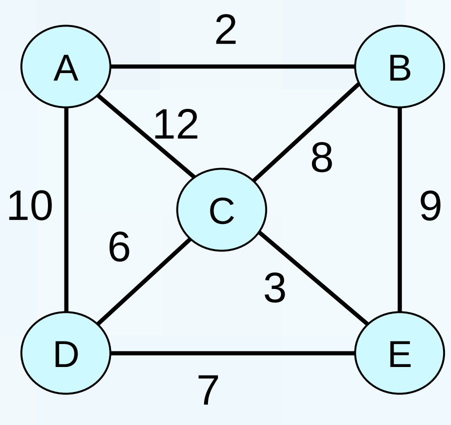
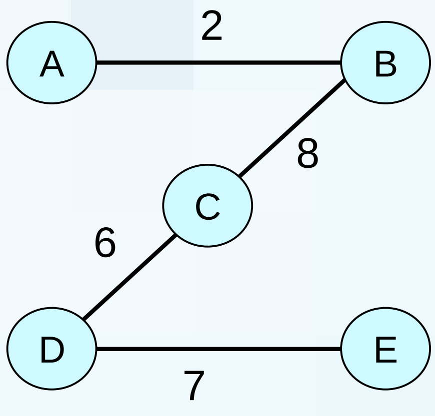
最小生成树问题

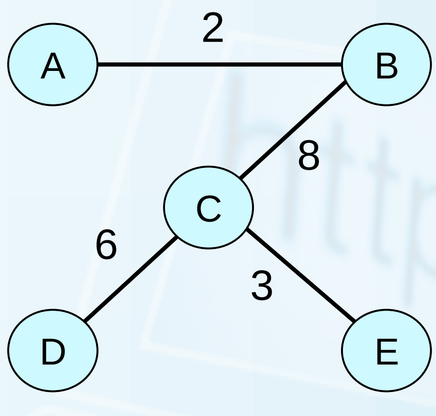
设$G=(V, E, W)$是一个无向连通赋权图，其生成树上各边的权值之和称为该生成树的权值，

在G的所有生成树中，权值最小的生成树称为最小生成树（**Minimal Spanning Trees**）。





（1）W=23



（2）W=19

最小生成树问题

最小生成树问题至少有两种合理的贪心策略：

（1）最近顶点策略：每一步的贪心选择是把不在生成树中的最近顶点添加到生成树中。

- Prim算法。

（2）最短边策略：每一次贪心选择都是在边集中选取不产生回路的最短边。

- Kruskal算法。

9.1 Prim算法

$Prim\ (G)$

//构造最小生成树的 Prim 算法

//输入：加权连通图 $G = \langle V, E \rangle$

//输出：E_T，组成 G 的最小生成树的边的集合

$V_T \leftarrow \{v_0\}$ //可以用任意顶点来初始化树的顶点集合

$E_T \leftarrow \Phi$

for $i \leftarrow 1$ **to** $|V|-1$ **do**

在所有的边 (v, u) 中，求权重最小的边 $e^* = (v^*, u^*)$，

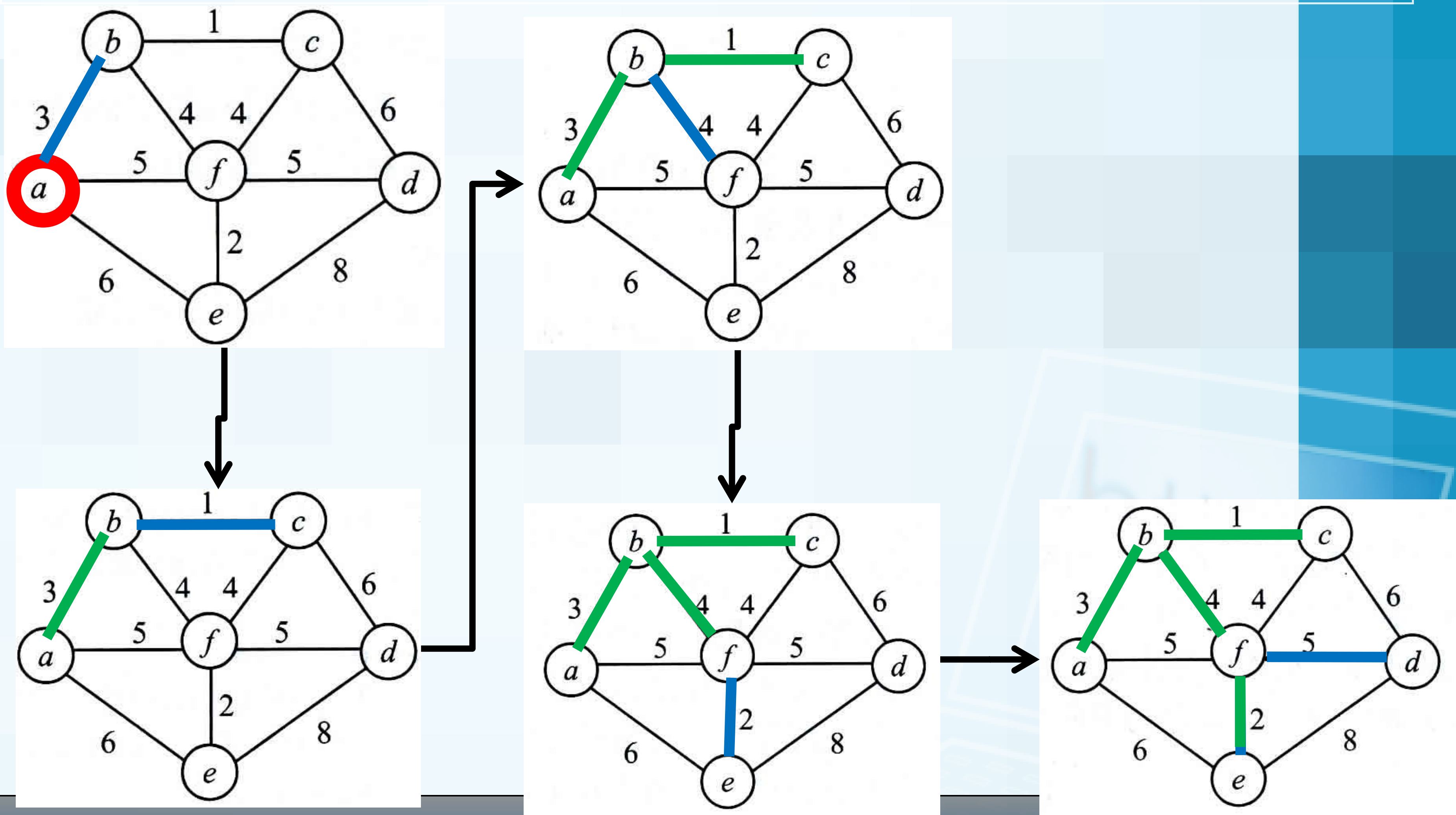
使得 v 在 V_T 中，而 u 在 $V - V_T$ 中

$V_T \leftarrow V_T \cup \{u^*\}$

$E_T \leftarrow E_T \cup \{e^*\}$

return E_T

Prim算法实例


b 1 c
3 4 4 6
a 5 f 5 d
2
6 8
e
b 1 c
3 4 4 6
a 5 f 5 d
2
6 8
e
b 1 c
3 4 4 6
a 5 f 5 d
2
6 8
e
b 1 c
3 4 4 6
a 5 f 5 d
2
6 8
e
b 1 c
3 4 4 6
a 5 f 5 d
2
6 8
e

9.2 Kruskal算法

- **Kruskal算法**

  该算法首先**将图中的边按照权重的非递减顺序进行排序**，然后从一个空子图开始，试图按顺序把边加到当前的子图中。

9.2 Kruskal算法

算法 Kruskal(G)

//构造最小生成树的Kruskal算法

//输入：加权连通图

//输出：E_T

按照边的权重$w(e_i)$的非递减顺序对集合$E=\{e_i\}$排序

$E_T \leftarrow \phi$； ecounter$\leftarrow$0; k$\leftarrow$0;

While ecounter < |V|-1

  k$\leftarrow$k+1

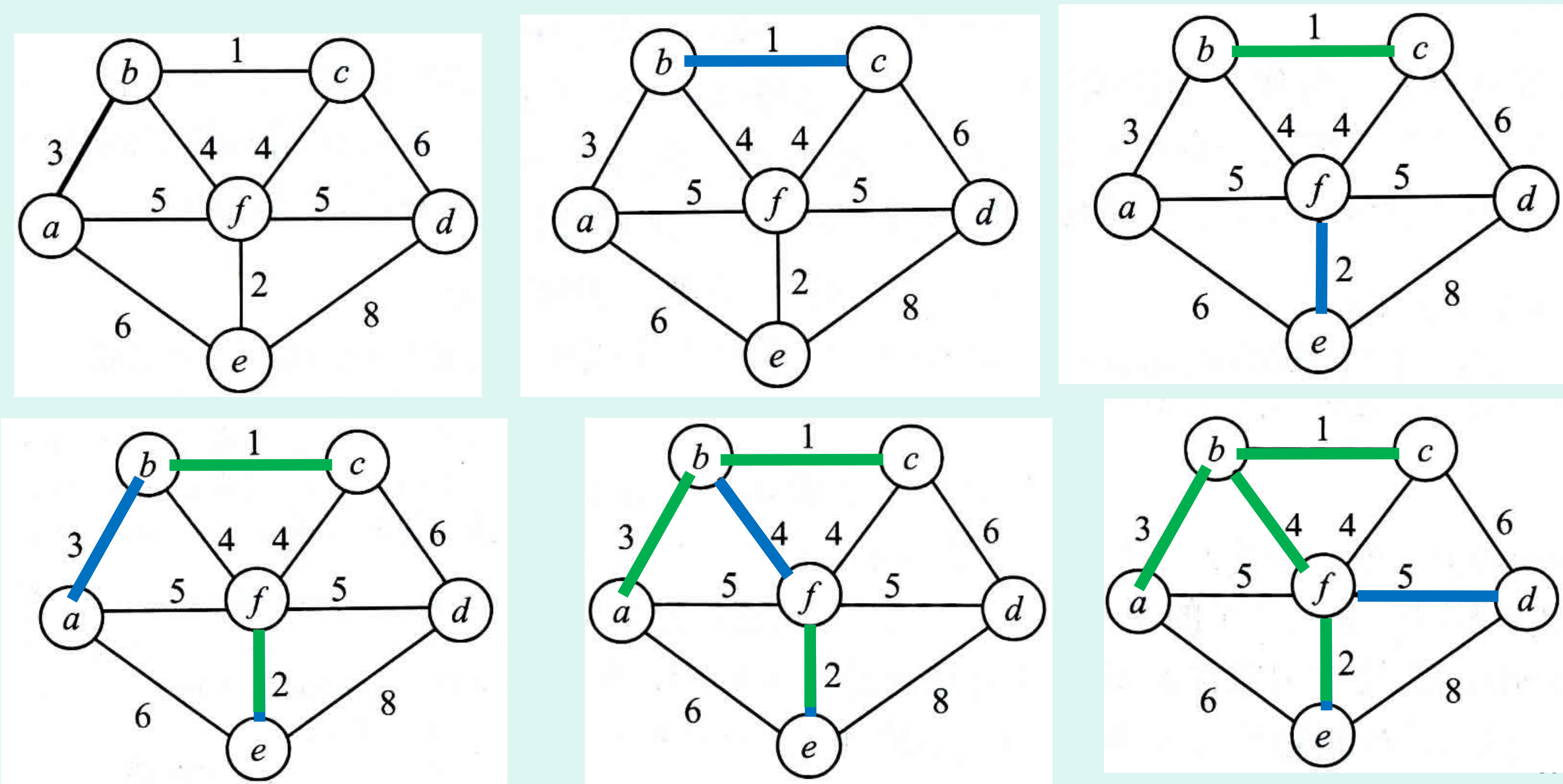
  if $E_T \cup \{e_k\}$ 无回路

    $E_T \leftarrow E_T \cup \{e_k\}$; ecounter$\leftarrow$ecounter+1;

Return E_T

9.2 Kruskal算法实例

bc	*ef*	*ab*	*bf*	*cf*	*af*	*df*	*ae*	*cd*	*de*
1	2	3	4	4	5	5	6	6	8



最小生成树问题

- 时间复杂度
  - ***Prim*** 算法复杂度取决于对每个顶点**求不在生成树中的最近顶点**，复杂度为$O(n^2)$
  - ***Kruskal*** 算法复杂度取决于对给定的边按**权重排序所需的时间**，复杂度为 $O(e\log e)$

其中*e*为边的数目， *n*为顶点的数目。