程序设计艺术与方法

第二讲　搜索专题

合肥工业大学计算机与信息学院　2015-9

掷色子

两小王和小刘玩掷色子的游戏，共有两枚色子一起掷出。若两枚色子的点数之和为8，则小王赢，若点数之和为9，则小刘赢。试判断他们两人谁胜的可能性大？

（枚举法，最简单的搜索）

三个小孩的年龄有多大

- 两个多年未见的朋友在街上偶遇，他们聊了很多事情，后来他们中的一个问道：“既然你是数学方面的教授，那你来算算这道题。我的三个儿子都在今天庆祝他们的生日！那么你能算出他们都有多大吗？”
- “好，”数学家回答说，“但是你得跟我讲讲他们的情况。”

三个小孩的年龄有多大

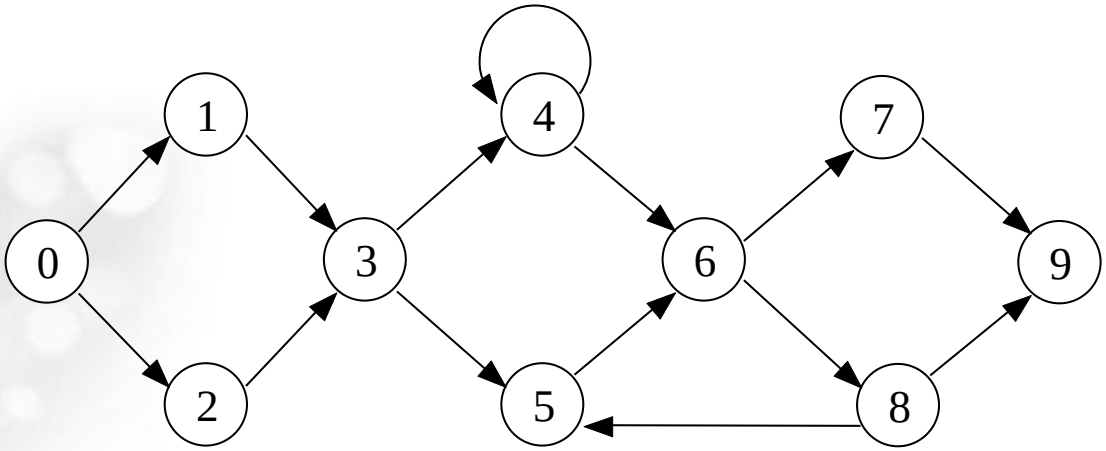
- “好的，我给你一些提示”那三个孩子的父亲说“他们三个的年龄之积是36”
- “很好，”数学家说“但我还需要更多的提示”
- “他们三个的年龄之和等于那栋房子的窗户数”这个父亲指者旁边的一栋房子说
- 数学家考虑了一下说“但是，我还要一点信息来解你的这个难题。”
- “我的大儿子的眼睛是蓝色的。”父亲说。
- “哦，够了。”数学家说道，并且给出了正确答案

解答

- 三个人的年龄之积是36 ⇒
  - **(1) 36,1,1 (2) 18,2,1 (3) 12,3,1 (4) 9,4,1**
    (5) 9,2,2 (6) 6,6,1 (7) 6,3,2 (8) 4,3,3
- 年龄之和等于那栋房子的窗户数
  - **和：38，21，16，14，13，13，11，10**
- 还要一点信息
  - **(5) 9,2,2 (6) 6,6,1**
- 大儿子的眼睛是蓝色的
  - **(5) 9,2,2**

街道赛跑

- 下图给出一个沿街赛跑路线。图中有许多点标以0, 1, …, N(下图N=9)，0为起点，N为终点，箭头表示可以前进的方向，运动员可以选择前进路线
- 任务A：输出所有必经点
- 任务B：输出所有分裂点



2.1 宽度优先搜索

2.2 最小生成树的形成和求解

2.3 深度优先搜索

2.4 更多的搜索

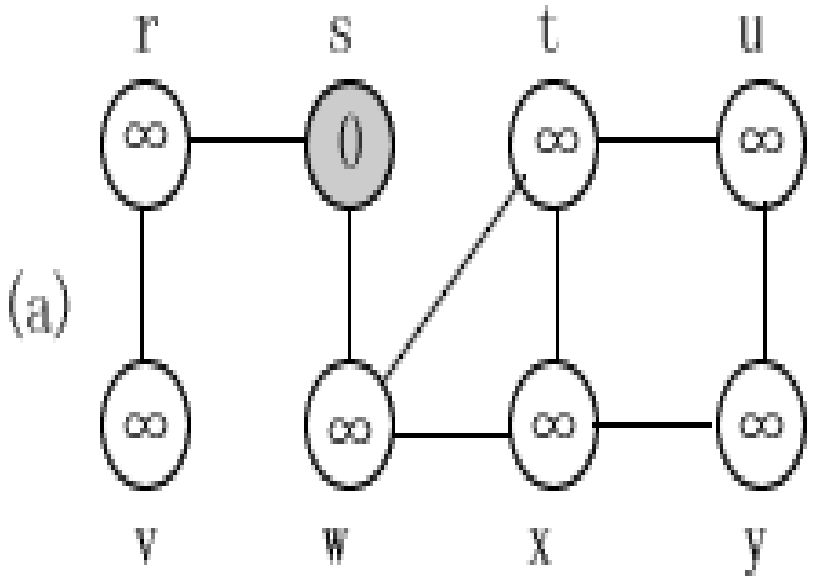
2.1 宽度优先搜索

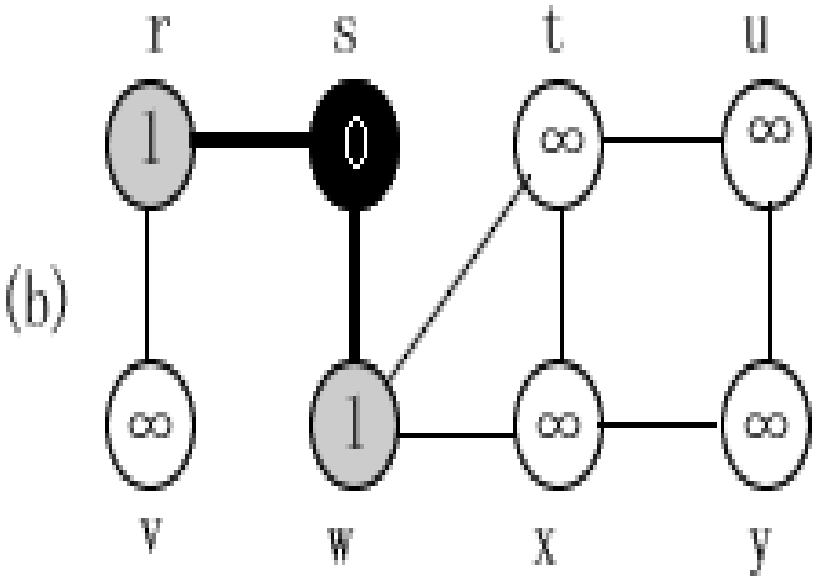
- 也叫广度优先搜索，是很多重要的图的算法的原型
- 基本思路是从源点出发，每一步找到当前点所有可能达到的所有情况（在图中就是相邻的点），逐层扩展，直到得到解为止。
- **Dijkstra**单源最短路径算法和**Prim**最小生成树算法都采用了和宽度优先搜索类似的思想
- 宽度优先搜索的全部运行时间为**O(V+E)**，宽度优先搜索的运行时间是图的邻接表大小的一个线性函数

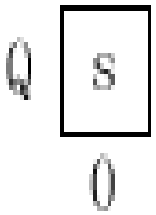
2.1 宽度优先搜索

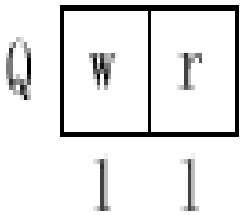
```
procedure BFS(G,S);
  begin
1.  for 每个节点u∈V[G]-{s} do
      begin
2.      color[u]←White;
3.      d[u]←∞;
4.      π[u]←NIL;
      end;
5.  color[s]←Gray;
6.  d[s]←0;
7.  π[s]←NIL;
8.  Q←{s}
9.  while Q≠φ do
     begin
10.     u←head[Q];
11.     for 每个节点v∈Adj[u] do
12.       if color[v]=White then
            begin
13.           color[v]←Gray;
14.           d[v]←d[u]+1;
15.           π[v]←u;
16.           Enqueue(Q,v);
            end;
17.     Dequeue(Q);
18.     color[u]←Black;
      end;
  end;
```

2.1 宽度优先搜索

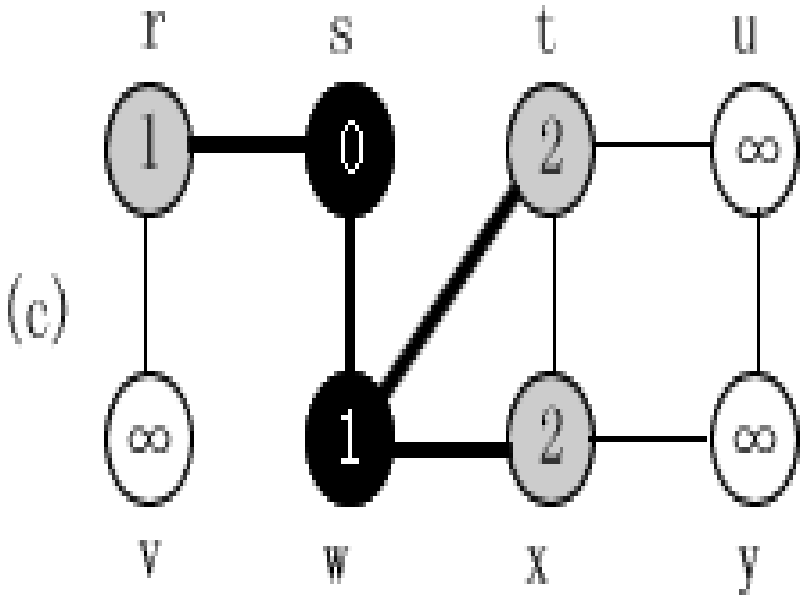




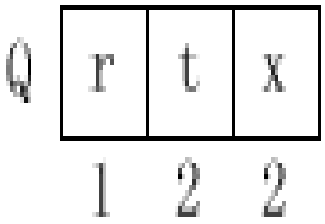


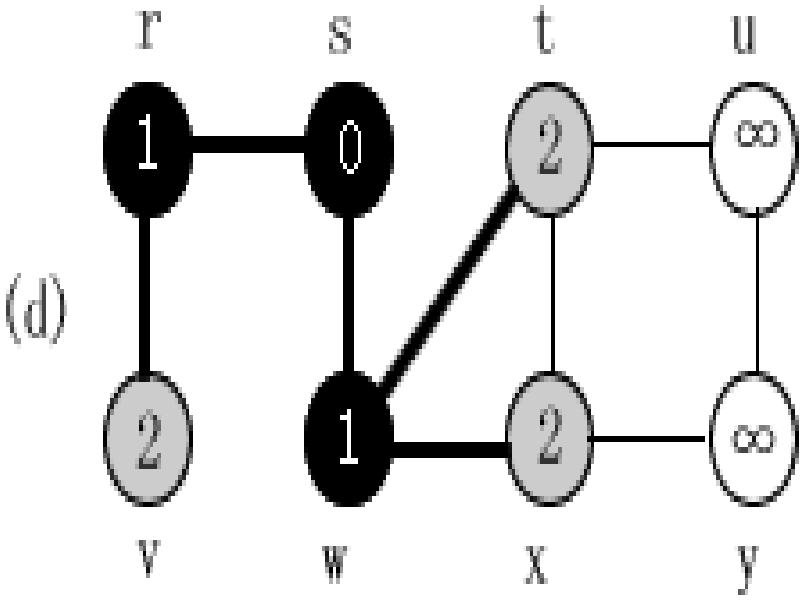


2.1 宽度优先搜索

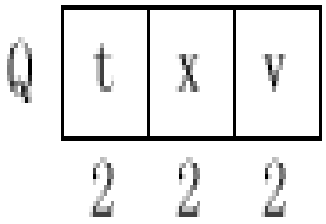


(c)

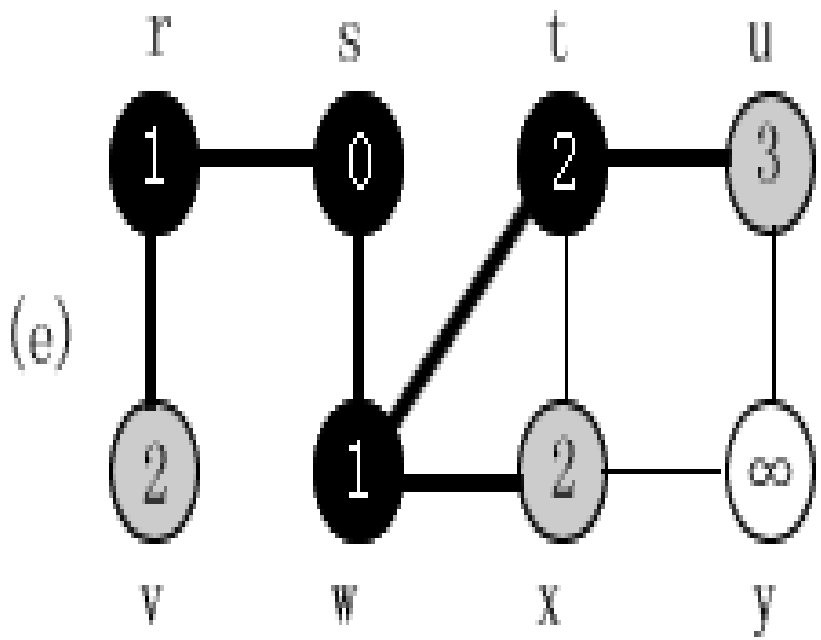




(d)



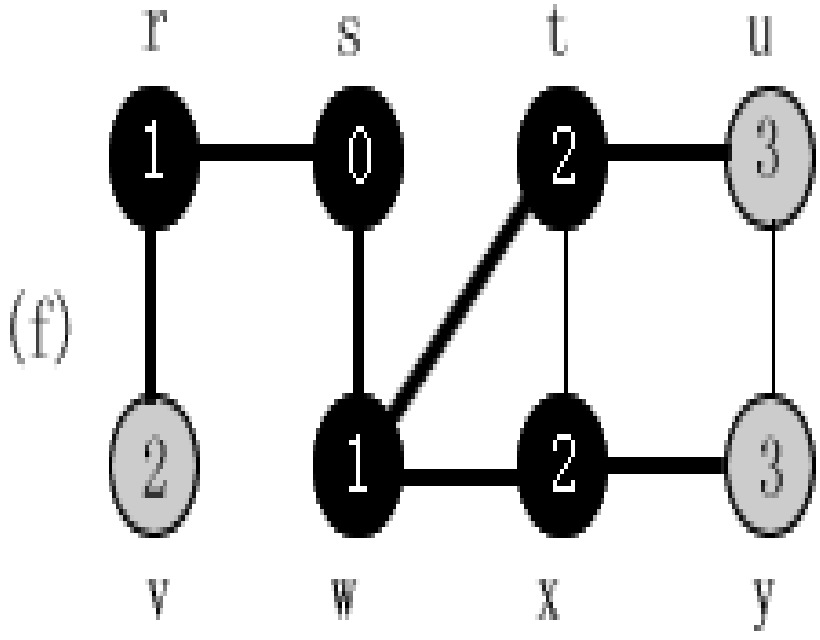
2.1 宽度优先搜索



(e)

Q | x | v | u

2 2 3

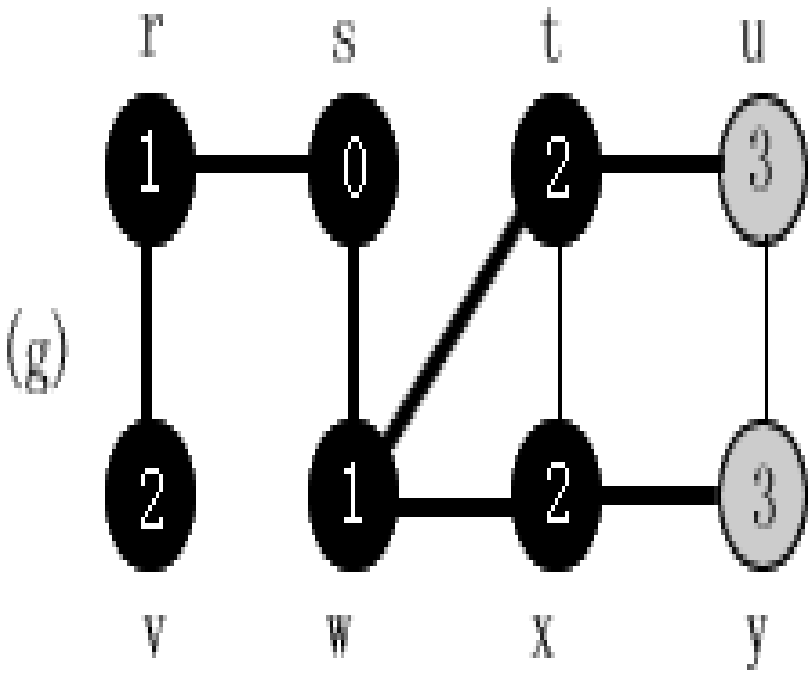


(f)

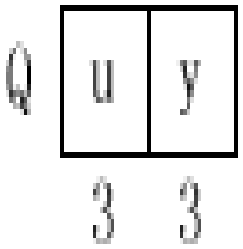
Q | v | u | y

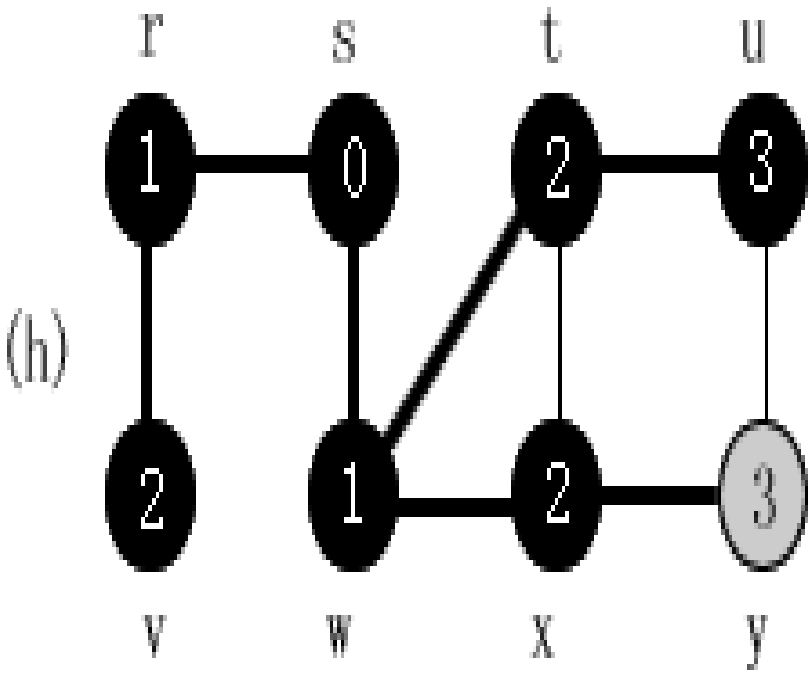
2 3 3

2.1 宽度优先搜索

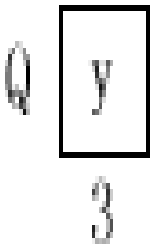


(g)

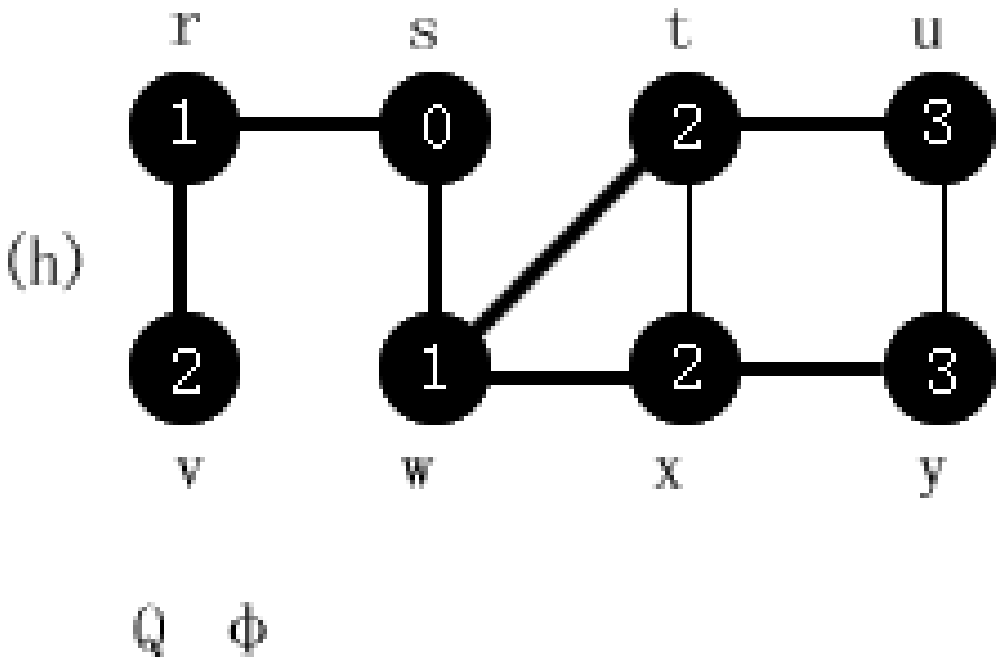




(h)



2.1 宽度优先搜索



宽度优先搜索的限制

- 引理1 设G=(V,E)是一个有向图或无向图，s∈V为G的任意一个结点，则对任意边(u,v)∈E, δ(s,v)≤δ(s,u)+1 (δ(s,v)为从顶点s到顶点v中具有最少边数的路径所包含的边数)
- 引理2 设G=(V,E)是一个有向或无向图，并假设算法BFS从G中一已知源结点s∈V开始执行，在执行终止时，对每个顶点v∈V，变量d[v]的值满足：d[v]≥δ(s,v)
- 引理3 假设过程BFS在图G=(V,E)之上的执行过程中，队列Q包含如下结点 $<v_1,v_2,...,v_r>$，其中v_1是队列Q的头，v_r是队列的尾，则$d[v_i]\le d[v_1]+1$且$d[v_i]\le d[v_{i+1}]$, i=1,2,..,r-1
- 引理4 当过程BFS应用于某一有向或无向图G=(V,E)时，该过程同时建立的π域满足条件：其先辈子图Gπ=(Vπ, Eπ)是一棵宽度优先树

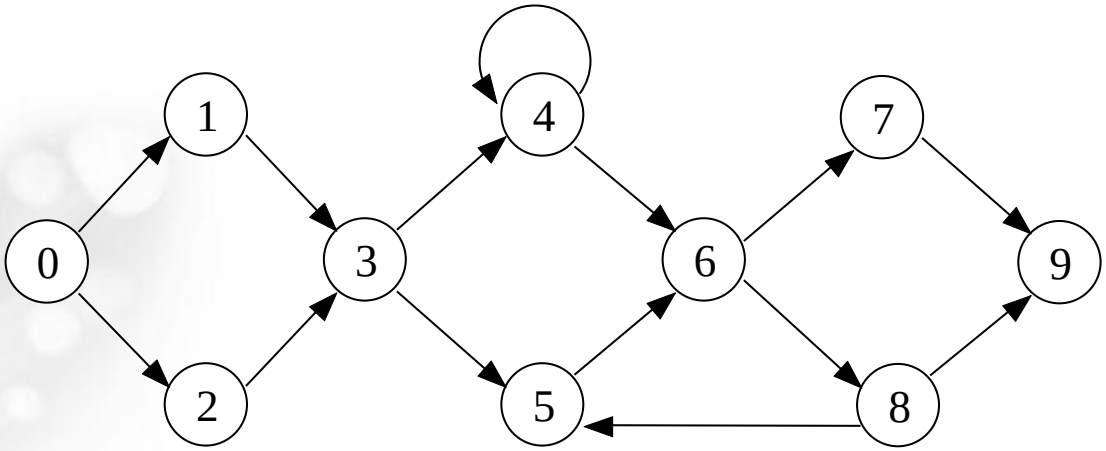
- 定理一 （宽度优先搜索的正确性）

例1

- 给定一个迷宫的地图，大小为n * m，0表示空地，1表示墙壁（不能走），2表示起点，3表示终点，问能否从起点到达终点，若能到达，同时给出起点与终点的最短距离。

街道赛跑（例2）

- 下图给出一个沿街赛跑路线。图中有许多点标以0, 1, …, N(下图N=9)，0为起点，N为终点，箭头表示可以前进的方向，运动员可以选择前进路线
- 任务A：输出所有必经点
- 任务B：输出所有分裂点



2.1 宽度优先搜索

2.2 最小生成树的形成和求解

2.3 深度优先搜索

2.4 更多的搜索

2.2 最小生成树的形成和求解

- 已知一无向连通图G=(V,E)，其加权函数为w:E→R，我们希望找到图G的最小生成树

- Prim算法和 Kruskal算法都运用了贪心方法，但在如何运用贪心法上却有所不同

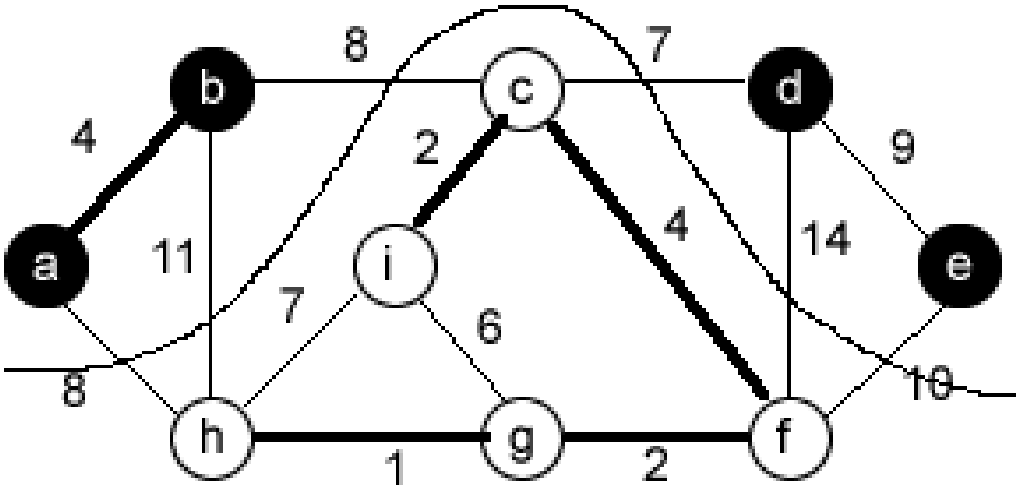
- 一个通用算法：

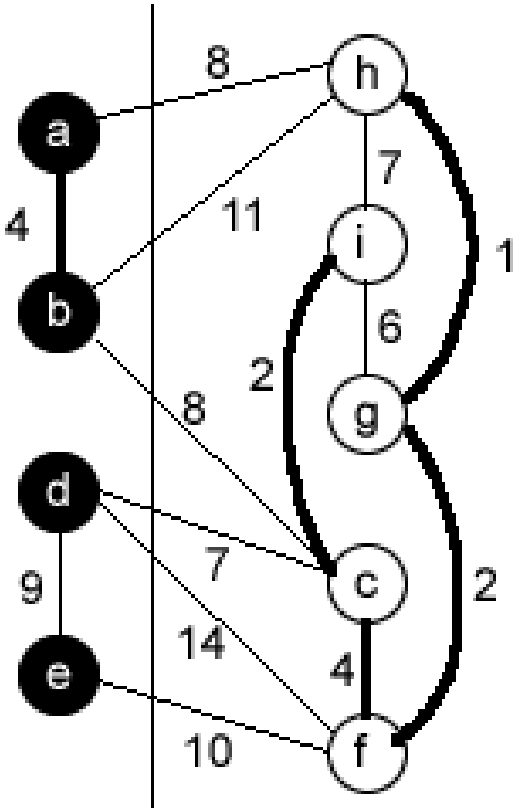
```
GENERNIC-MIT(G,W)
1. A←Φ
2. while A没有形成一棵生成树
3    do  找出A的一条安全边(u,v);
4.       A←A∪{(u,v)};
5. return A
```

2.2 最小生成树的形成和求解

- 几个概念
  - 割：图G=(V,E)的割(S,V-S)是V的一个分划
  - 当一条边(u, v) ∈E的一个端点属于S而另一端点属于V-S，则我们说边(u,v)通过割(S,V-S)
  - 若集合A中没有边通过割，则我们说割不妨害边的集合A (安全的)
  - 如果某边是通过割的具有最小权值的边，则称该边为通过割的一条轻边

2.2 最小生成树的形成和求解





- 集合S中的结点为黑色结点
- V-S中的那些结点为白色结点
- 连接白色和黑色结点的那些边为通过该割的边

-边(d,c)为通过该割的唯一一条轻边

- 子集A包含阴影覆盖的那些边，注意，由于A中没有边通过割，所以割(S,V-S)不妨害A

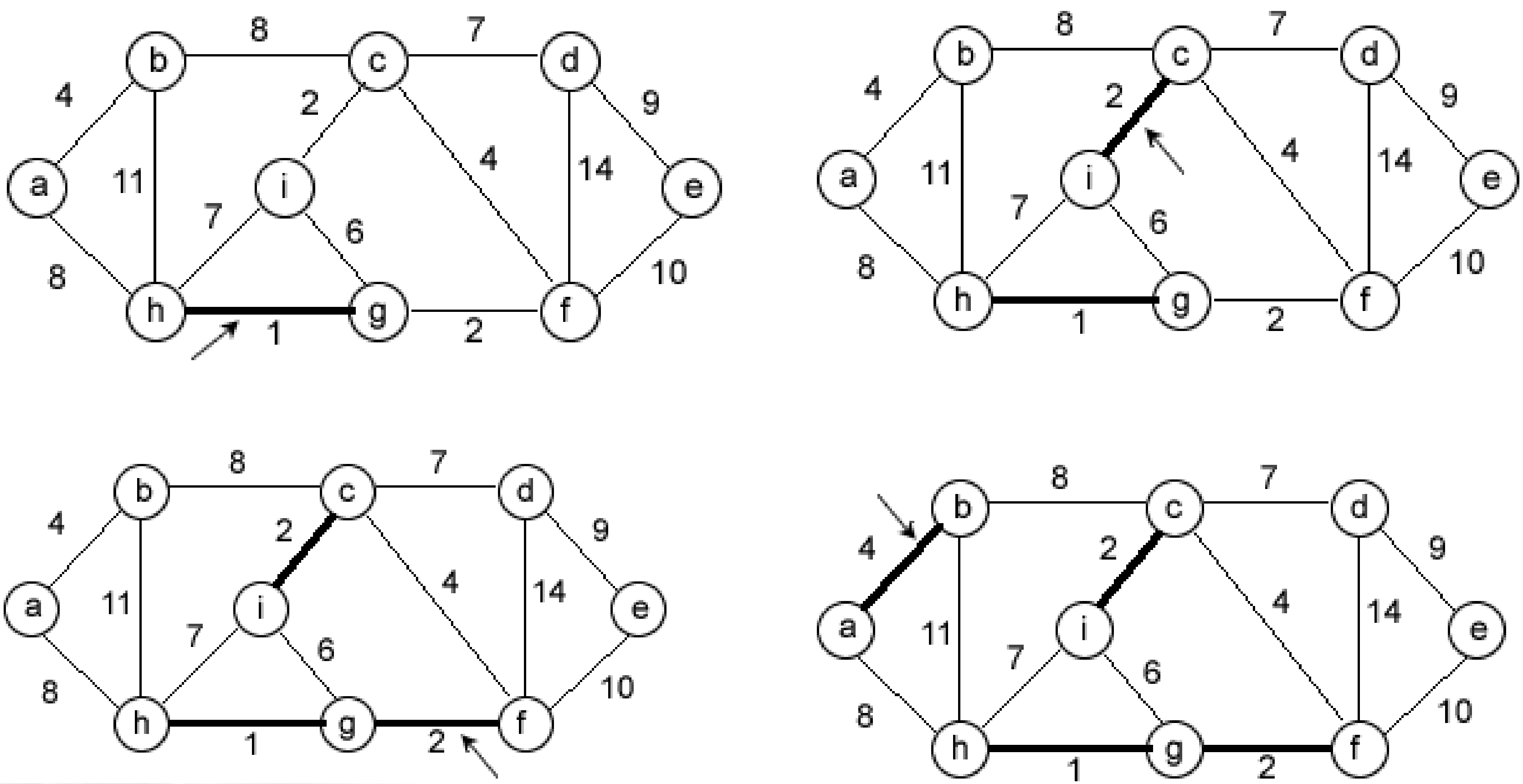
定理1：安全边的规则

- 设图G(V,E)是一无向连通图，且在E上定义了相应的实数值加权函数w，设A是E的一个子集且包含于G的某个最小生成树中，割(S,V-S)是G的不妨害A的任意割且边(u,v)是穿过割(S,V-S)的一条轻边，则边(u,v)对集合A是安全的

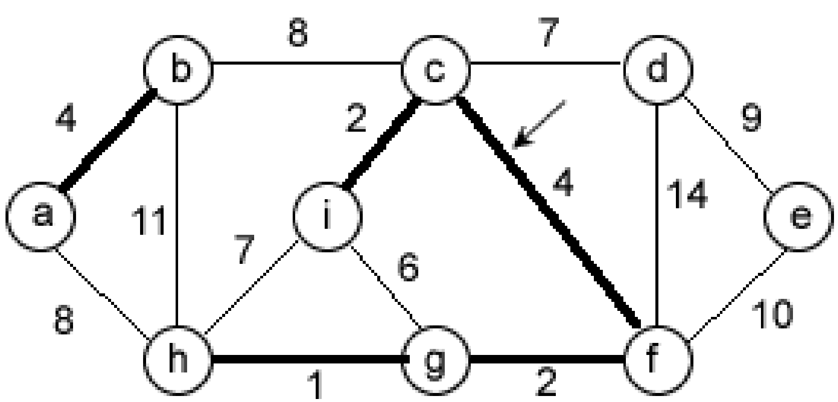
- 推论2 （Kruskal算法原理）
- 设G=(V,E)是一无向连通图，且在E上定义了相应的实数值加权函数 w，设A是E的子集且包含于G的某最小生成树中，C为森林G_A=(V,A)中的连通支(树)。 若边(u,v)是连接C和G_A中其他某连通支的一轻边，则边(u,v)对集合A是安全的

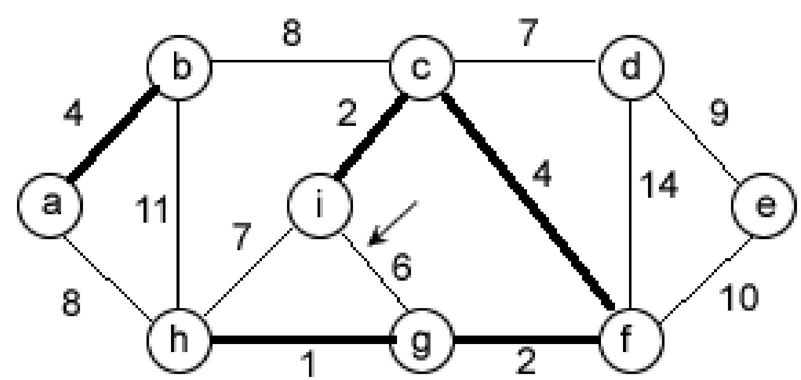
2.2 最小生成树的形成和求解

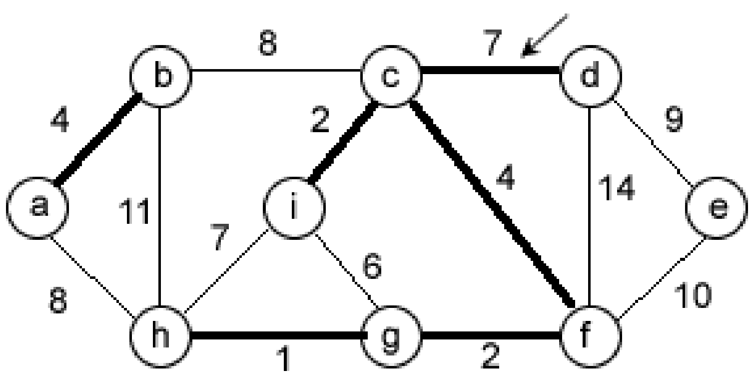
Kruskal算法

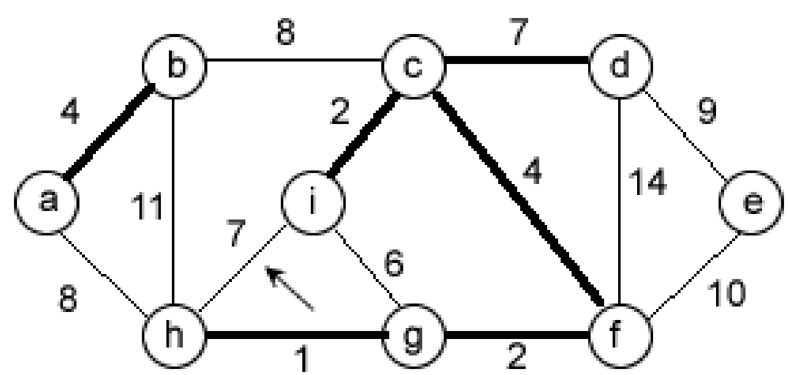


Kruskal算法


a
b
c
d
e
f
g
h
i
4
8
7
2
9
11
7
4
14
6
8
10
1
2

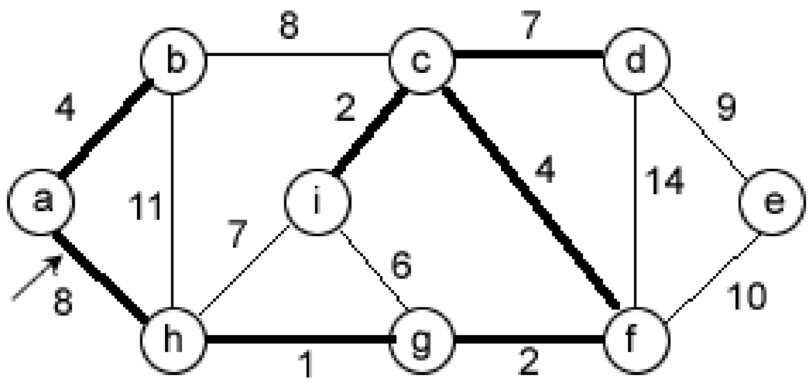

a
b
c
d
e
f
g
h
i
4
8
7
2
9
11
7
4
14
6
8
10
1
2

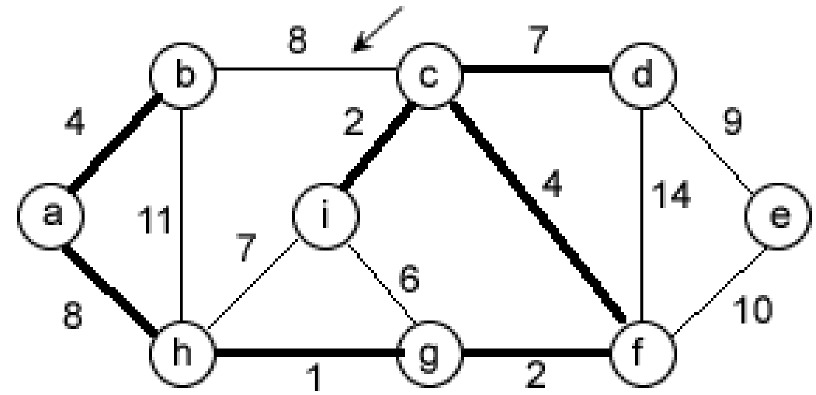

a
b
c
d
e
f
g
h
i
4
8
7
2
9
11
7
4
14
6
8
10
1
2

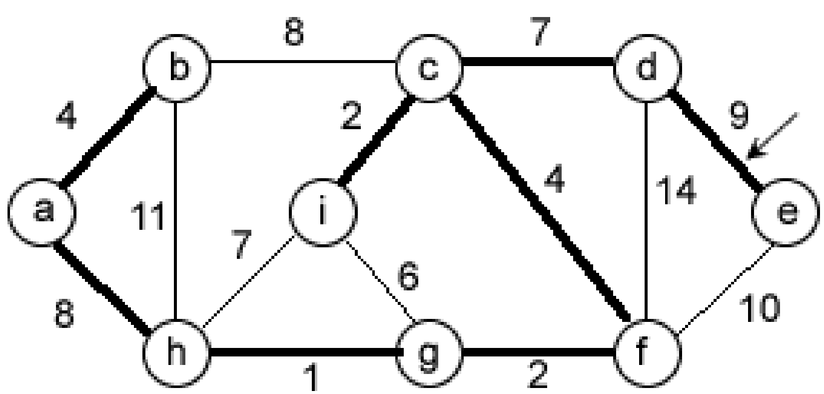

a
b
c
d
e
f
g
h
i
4
8
7
2
9
11
7
4
14
6
8
10
1
2

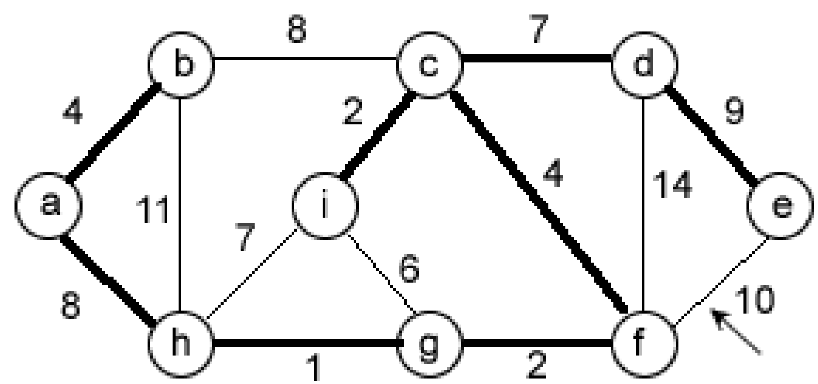
2.2 最小生成树的形成和求解

Kruskal算法



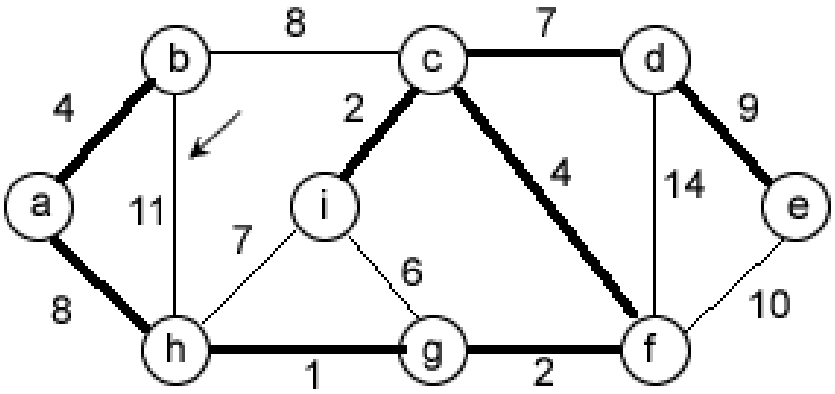


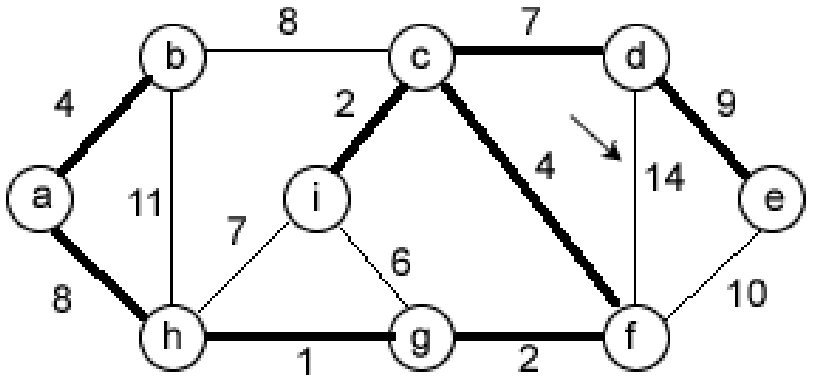




2.2 最小生成树的形成和求解

Kruskal算法



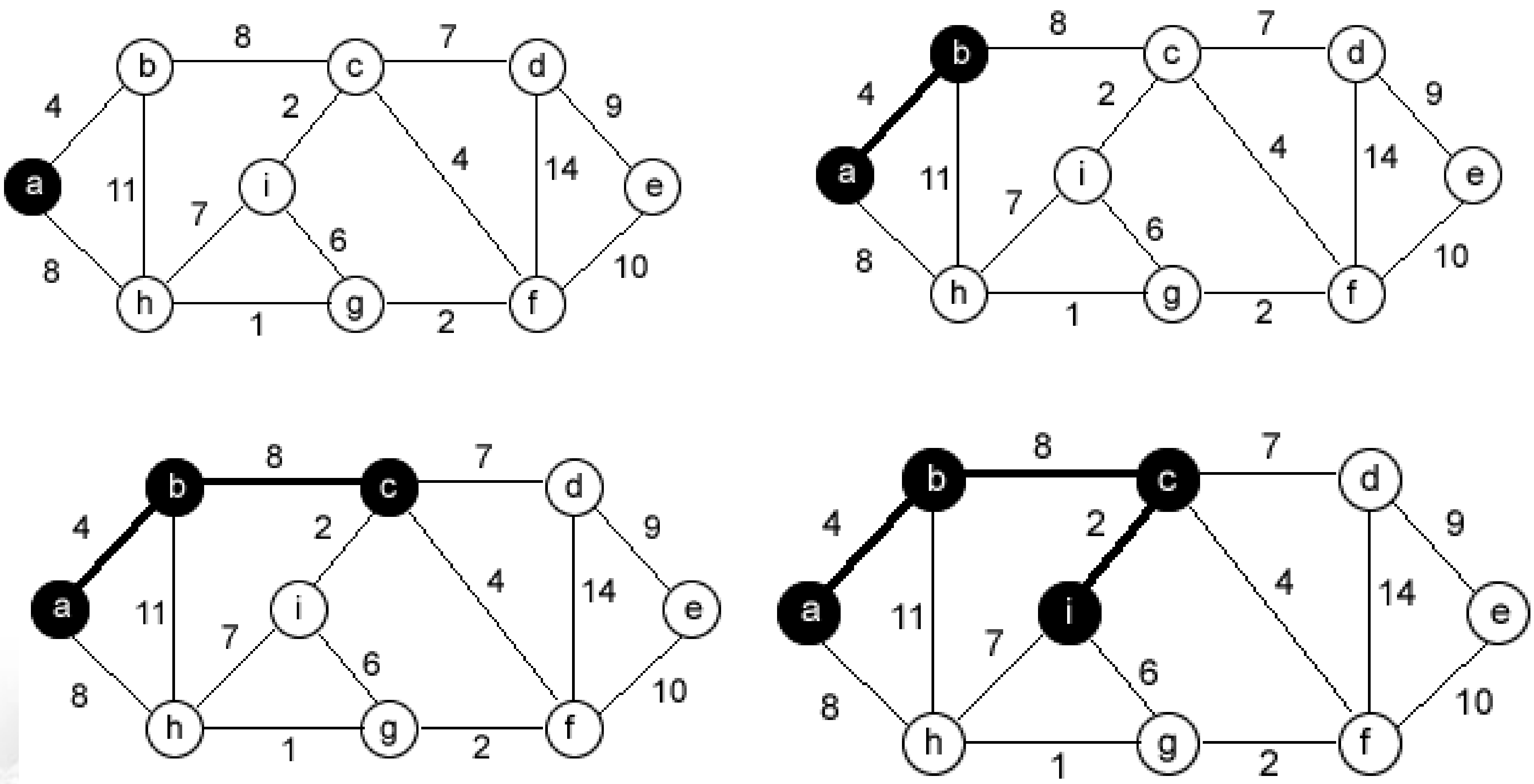


例3

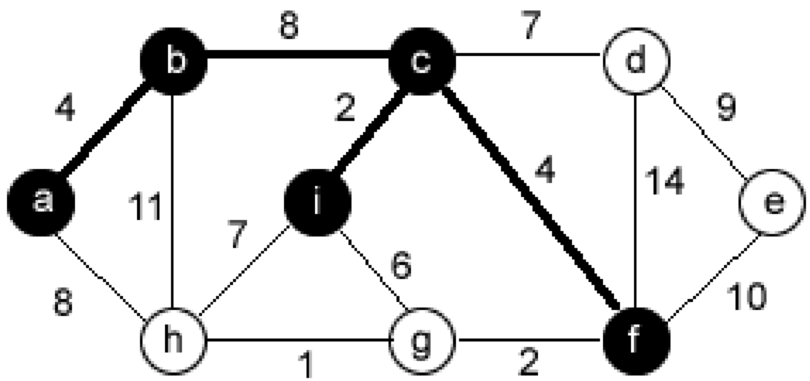
- 给定n个村子和m条道路，村子原本相互之间是没有道路连接的，现需要将村子与村子之间相互链接起来，再给定m条道路，表示从u到v修建一条道路需要w的花费，问若要将村子之间相互连接起来，最小花费是多少？（输入数据保证存在可行解）

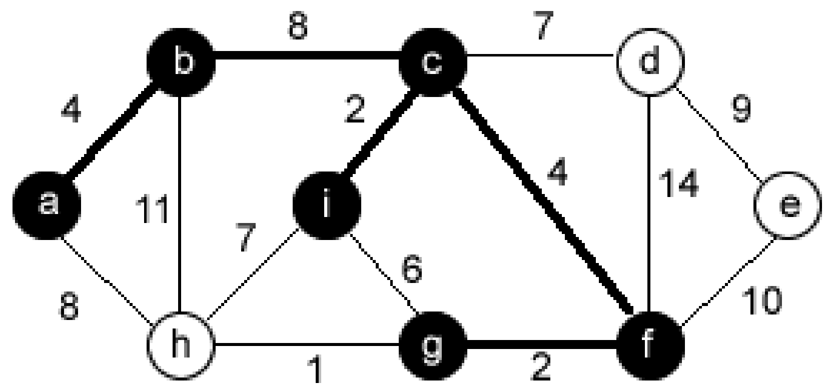
2.2 最小生成树的形成和求解

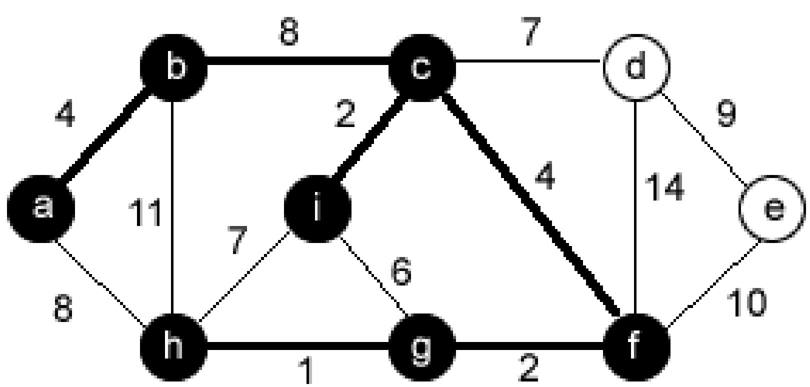
Prim算法

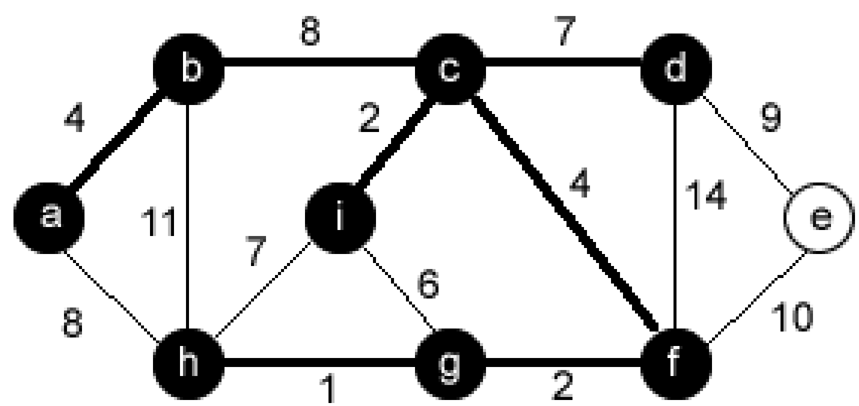


Prim算法

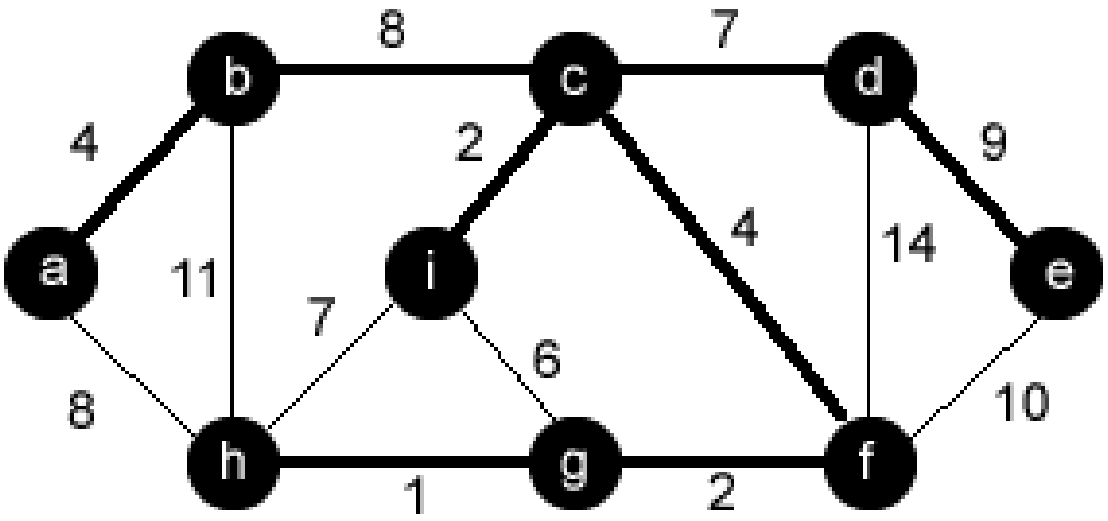

a b c d e f g h i
8 7 4 2 9 4 14 11 7 6 8 10 1 2


a b c d e f g h i
8 7 4 2 9 4 14 11 7 6 8 10 1 2


a b c d e f g h i
8 7 4 2 9 4 14 11 7 6 8 10 1 2


a b c d e f g h i
8 7 4 2 9 4 14 11 7 6 8 10 1 2

Prim算法



2.1 宽度优先搜索

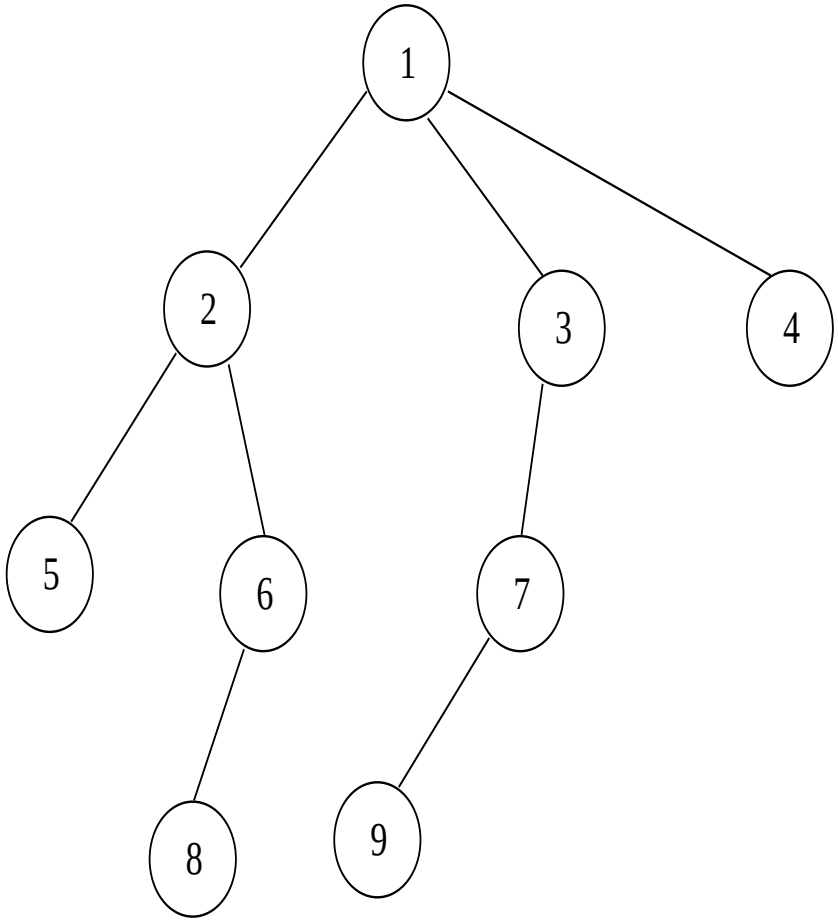
2.2 最小生成树的形成和求解

2.3 深度优先搜索

2.4 更多的搜索

另一道题目

- 一颗具有n（n<=10 0000）个节点的树，节点编号1~n
- 给定m（m<=1 0000）个节点对（i，j），输出满足i为j后裔或者j为i后裔的节点对总个数



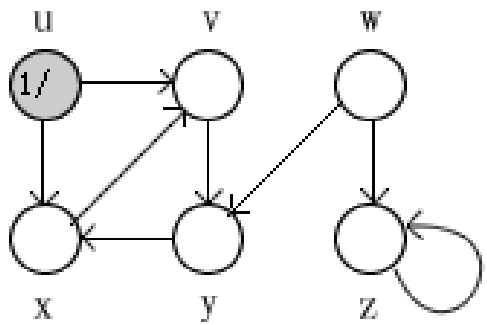
2.3 深度优先搜索

- $G_\pi=(V,E_\pi),E_\pi=\{(\pi[v],v)\in E:v\in V\wedge\pi[v]\neq NIL\}$
- 深度优先搜索形成一个由数个深度优先树组成的深度优先森林
- E_π中的边称为树枝。和宽度优先搜索类似，深度优先在搜索过程中也为结点着色以表示结点的状态
- 每个顶点开始均为白色，搜索中被发现时置为灰色，结束时又被置成黑色(即当其邻接表被完全检索之后)
- 这一技巧可以保证每一顶点搜索结束时只存在于一棵深度优先树上，因此这些树都是分离的

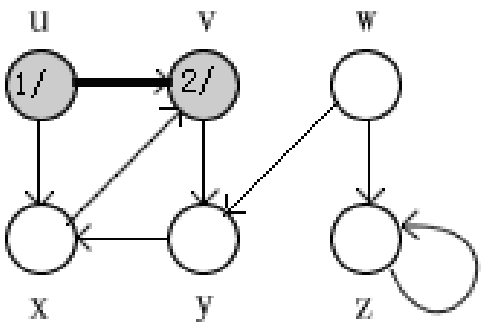
2.3 深度优先搜索

- 深度优先搜索同时为每个结点加盖时间戳
- 每个结点v有两个时间戳：
  - 当结点v第一次被发现(并置成灰色)时记录下第一个时间戳d[v]，
  - 当结束检查v的邻接表时(并置v为黑色)记录下第二个时间截f[v]
- 许多图的算法中都用到时间戳，他们对推算深度优先搜索进行情况是很有帮助的

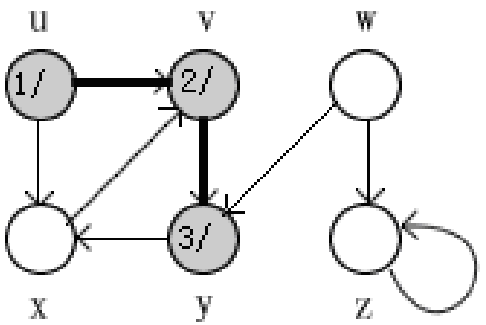
2.3 深度优先搜索



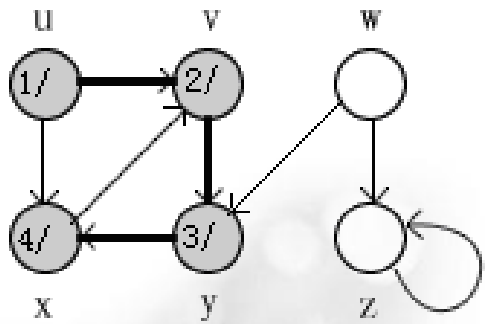
(a)



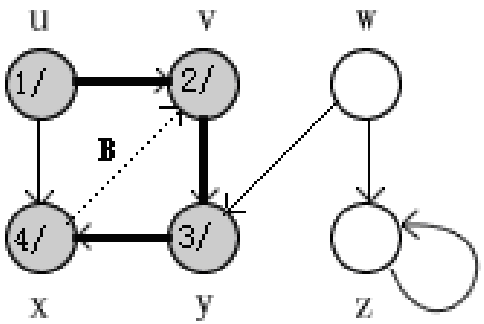
(b)



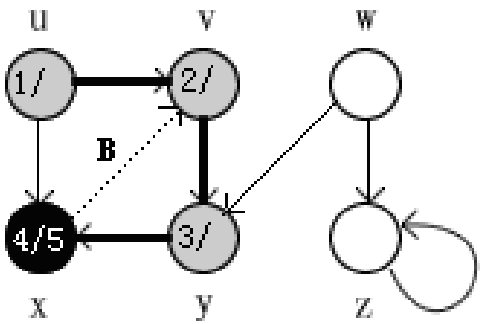
(c)



(d)

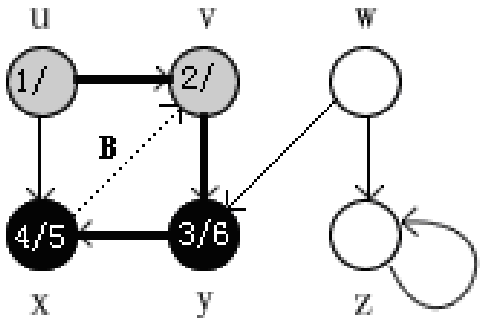


(e)

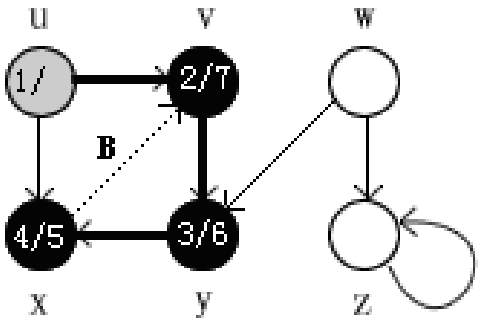


(f)

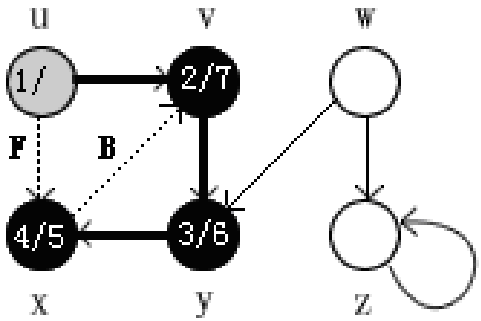
2.3 深度优先搜索



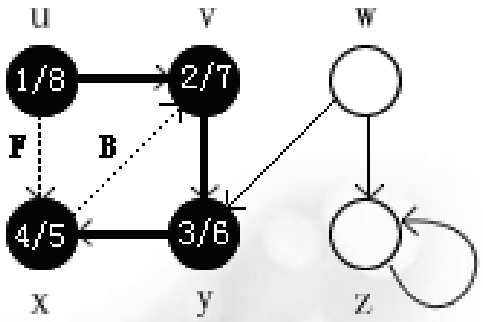
(g)



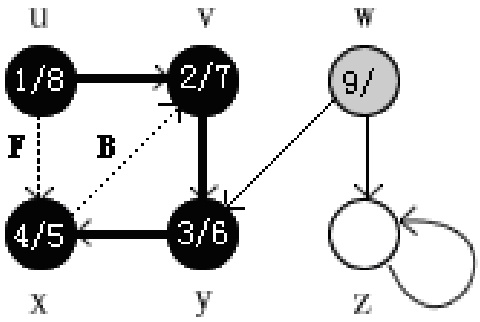
(h)



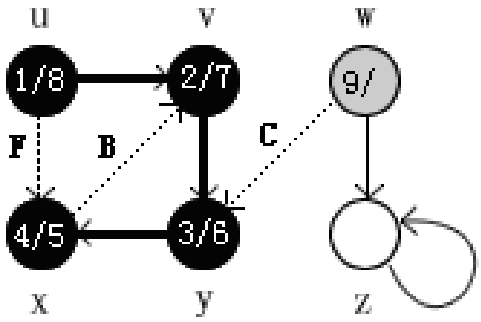
(i)



(j)

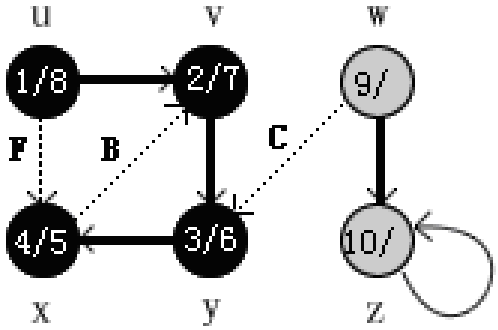


(k)

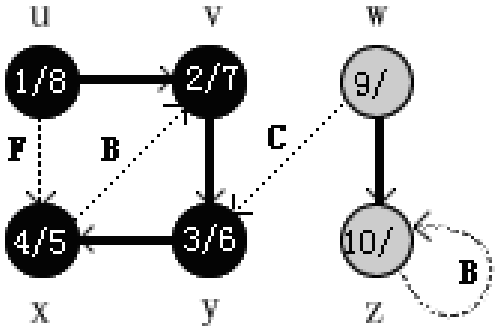


(l)

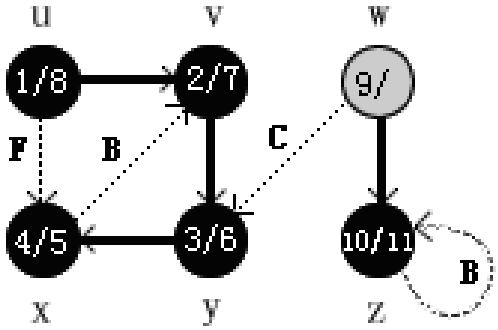
2.3 深度优先搜索



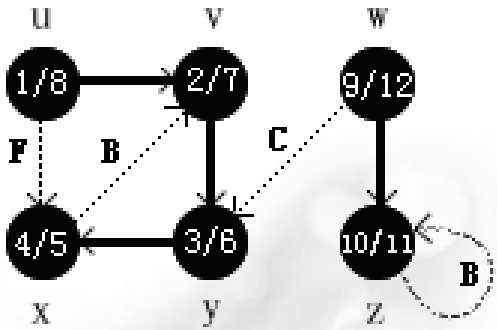
(m)



(n)



(o)



(p)

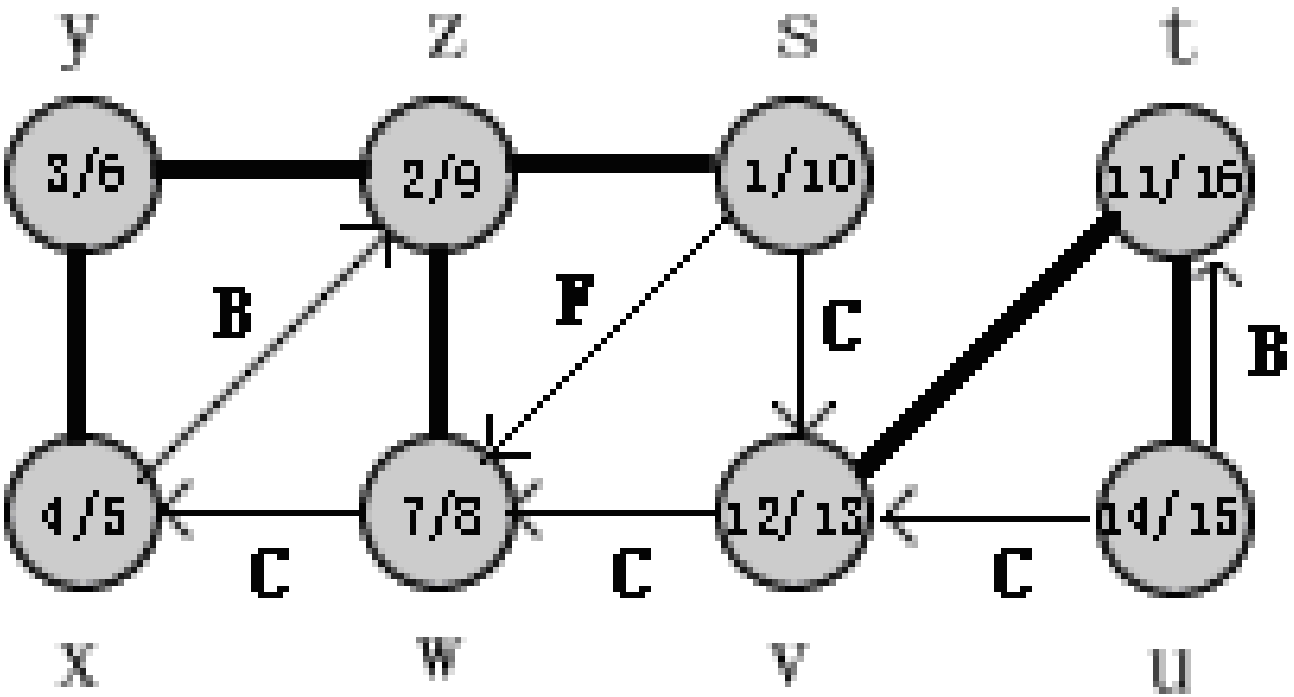
2.3 深度优先搜索

```
procedure DFS(G);
  begin
1   for 每个顶点u∈V[G] do
     begin
2      color[u]←White;
3      π[u]←NIL;
     end;
4   time←0;
5   for 每个顶点u∈V[G] do
6    if color[u]=White
7       then DFS_Visit(G,u);
end;
```

```
procedure DFS_Visit(G,u);
 begin
1   color[u]←Gray;  Δ白色结点u
2   d[u]←time←time+1;
3   for 每个顶点v∈Adj[u] do
                        Δ探寻边(u,v)
4     if color[v]=White
       then begin
5            π[v]←u;
6            DFS_Visit(G,v);
          end;
7   color[u]←Black;  Δ完成置u黑色
8   f[u]←time←time+1;
end;
```

深度优先搜索的性质

- 深度优先搜索可以获得有关图的结构的大量信息



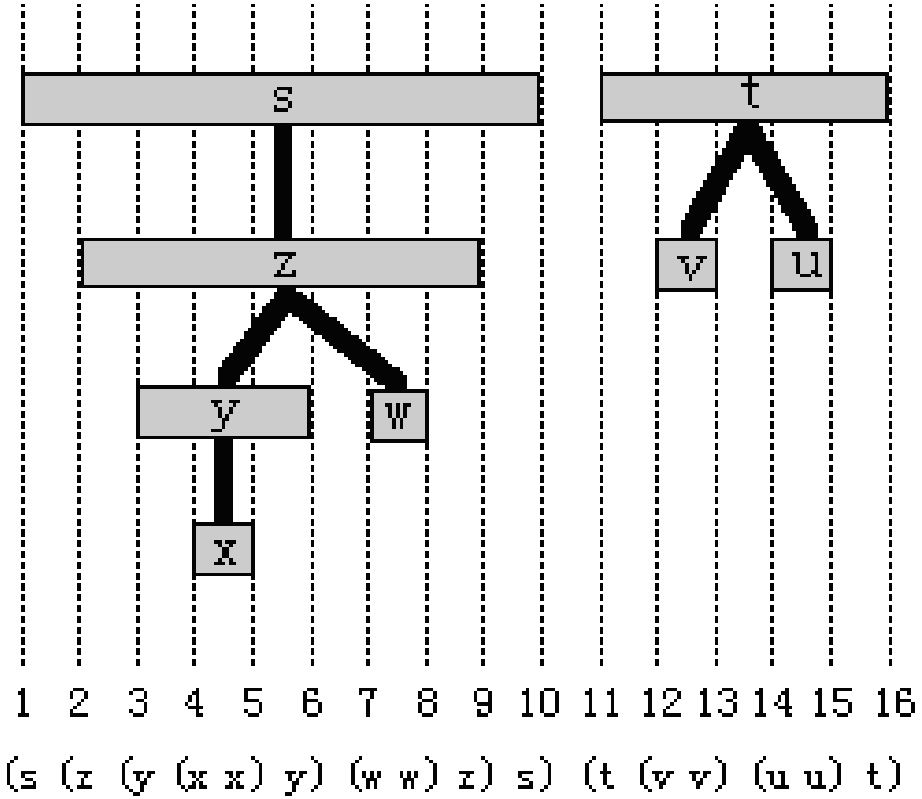
- 深度优先搜索的性质
  - 定理一：括号定理
  - 推论一：后裔区间的嵌入
  - 定理二：白色路径定理

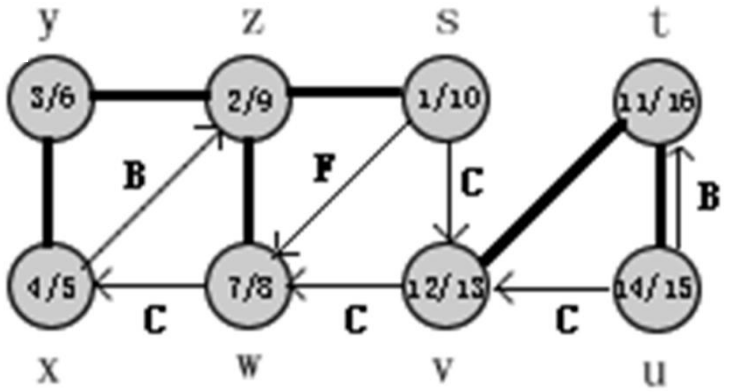
定理1 括号定理

- **在对有向图或无向图G=(V,E)的任何深度优先搜索中，对于图中任意两结点u和v，下述三个条件中有且仅有一条成立：**
  - 区间[d[u],f[u]]和区间[d[v],f[v]]是完全分离的；
  - 区间[d[u],f[u]]完全包含于区间[d[v],f[v]]中且在深度优先树中u是v的后裔；
  - 区间[d[v],f[v]]完全包含于区间[d[u],f[u]]中且在深度优先树中v是u的后裔

推论1 后裔区间的嵌入

- **在有向或无向图G的深度优先森林中，结节v是结点u的后裔当且仅当d[u]<d[v]<f[v]<f[u]**


s
t
z
v
u
y
w
x
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
(s (z (y (x x) y) (w w) z) s) (t (v v) (u u) t)


y
z
s
t
3/6
2/9
1/10
11/16
B
F
C
B
4/5
7/8
12/13
14/15
C
C
C
x
w
v
u

定理2 白色路径定理

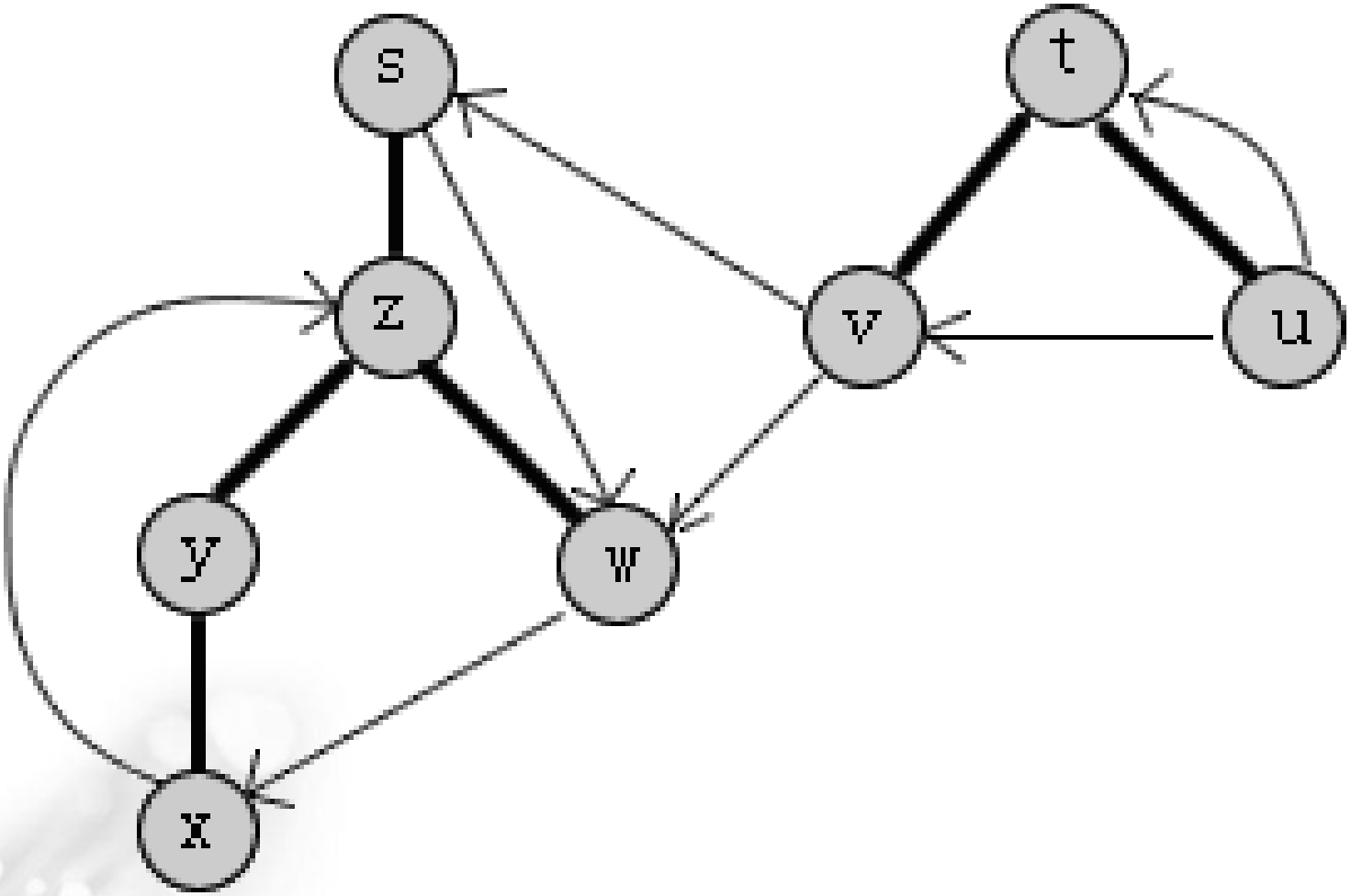
- 在一个有向或无向图G=(V,E)的深度优先森林中，结点v是结点u的后裔当且仅当在搜索发现u的时刻d[u]，从结点u出发经一条仅由白色结点组成的路径可达v

证明:

- →：假设v是u的后裔，w是深度优先树中u和v之间的通路上的任意结点，则w必然是u的后裔，由推论1可知d[u]<d[w]，因此在时刻d[u]，w应为白色
- ←：设在时刻d[u]，从u到v有一条仅由白色结点组成的通路，但在深度优先树中v还没有成为u的后裔。不失一般性，我们假定该通路上的其他顶点都是u的后裔(否则可设v是该通路中最接近u的结点，且不为u的后裔)，设w为该通路上v的祖先，使w是u的后裔(实际上w和u可以是同一个结点）。根据推论1得f[w]≤f[u]，因为v∈Adj[w]，对DFS_Visit(w)的调用保证完成w之前先完成v，因此f[v]<f[w]≤f[u]。因为在时刻d[u]结点v为白色，所以有d[u]<d[v]。由推论1可知在深度优先树中v必然是u的后裔

边的分类

- **根据在图G上进行深度优先搜索所产生的深度优先森林G，把图的边分为四种类型：**
  - **树枝**，是深度优先森林Gπ中的边，如果结点v是在探寻边(u,v)时第一次被发现，那么边(u,v)就是一个树枝
  - **反向边**，是深度优先树中连结结点u到它的祖先v的那些边，环也被认为是反向边
  - **正向边**，是指深度优先树中连接顶点u到它的后裔的非树枝的边
  - **交叉边**，是指所有其他类型的边，它们可以连结同一棵深度优先树中的两个结点，只要一结点不是另一结点的祖先，也可以连结分属两棵深度优先树的结点


s
t
z
v
u
y
w
x

- 可以对算法DFS进行一些修改，使之遇到边时能对其进行分类：
  - 算法的核心思想在于可以根据第一次被探寻的边所到达的结点v的颜色来对该边(u,v)进行分类(但正向边和交叉边不能用颜色区分出)
    - 白色表明它是树枝。
    - 灰色说明它是反向边。
    - 黑色说明它是正向边或交叉边
      - 如果d[u]<d[v]，则边(u,v)就是正向边
      - 若d[u]>d[v]，则(u,v)便是交叉边
- 定理3
  - 在对无向图G进行深度优先搜索的过程中，G的每条边要么是树枝要么是反向边

例5

- 给定一个迷宫的地图，大小为n * m，0表示空地，1表示墙壁（不能走），2表示起点，3表示终点，问能否从起点到达终点。

2.1 宽度优先搜索

2.2 最小生成树的形成和求解

2.3 深度优先搜索

2.4 更多的搜索

2.4 更多的搜索

- 搜索的进一步扩充
  - 递归前预处理；
  - 减少搜索范围；
  - 记忆化搜索；
  - 寻找解析式或贪心策略；
  - 搜索剪枝、二分搜索、参数搜索……

例6

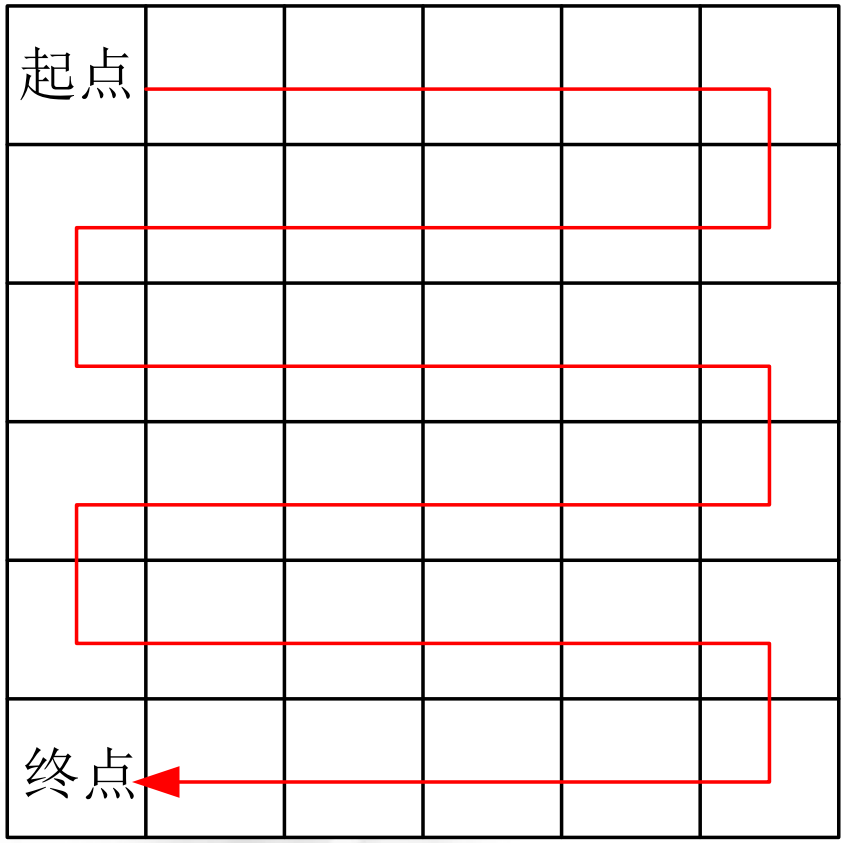
- 一个人带着一只狼一只羊和一捆白菜渡河，一次只能带一件，要求渡河次数尽量少，怎么带？

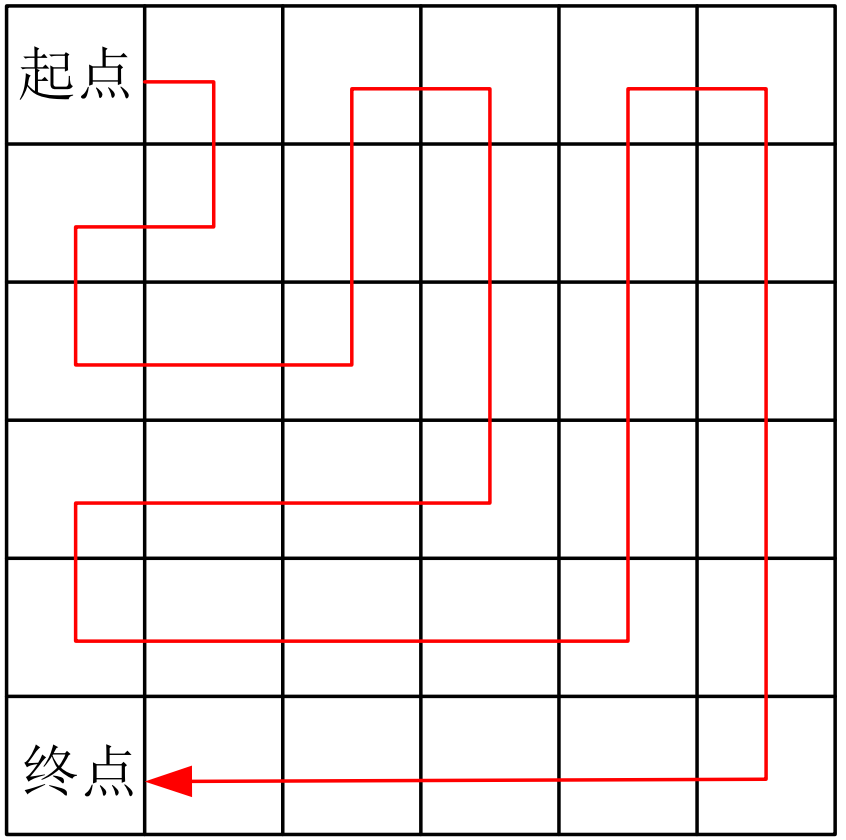
2.4 更多的搜索

例7

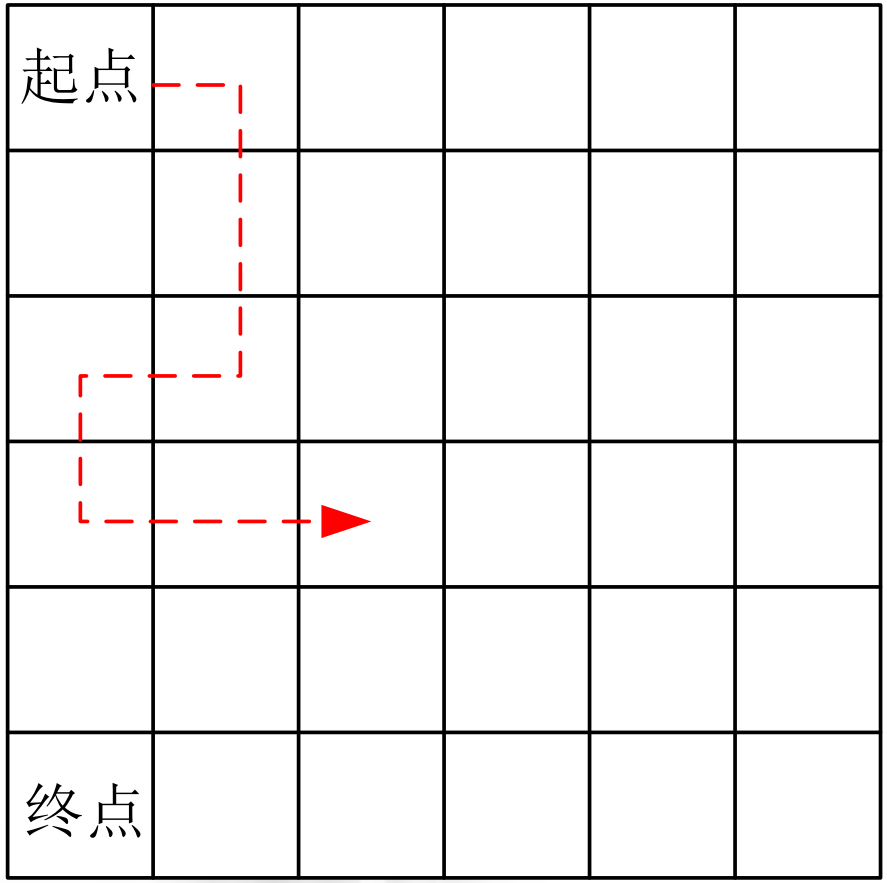
- 一个正方形的小镇被分成N^2个小方格，Betsy要从左上角的方格到达右下角的方格，并且经过每个方格恰好一次。编程对于给定的N，计算出Betsy能采用的所有旅行路线的数据。
- 输入：N（1<=N<=10）
- 输出：旅游路线数

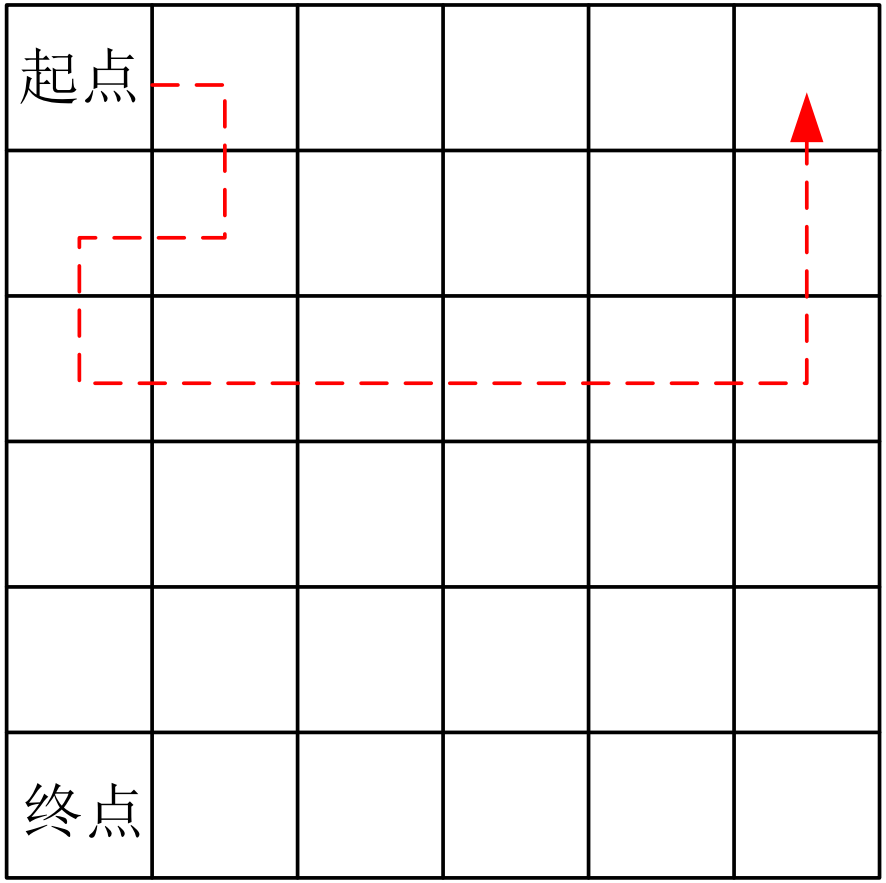
2.4 更多的搜索





2.4 更多的搜索





2.3 深度优先搜索

- 除了出发点和目标格子外的每一个中间格子，都必然有“一进一出”的过程，因此必须保证每个尚未经过的格子都与至少两个尚未经过的格子相邻。
- 如果进入某区域后就无法移至区域外的任何一个格子，这个区域成为封闭区域。在一个合法移动方案的任何时刻，都不可能有封闭区域的存在。

例8

- 如果一个数m，它的约数的个数大于任何一个比m小的自然数的约数个数，那么称这种数为Antiprime数。例如1、2、4、6、12和24。
- 输入：输入整数n（1<=n<=2000000000）；
- 输出：不大于n的最大Antiprime数。

解题思路

$$9720 = 2^3 \times 3^5 \times 5 \qquad 4320 = 2^5 \times 3^3 \times 5$$

- 定理：设 $m = 2^{t_1} \times 3^{t_2} \times \cdots \times p^{t_k}$ （其中p是第k大的素数）是Antiprime数，则必有t1>=t2>=……>=tk>=0。