生成树

回顾

- 内容1：表达式的（逆）波兰记法
- 内容2：二叉搜索树
- 内容3：前缀码与Huffman编码

本节提要

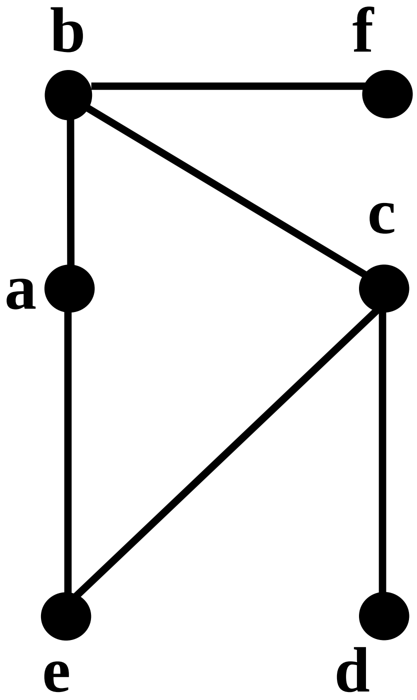
- 内容1：生成树及其构造方法
- 内容2：最小生成树算法

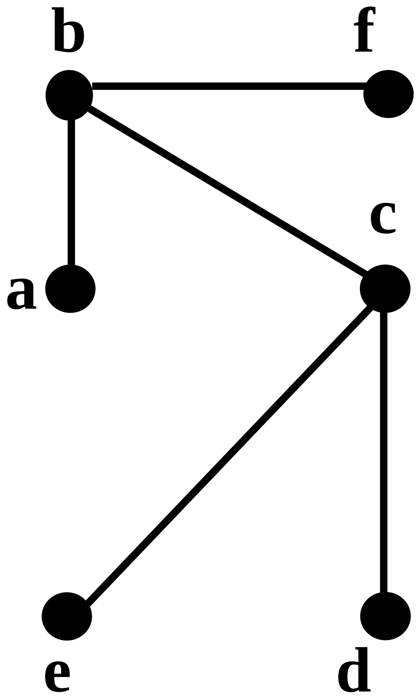
生成树

- 定义：若图G的生成子图是树，则该子图称为G的生成树。
  - 注：含有G的所有顶点的子图称为G的生成子图
- 无向图G连通 当且仅当 G有生成树
  - 证明(充分性显然):

  ⇒ 注意：若G是有简单回路的连通图，删除回路上的一条边，G中的回路一定减少。(因此，用“破圈法”总可以构造连通图的生成树)
- 简单无向图G是树 当且仅当 G有唯一的生成树。

构造生成树：深度优先搜索





深度优先搜索算法

Procedure DFS(G: 带顶点v_1, …,v_n的连通图)

T:=只包含顶点v_1的树;

visit(v_1);

Procedure visit(v: G的顶点)

for v每个邻居w {

```
if w不在T中 then {
   加入顶点w和边{v, w}到T;
    visit(w);
}
}
```

回溯（八皇后）

在n×n格的棋盘上放置彼此不受攻击的n个皇后。

从空棋盘开始

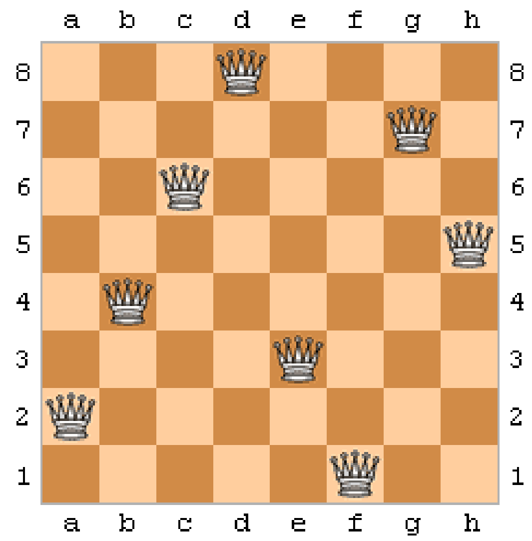
尝试第1列，第1行，…n行；

尝试第2列，第1行，…n行；

….

尝试第k+1列，第1行，…n行；

…



回溯（子集和）

给定一组正整数$x_1, \ldots, x_n$，和为**M**的一个子集?

从空子集开始

尝试添加一项,

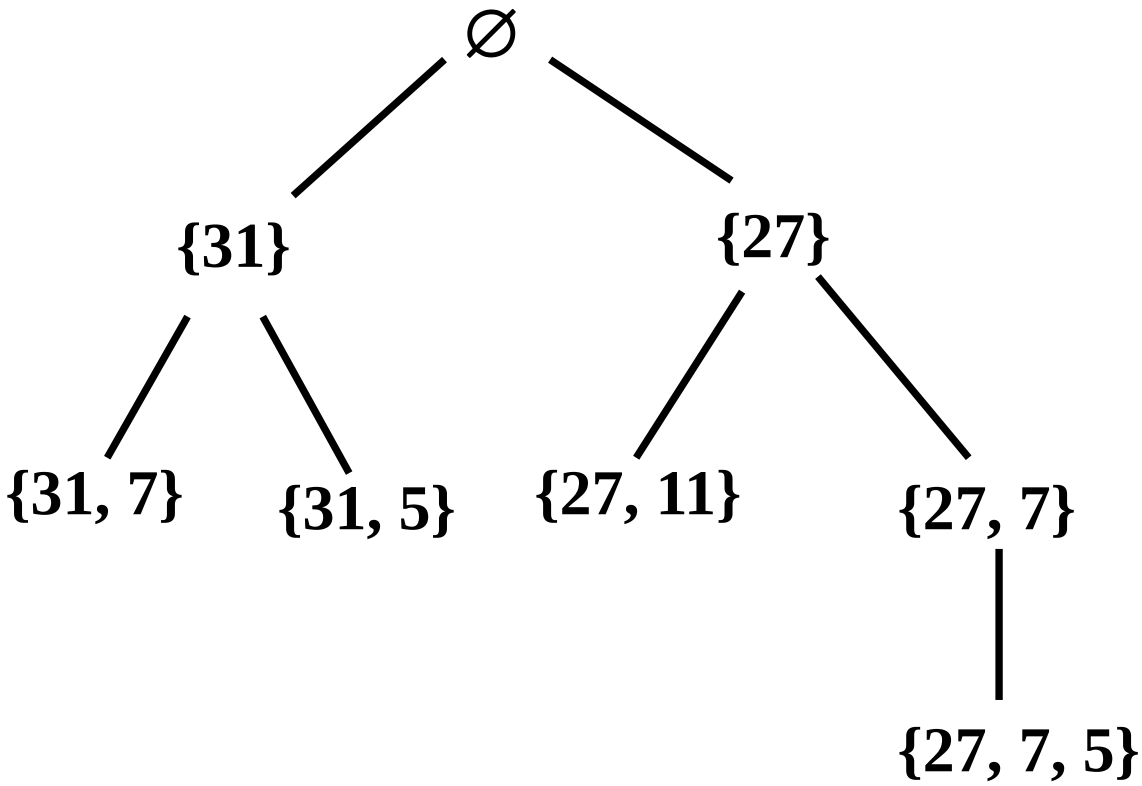
和等于**M**，结束;

和不超过**M**，子集包含它;

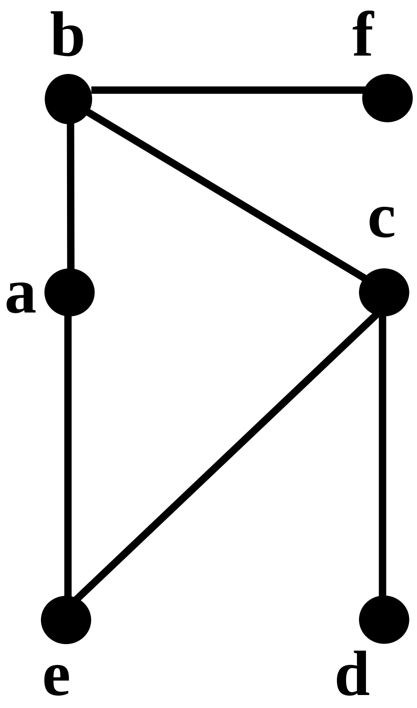
没有合适添加项，去掉最后一项,

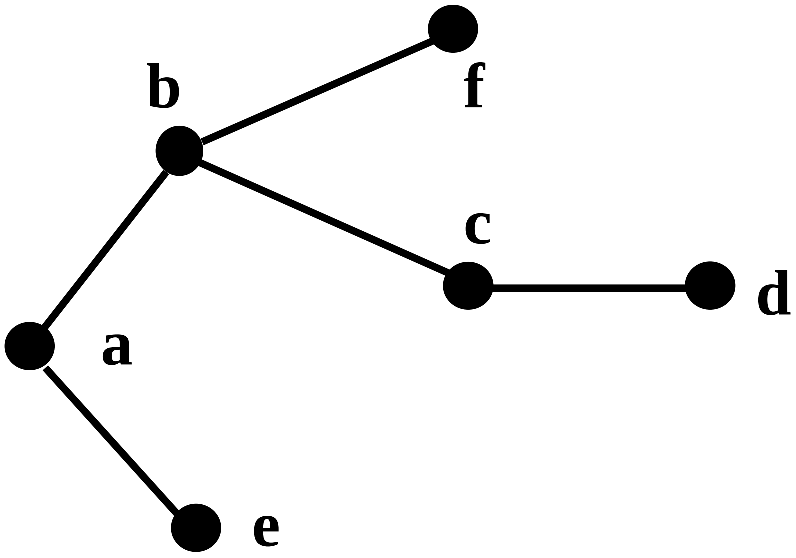
回溯（子集和）

举例：{31, 27, 15, 11, 7, 5}, 和为39的子集?



构造生成树：广度优先搜索





广度优先搜索算法

Procedure BFS(G: 带顶点v_1, …,v_n的连通图)

T:=只包含顶点v_1的树; L:=空表; 把v_1放入表L中

While L非空 {

  删除L中的第一个顶点v;

  for v的每个邻居w {

    if w既不在L中也不在T中 then {

      加入w到L的末尾;

      加入顶点w和边{v, w}到T;

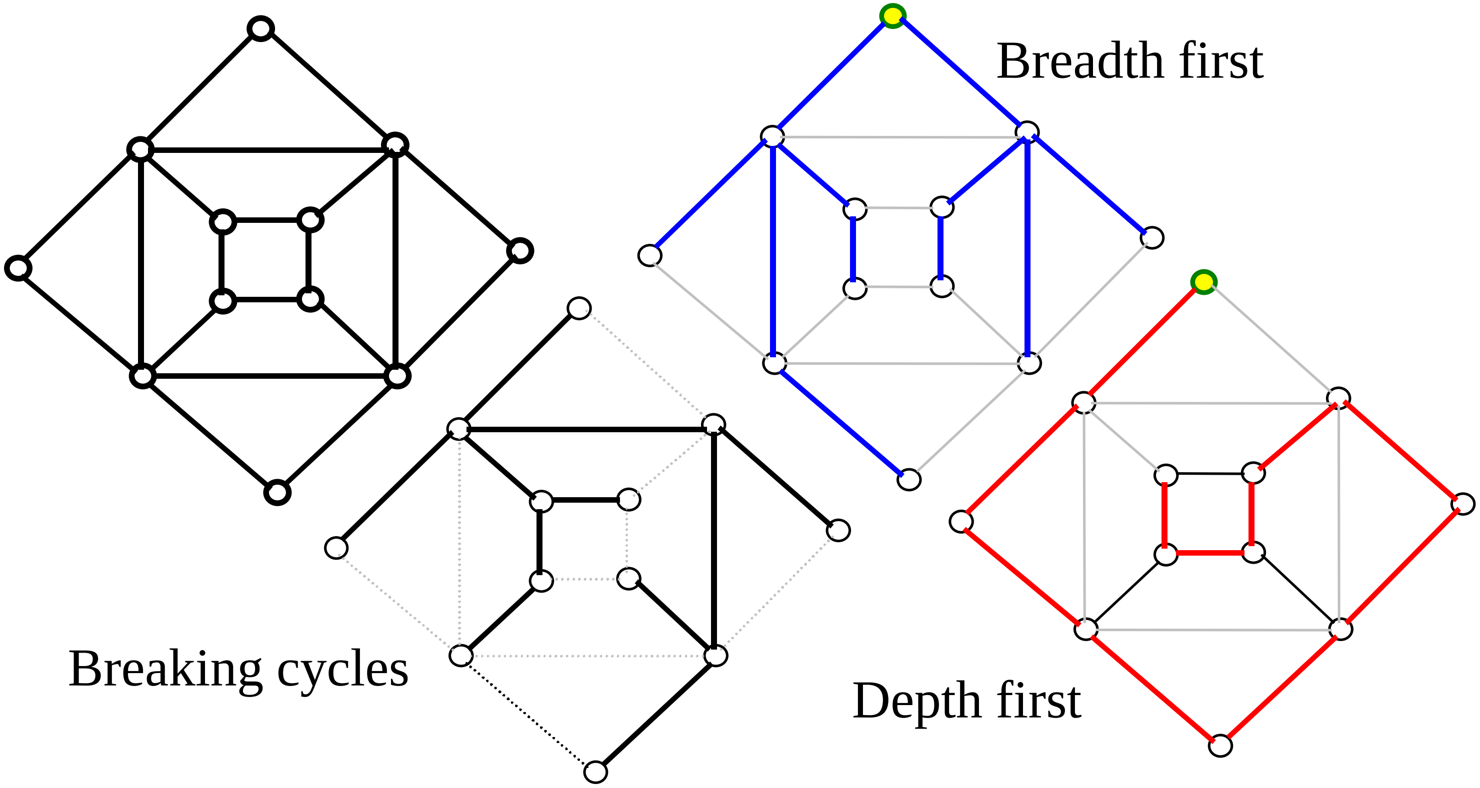
    }

  }

}

例

- 不同的生成树



本节提要

- 内容1：生成树及其构造方法
- 内容2：最小生成树算法

最小生成树 Minimum Spanning Tree

- 考虑边有权重的连通无向图。其生成树可能不唯一。定义生成树的权重为其所含各边之和。一个带权连通图的最小生成树是其权重最小的生成树。
  - 注意，这里的最小(Minimum)并不意味着唯一。

- 最小生成树有广泛的应用。

Prim算法（求最小生成树）

1: E={e}, e是权最小的边

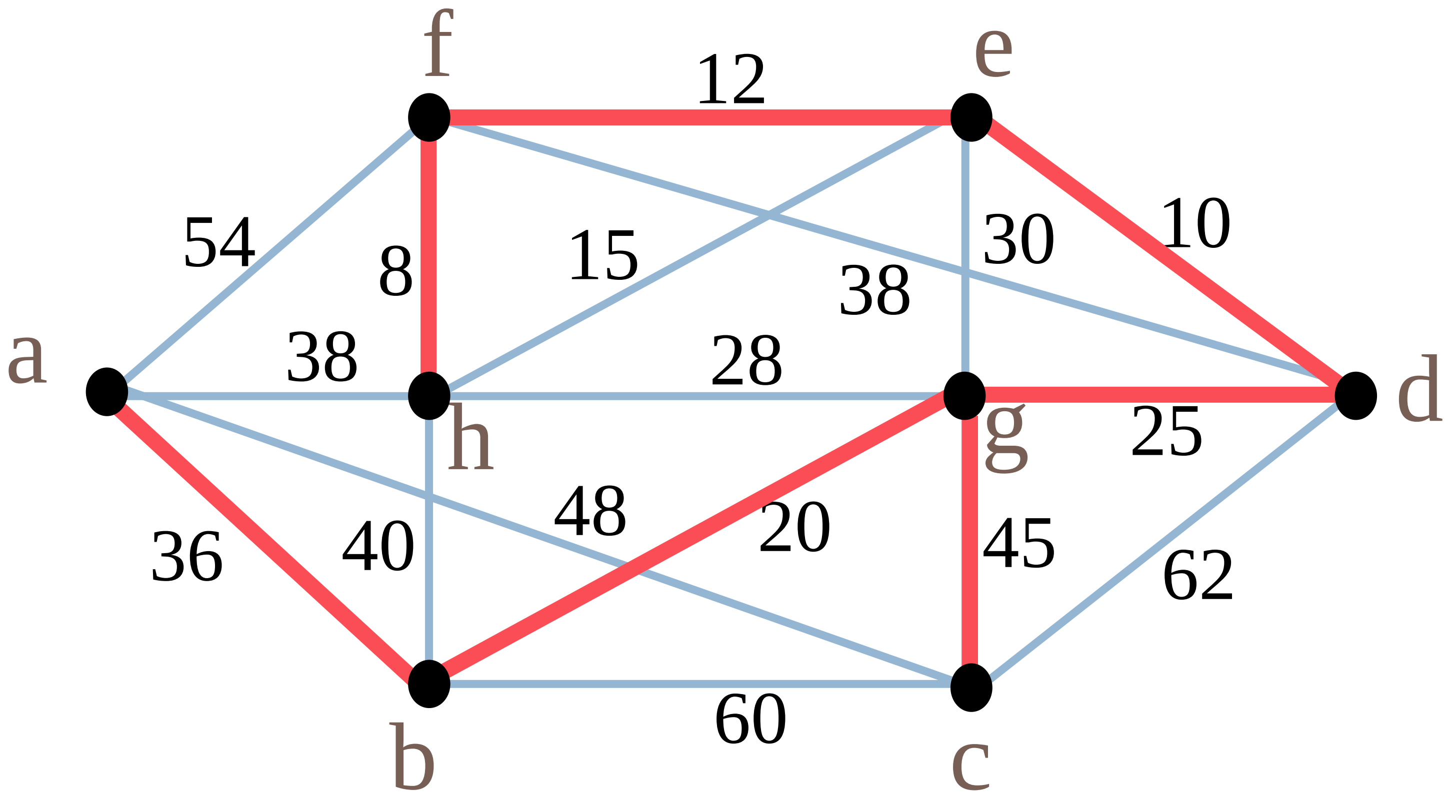
2: 从E以外选择与E里顶点关联，又不会与E中的边构成回路的权最小的边加入E

3: 重复第2步，直到E中包含n-1条边

算法结束

Prim算法（举例）

- 铺设一个连接各个城市的光纤通信网络（单位：万元）。

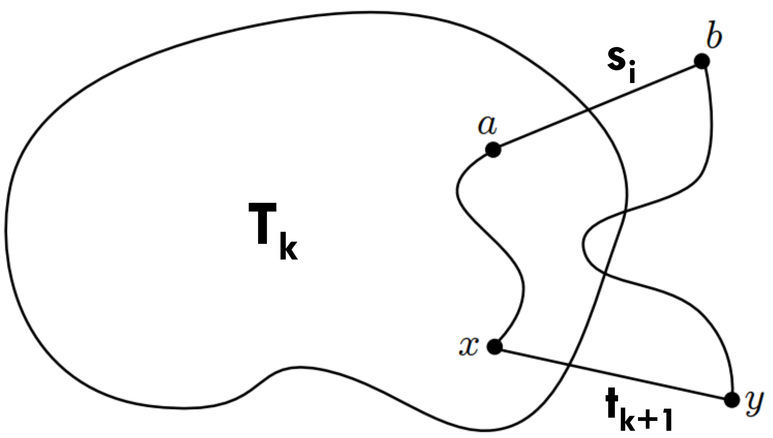


Prim 算法的正确性

令T为Prim算法的输出，并假设T按照边被选择的顺序包含了边$\{t_1,t_2,...,t_{n-1}\}$。定义$T_i=\{t_1,t_2,...,t_i\}$。只需证明T_i在一个MST中。

假设T_k在一个MST T'中，如果t_{k+1}不属于T'，则$T' + t_{k+1}$存在回路。

令s_i为回路上不在T_k的、且s_i的一个端点在T_k中的边。这也就意味着，当选择t_{k+1}时s_i也可选而没有被选，即t_{k+1}的权重不大于s_i的权重。那么$T' - s_i + t_{k+1}$为包含T_{k+1}的MST。



Kruskal算法（求最小生成树）

1: E={ }

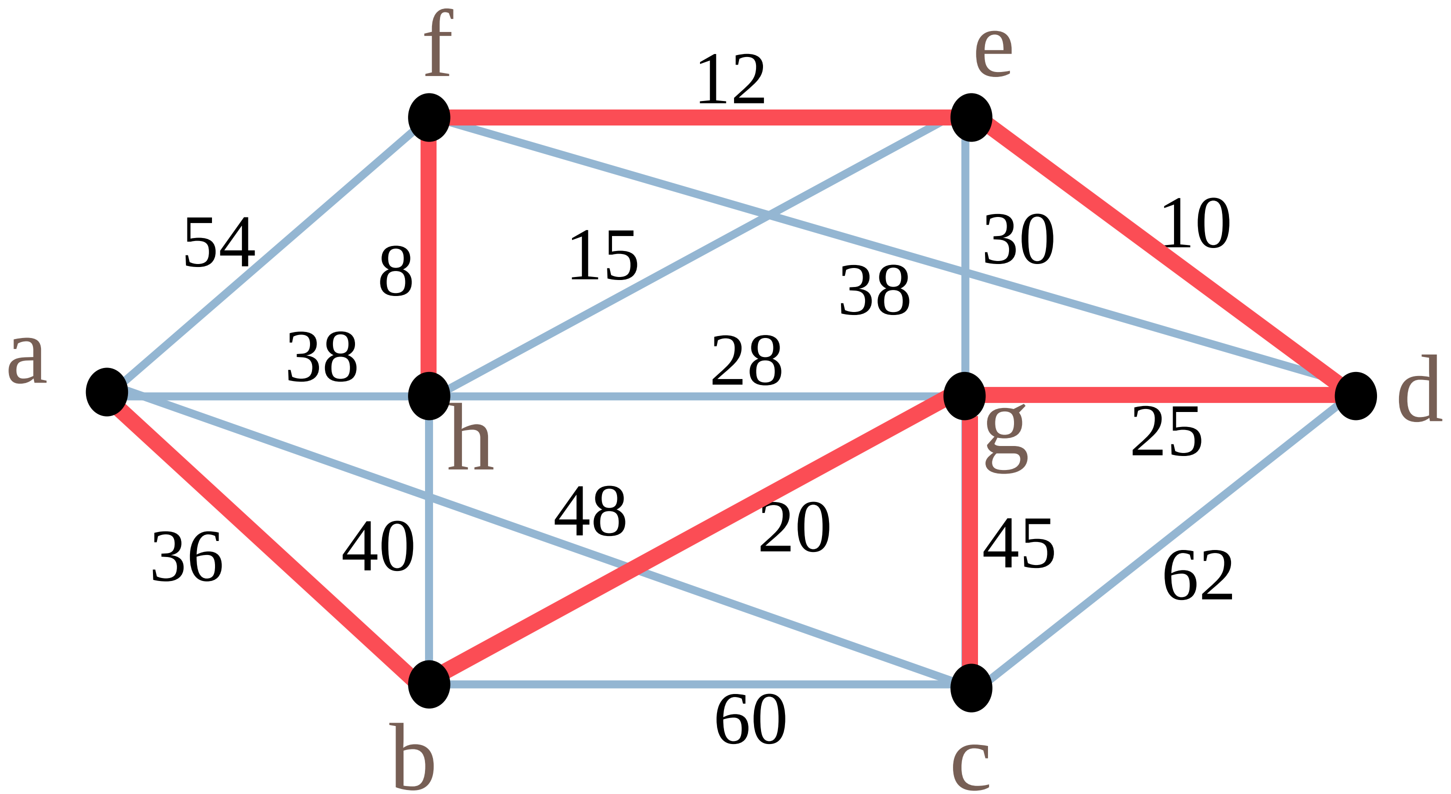
2: 从E以外选择不会与E中的边构成回路的权最小的边加入E

3: 重复第2步，直到E中包含n-1条边

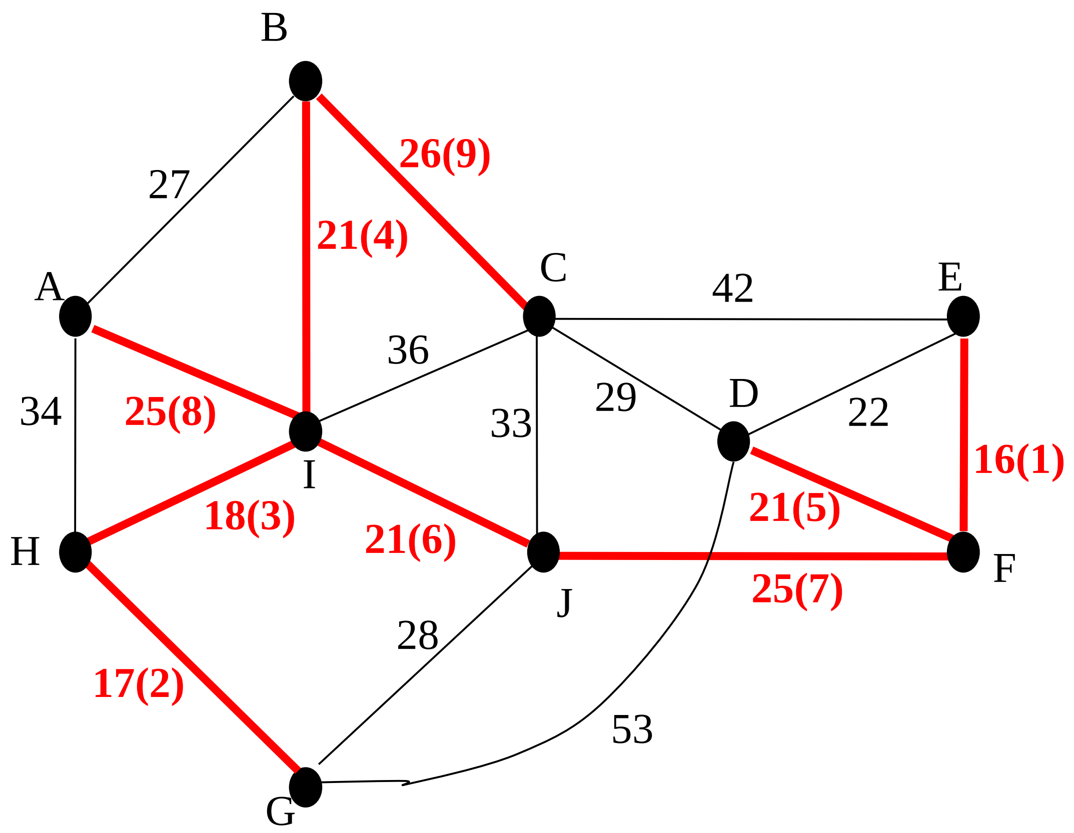
算法结束

Kruskal算法（举例）

- 铺设一个连接各个城市的光纤通信网络（单位：万元）。



Kruskal算法（举例）

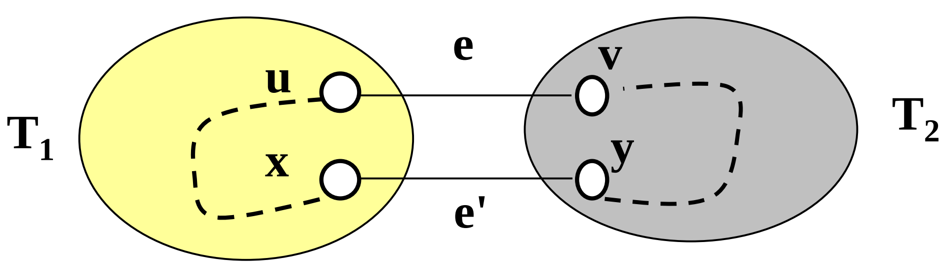


后面证明：Kruskal算法的正确性

引理（更换生成树的边）

- T与T'均是图G的生成树，若$e\in E_T$且$e\notin E_{T'}$，则必有$e'\in E_{T'}$, $e'\notin E_T$, 且T-{e}U{e'}和T'-{e'}U{e}均是G的生成树。
  - 设e=uv, T-{e}必含两个连通分支，设为T_1, T_2。因T'是连通图，T'中有uv-通路，其中必有一边满足其两个端点x,y分别在T_1, T_2中，设其为e'，显然T-{e}U{e'}是生成树。

  而T’-{e’}中x,y分属两个不同的连通分支，但在T*=T’-{e’}U{e}中，**xu-通路+e+vy通路**是一条xy-通路，因此T’-{e’}U{e}连通，从而T'-{e'}U{e}是生成树。



Kruskal算法的正确性

- 显然T是生成树。
- 按在算法中加边顺序，T中边是$e_1,e_2,\dots e_{k-1}, e_k,\dots e_{n-1}$。
- 假设T不是最小生成树。对于任意给定的一棵最小生成树T’, 存在唯一的k, 使得$e_k \notin E_{T'}$，且$e_i \in E_{T'}(1\le i<k)$. 设T’是这样的一棵最小生成树，使得上述的k达到最大。
- 根据前述引理，T’中存在边e’，e’不属于T, 使得T*=T’-{e’}U{e_k}也是生成树。e’∈T’与$e_1,e_2,\dots e_{k-1}$不会构成回路，因此w(e’)≥w(e_k). 所以w(T*)≤w(T’), 即T*也是最小生成树。但T*包含$e_1,e_2,\dots e_{k-1},e_k$，矛盾。

Generic Algorithm for MST Problem

Input: G: a connected, undirected graph

w: a function from V_G to the set of real number

Generic-MST(G,w)

$A \leftarrow 0$

while A does not form a spanning tree

  do find an edge (u,v) that is safe for A

  $A \leftarrow A \cup \{(u,v)\}$

return A

Output: a minimal spanning tree of G

“避圈法”与“破圈法”

- 上述算法都是贪心地增加不构成回路的边，以求得最优树，通常称为“避圈法”；
- 从另一个角度来考虑最优树问题，在原连通带权图G中逐步删除构成回路中权最大的边，最后剩下的无回路的子图为最优树。我们把这种方法称为“破圈法”。

作业

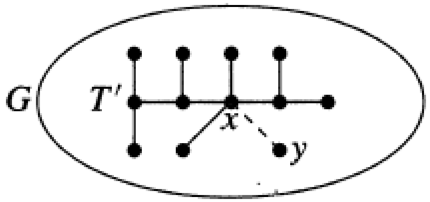
- 见课程网站

例题

- 证明：如果一个连通图的边权重互不相等，则其具有唯一的MST。
  - 假设G有两棵不同的MST T1与T2，其边的集合按照权值从小到大排列分别是$e_1,e_2,\ldots,e_m$和$e_1',e_2',\ldots,e_m'$。假设i是最小的、使得e_i和e_i'不相同的边。不失一般性，假设e_i的权值小于e_i'。
  - 考察T2+e_i，必然存在包含e_i的回路，并且这个回路中必然包含了不同于$e_1,e_2,\ldots,e_{i-1}$的在T2中的边。假设这条边是e'。根据假设可知：e'的权值大于e_i的权值。因此我们可以从T2中删除e'得到一棵更小的生成树。这和T2是MST树矛盾。

例题

- 假设树T有k条边，简单图G的最小度数为k。则G一定包含形如T的子图。
  - 证明：k=0显然成立。假设k<=n时成立。k=n+1时：
  - 从T中删除一个叶子节点v，T'=T-v。根据归纳假设，则G一定包含T'这个子图。令x为T中与v相邻的点，$d_G(x)$>=k且T'中除了x以外有k-1个点，所以x必定和T'以外的某个点y相邻。T'+xy边即为题意中的T。



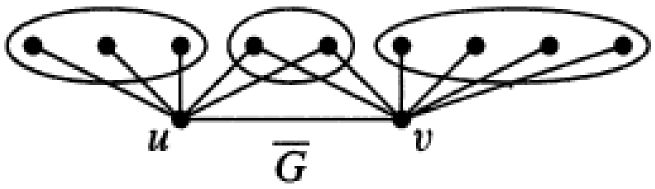
例题

- 图的直径指图中任意两点间最短距离的最大值。
- 某个节点u的离心率指u到其他节点最短距离的最大值。
- 图的半径指所有节点的离心率的最小值。
- 显然，图直径等于所有节点离心率的最大值。

- 图的中心指所有最小离心率节点的导出子图。
- 显然，图的中心是自身当且仅当直径等于半径。

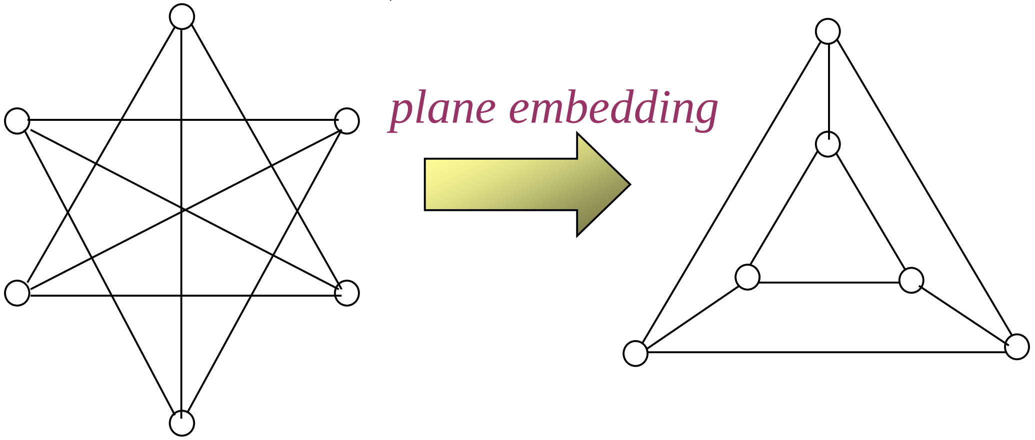
例题

- 证明：如果简单图G的直径大于2，则其补图直径小于4。
  - 证明：G的直径大于2，则必定存在两个不相邻的节点u和v，它们之间没有公共邻居。即任意一个节点x至少跟u和v其中的一个不相邻。即任意x在$\overline{G}$中至少跟u和v中的一个相邻。又因为uv边属于$\overline{G}$，则$\overline{G}$中任意的两点x和y通过uv边三跳可达。

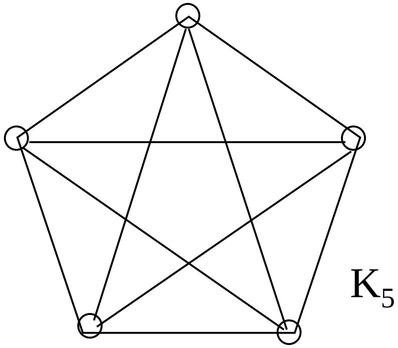


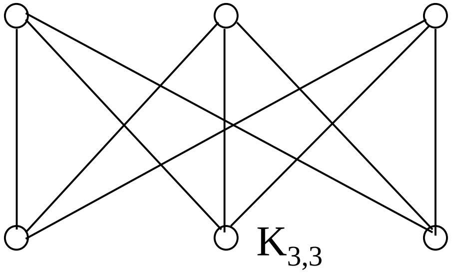
平面图

- 可以在一个平面上画出来，使得任意两条边不交叉

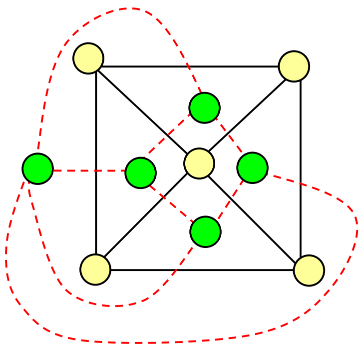


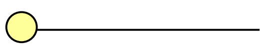
- 典型的非平面图





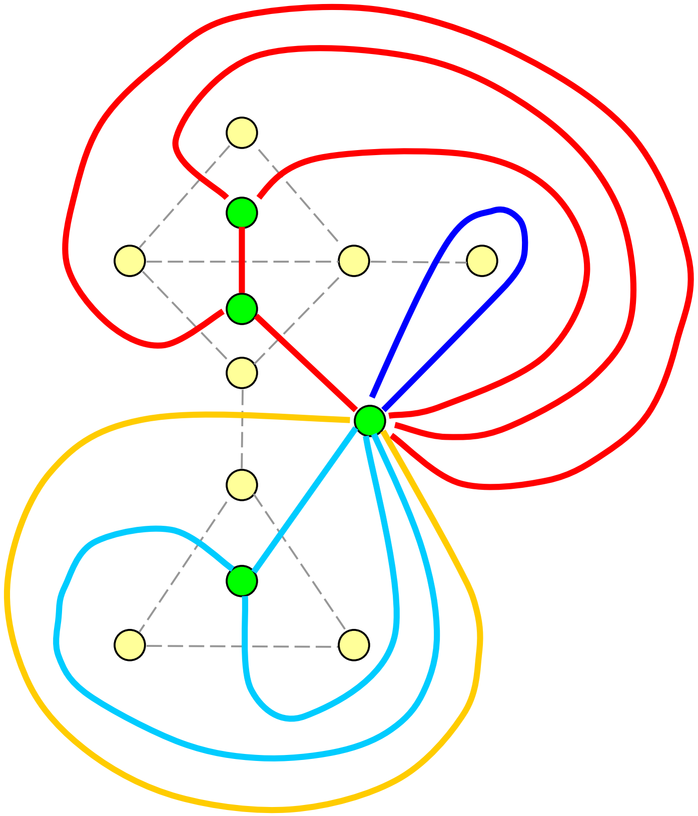
平面图的对偶图





Vertices and edges
in G

Vertices and edges
in the dual graph G*
of G



G平面图充要条件：G*平面图

欧拉公式：V-E+F=2