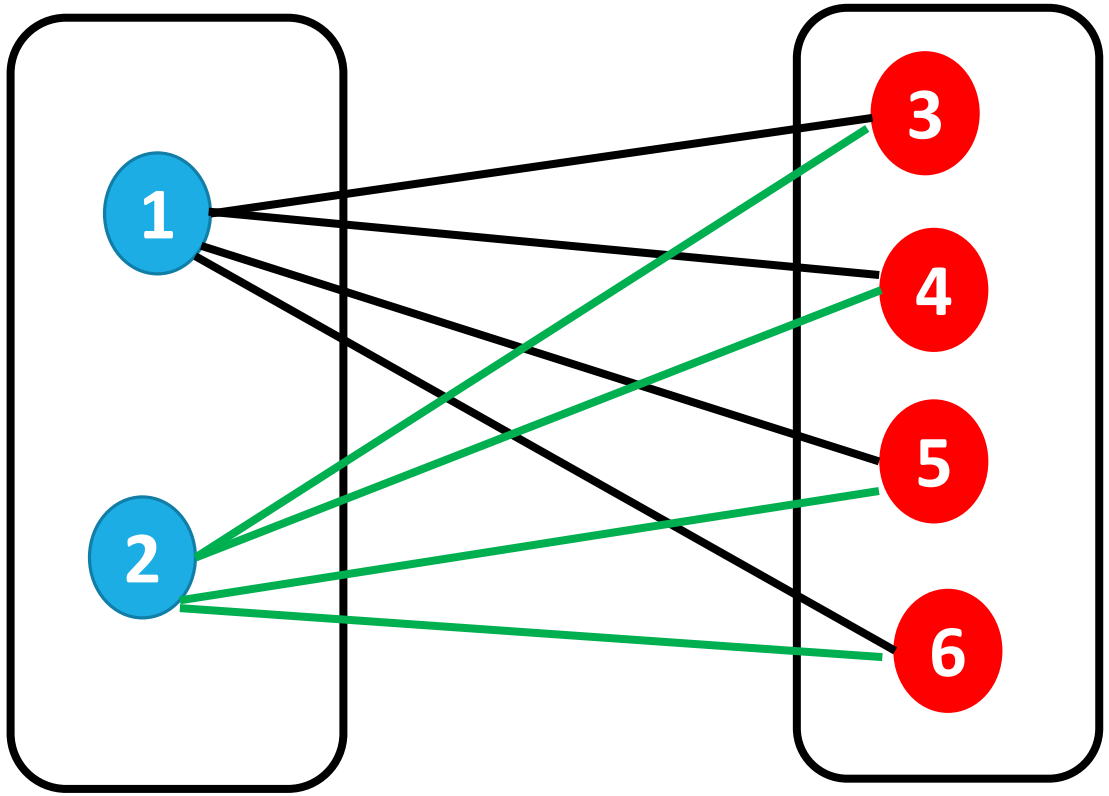
6.4 Prim算法 实例



若侯选边集中最短边不止一条，
可任选其中一条扩充

连通网络的最小生成树不一定是
唯一的，但它们的权是相等的

6.4 Prim算法



先两两比较
1-3 2-3
1-4 2-4
1-5 2-5
1-6 2-6

再四个比较
2-3
1-4
1-5
1-6

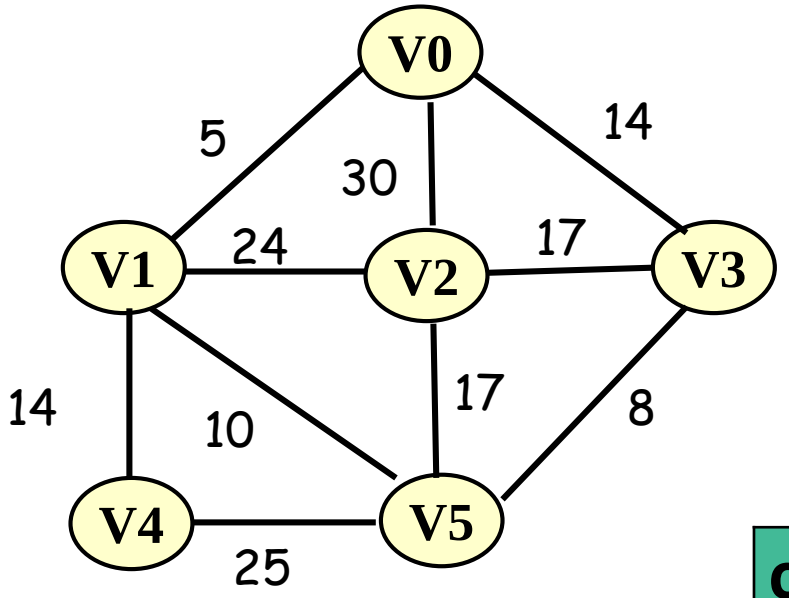
6.4 Prim（普里姆）算法

设置三个数组

1. component [j]数组用来记录已加入最小生成树的顶点j，
   初始化component [j]=0，当顶点j加入最小生成树后，设置component [j]=1

2. distance[j]数组用来记录代价最小的边（V_k, V_j）的权值，其中$V_k \in component[]$
   初始化distance[j]=graphMatrix->graph[0][j]

3. neighbor[j] 数组用来记录代价最小的边（V_k, V_j）对应的顶点V_k，
   初始化neighbor[j] = 0;



V1加入最小生成树之后，结点2，4，5需要更新

V0—V2 > V1—V2

V0—V3 < V1—V3

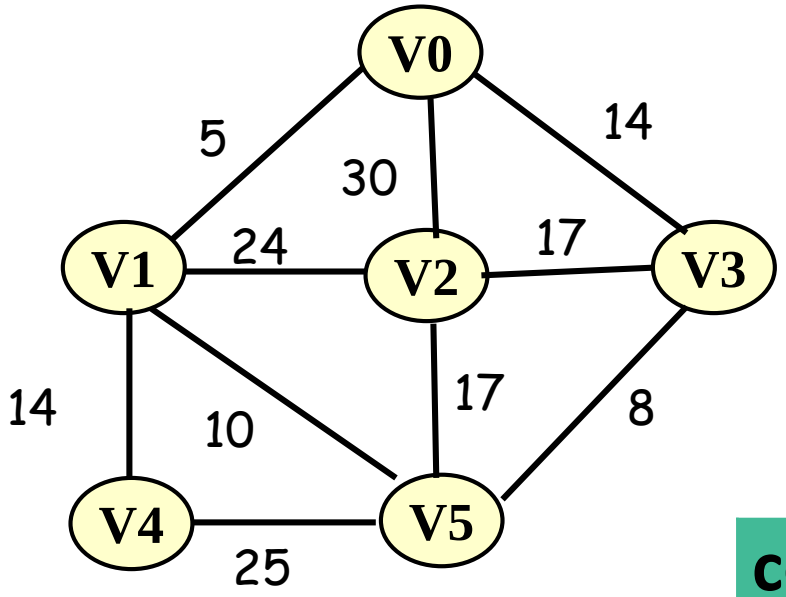
V0—V4 > V1—V4

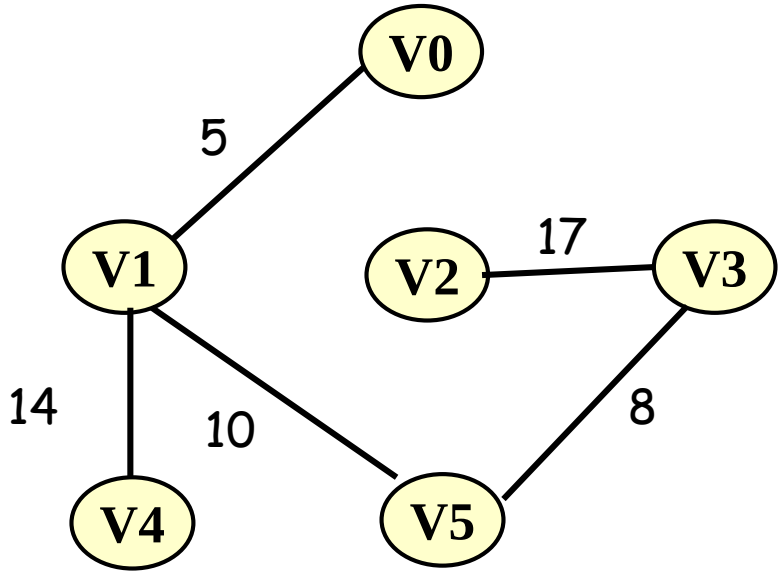
V0—V5 > V1—V5

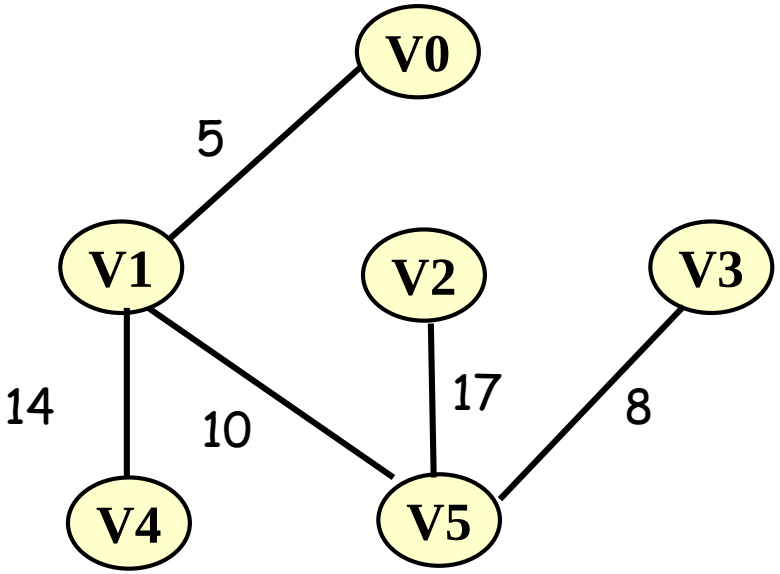
component[]、distance[]和 neighbor[]数组的变化					
0	1	2	3	4	5
1,0,0	0,5,0	0,30,0	0,14,0	0,∞,0	0,∞,0
1,0,0	1,5,0	0,24,1	0,14,0	0,14,1	0,10,1
1,0,0	1,5,0	0,17,5	0,8,5	0,14,1	1,10,1
1,0,0	1,5,0	0,17,5	1,8,5	0,14,1	1,10,1
1,0,0	1,5,0	0,17,5	1,8,5	1,14,1	1,10,1
1,0,0	1,5,0	1,17,5	1,8,5	1,14,1	1,10,1

蓝色表示已加入最小生成树

红色表示数组有更新







component[] 、 distance[]和 neighbor[]数组的变化					
0	1	2	3	4	5
1,0,0	0,5,0	0,30,0	0,14,0	0,∞,0	0,∞,0
1,0,0	1,5,0	0,24,1	0,14,0	0,14,1	0,10,1
1,0,0	1,5,0	0,17,5	0,8,5	0,14,1	1,10,1
1,0,0	1,5,0	0,17,5	1,8,5	0,14,1	1,10,1
1,0,0	1,5,0	0,17,5	1,8,5	1,14,1	1,10,1
1,0,0	1,5,0	1,17,5	1,8,5	1,14,1	1,10,1