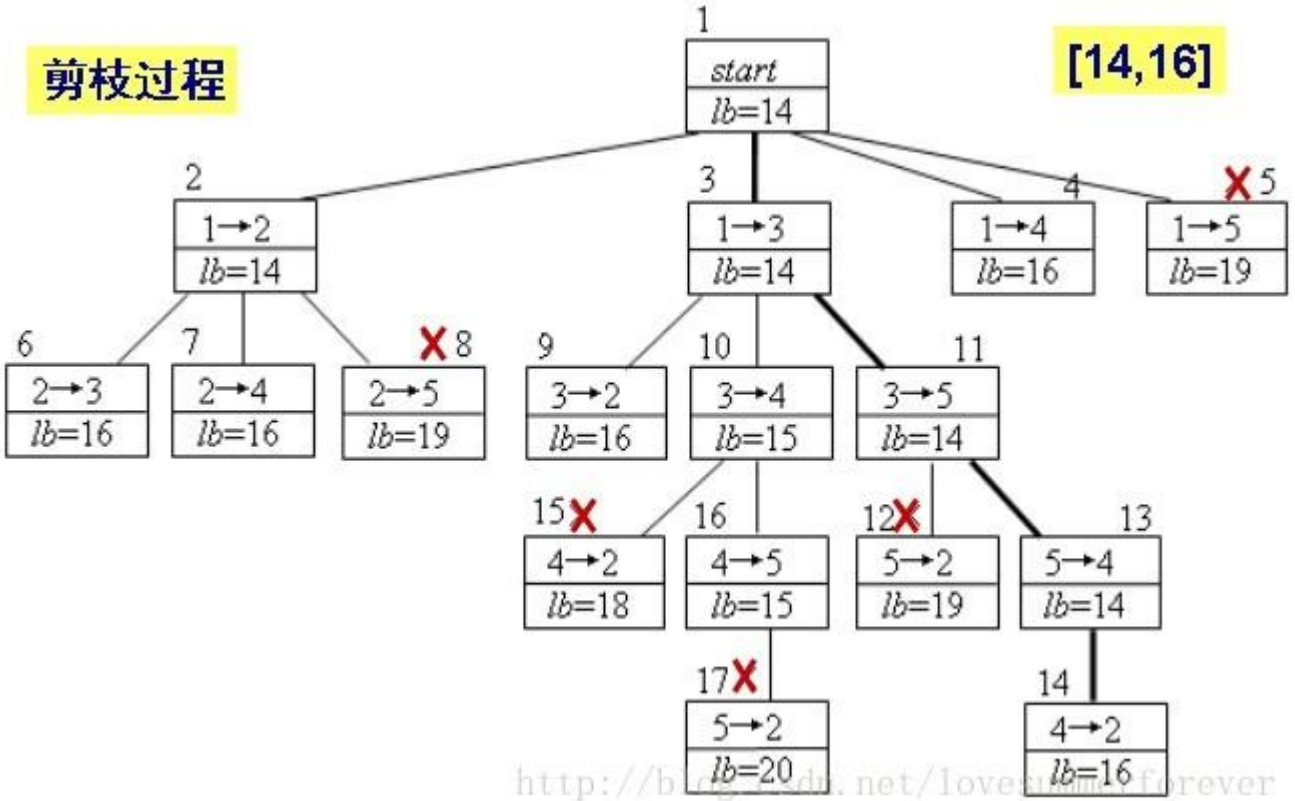
算法设计与分析

分支限界法



主讲： 朱东杰 博士、硕导

地点： M楼305

电话/微信： 18953856806

Email： zhudongjie@hit.edu.cn

分支限界法

- 学习要点
- 理解分支限界法的剪枝搜索策略。
- 掌握分支限界法的算法框架
- （1）队列式(FIFO)分支限界法
- （2）优先队列式分支限界法
- 通过应用范例学习分支限界法的设计策略。
- （1）0-1背包问题；
- （2）旅行售货员（TSP）问题
- （3）单源最短路径问题
- （4）布线问题
- （5）最大团问题；
- （6）装载问题
- （7）电路板排列问题
- （8）批处理作业调度问题

分支限界法概念

分支限界法（branch and bound method）按广度优先策略搜索问题的解空间树，在搜索过程中，对待处理的节点根据限界函数估算目标函数的可能取值，从中选取使目标函数取得极值（极大或极小）的结点优先进行广度优先搜索，从而不断调整搜索方向，尽快找到问题的解。

分支限界法适合求解最优化问题。

分支限界法与回溯法

（1）求解目标：回溯法的求解目标是找出解空间树中满足约束条件的所有解，而分支限界法的求解目标则是找出满足约束条件的一个解，或是在满足约束条件的解中找出在某种意义下的最优解。

（2）搜索方式的不同：回溯法以深度优先的方式搜索解空间树，而分支限界法则以广度优先或以最小耗费优先的方式搜索解空间树

- 回溯法深度优先搜索堆栈活结点的所有可行子结点被遍历后才被从栈中弹出找出满足约束条件的所有解
- 分支限界法广度优先或最小消耗优先搜索队列、优先队列每个结点只有一次成为活结点的机会找出满足约束条件的一个解或特定意义下的最优解

分支限界法基本思想

分支限界法常以广度优先或以最小耗费（最大效益）优先的方式搜索问题的解空间树。问题的解空间树是表示问题解空间的一棵有序树，常见的有**子集树和排列树**

在分支限界法中，每一个活结点只有一次机会成为扩展结点。活结点一旦成为扩展结点，就一次性产生其所有儿子结点。在这些儿子结点中，导致不可行解或导致非最优解的儿子结点被舍弃，其余儿子结点被加入活结点表中。

此后，从活结点表中取下一结点成为当前扩展结点，并重复上述结点扩展过程。这个过程一直持续到找到所需的解或活结点表为空时为止。

分支限界法基本思想

常见的两种分支限界法

（1）队列式(FIFO)分支限界法

按照队列先进先出（FIFO）原则选取下一个节点为扩展节点。

（2）优先队列式分支限界法

按照优先队列中规定的优先级选取优先级最高的节点成为当前扩展节点。

图的基本算法——广度优先搜索

0-1背包问题：

$$Q = \max_{(x_1, \ldots, x_n) \in X} \{ p_1 x_1 + p_2 x_2 + \ldots + p_n x_n \}$$

其中 $X = \{ (x_1, x_2, \ldots, x_n) \mid w_1 x_1 + w_2 x_2 + \ldots + w_n x_n \leqslant C \}$

例. N = 3时的0-1背包问题，考虑下面的具体例子。

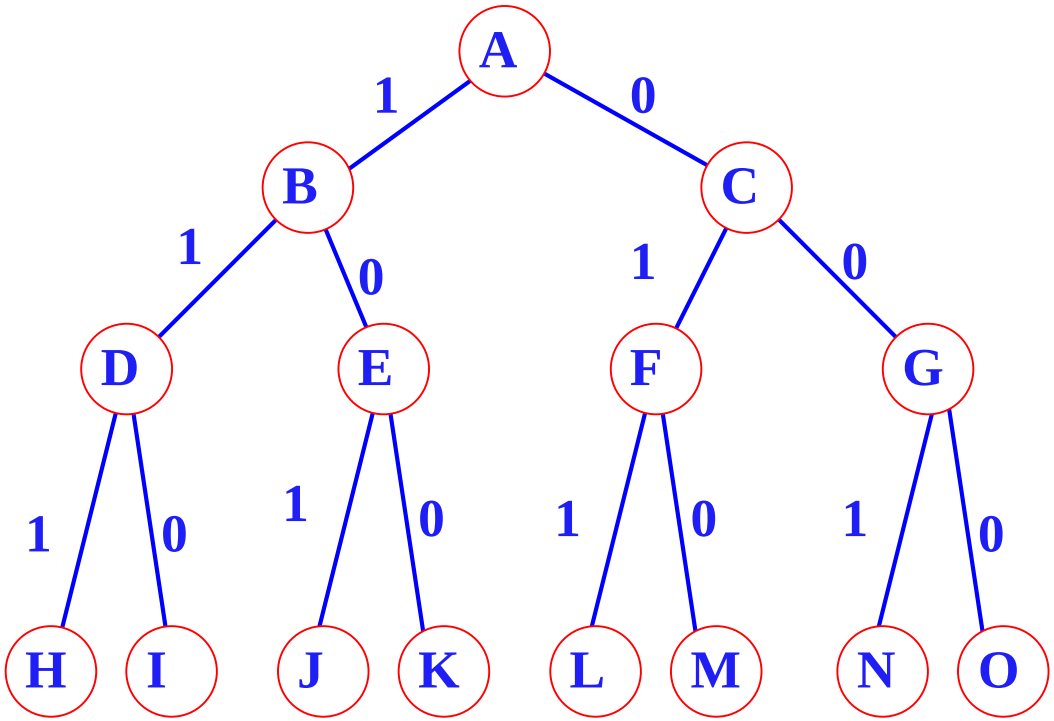
W = [16,15,15]，　P = [45,25,25]，　C = 30。

分析： 问题的解空间为{ (0,0,0)，(0,1,0)，(0,0,1)，(1,0,0)，(0,1,1)，(1,0,1)，(1,1,0)，(1,1,1) }

图的基本算法——广度优先搜索

输入参数： W=[16,15,15],P=[45,25,25], C=30。

问题的解空间为{ (0,0,0), (0,1,0), (0,0,1), (1,0,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1) }

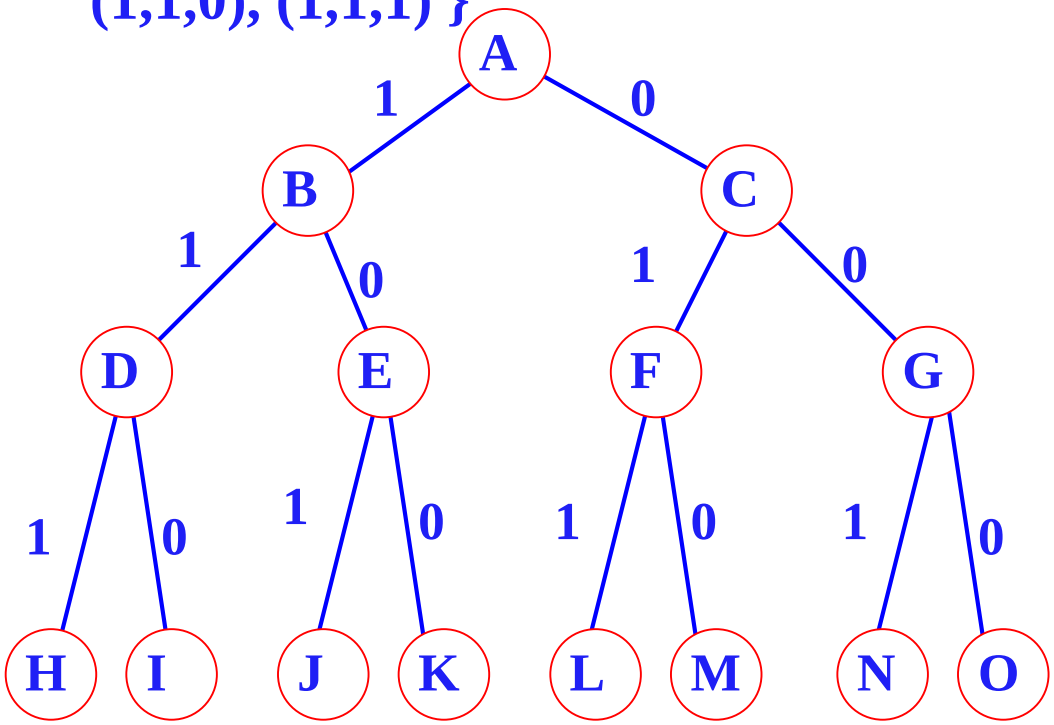


图的基本算法——广度优先搜索

队列式(FIFO)分支限界法

输入参数：W=[16,15,15],P=[45,25,25], C=30。

问题的界空间为{ (0,0,0), (0,1,0), (0,0,1), (1,0,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1) }

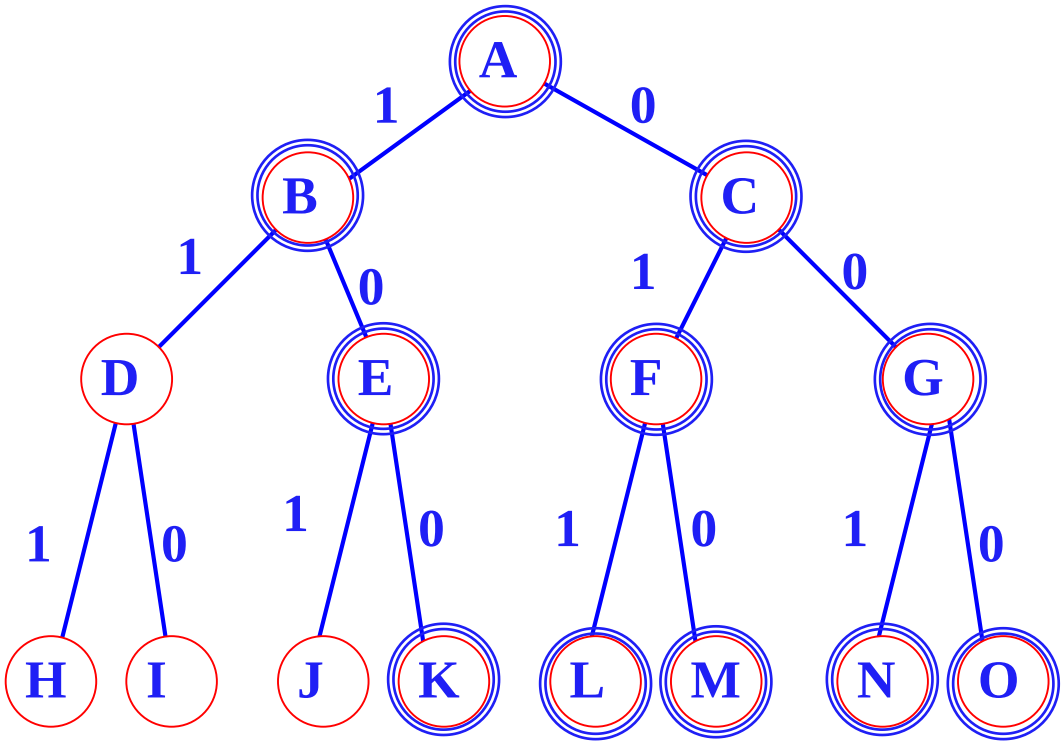


图的基本算法——广度优先搜索

队列式(FIFO)分支限界法

输入参数：W=[16,15,15],P=[45,25,25], C=30。

问题的界空间为{ (0,0,0), (0,1,0), (0,0,1), (1,0,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1) }



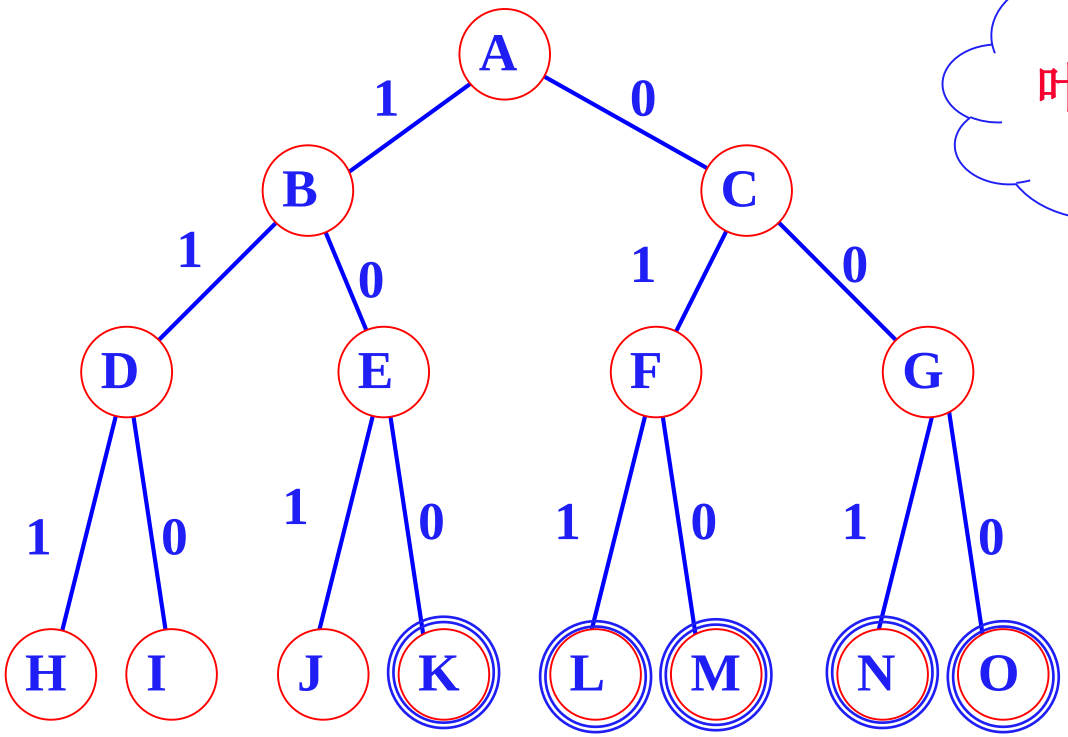
A	0	0
B	16	45
C	0	0
E	16	45
F	15	25
G	0	0
K	16	45
L	30	50
M	15	25
N	15	25
O	0	0

图的基本算法——广度优先搜索

队列式(FIFO)分支限界法

输入参数：W=[16,15,15],P=[45,25,25], C=30。

问题的界空间为{ (0,0,0), (0,1,0), (0,0,1), (1,0,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1) }



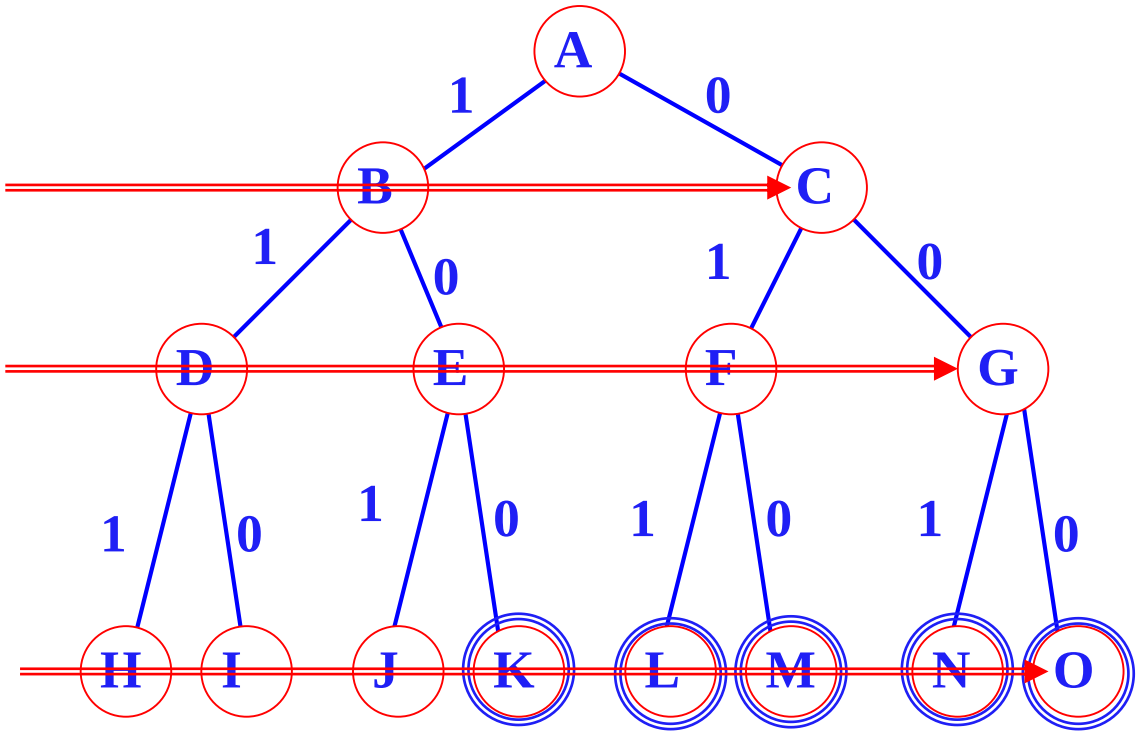
K	16	45
L	30	50
M	15	25
N	15	25
O	0	0

图的基本算法——广度优先搜索

队列式(FIFO)分支限界法

输入参数：W=[16,15,15],P=[45,25,25], C=30。

问题的界空间为{ (0,0,0), (0,1,0), (0,0,1), (1,0,0), (0,1,1), (1,0,1), (1,1,0), (1,1,1) }



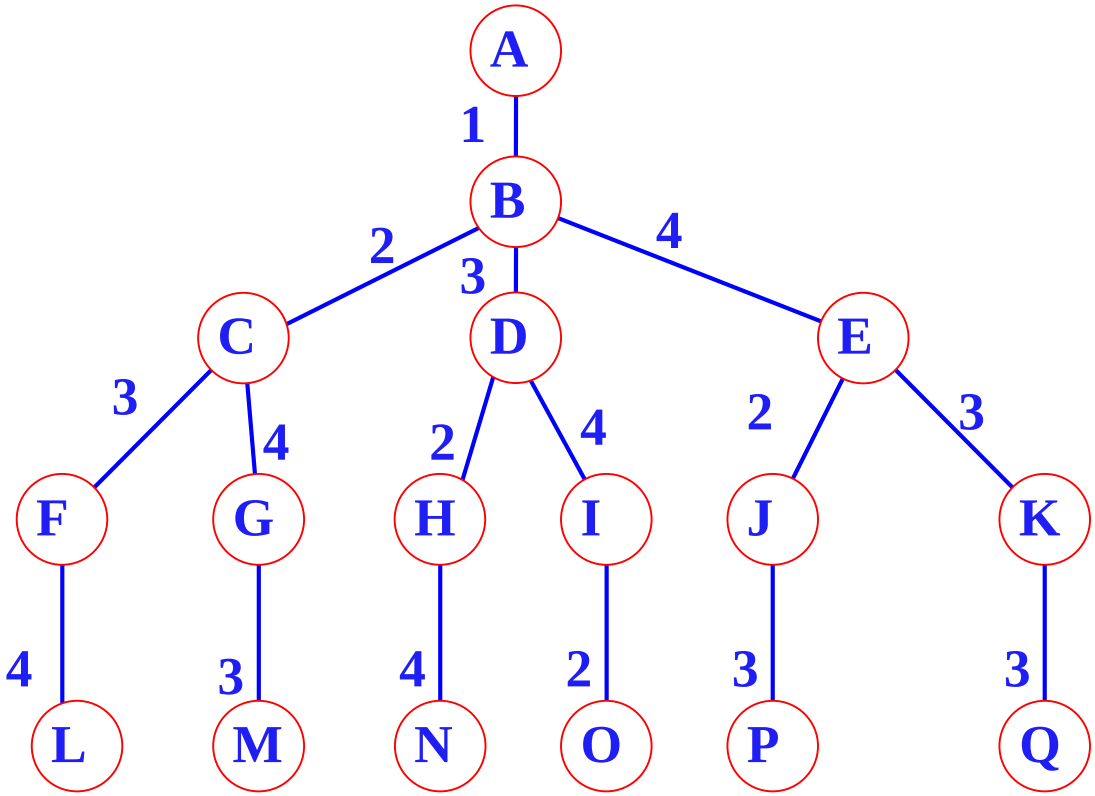
K	16	45
L	30	50
M	15	25
N	15	25
O	0	0

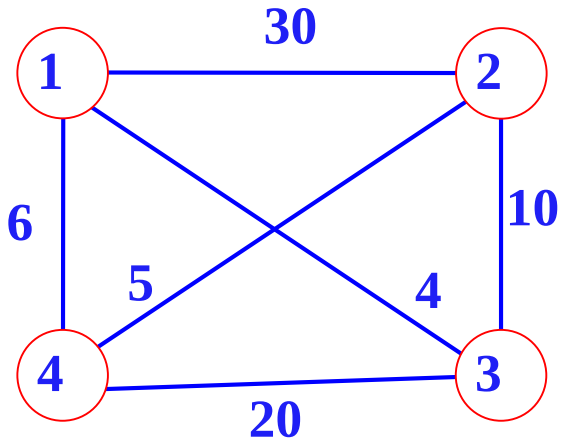
分支限界法的一般过程

- 分支限界法的搜索策略是：在扩展结点处，先生成其所有的儿子结点（分支），然后再从当前的活结点表中选择下一个扩展对点。为了有效地选择下一扩展结点，以加速搜索的进程，在每一活结点处，计算一个函数值（限界），并根据这些已计算出的函数值，从当前活结点表中选择一个最有利的结点作为扩展结点，使搜索朝着解空间树上有最优解的分支推进，以便尽快地找出一个最优解。
- 分支限界法常以广度优先或以最小耗费（最大效益）优先的方式搜索问题的解空间树（**子集树或排列树**）。在搜索问题的解空间树时
  - 每一个活结点只有一次机会成为扩展结点。
  - 活结点一旦成为扩展结点，就一次性产生其所有儿子结点。
  - 在这些儿子结点中，那些导致不可行解或导致非最优解的儿子结点被舍弃，其余儿子结点被子加入活结点表中。
  - 从活结点表中取下一结点成为当前扩展结点，并重复上述结点扩展过程。
  - 一直持续到找到所求的解或活结点表为空时为止。

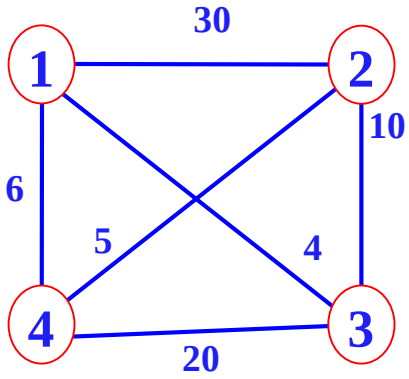
图的基本算法——广度优先搜索

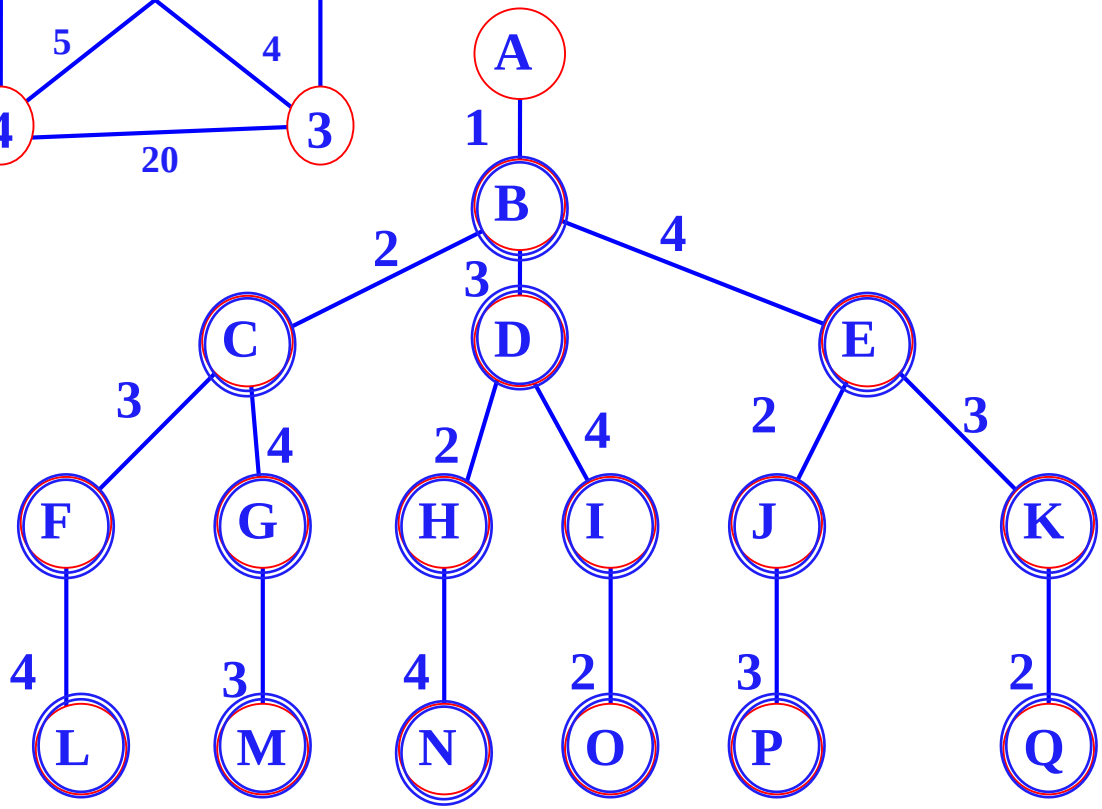
队列式(FIFO)分支限界法





图的基本算法——广度优先搜索

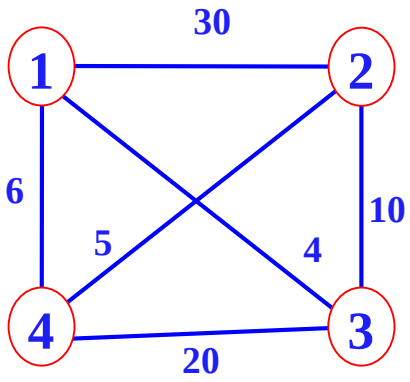


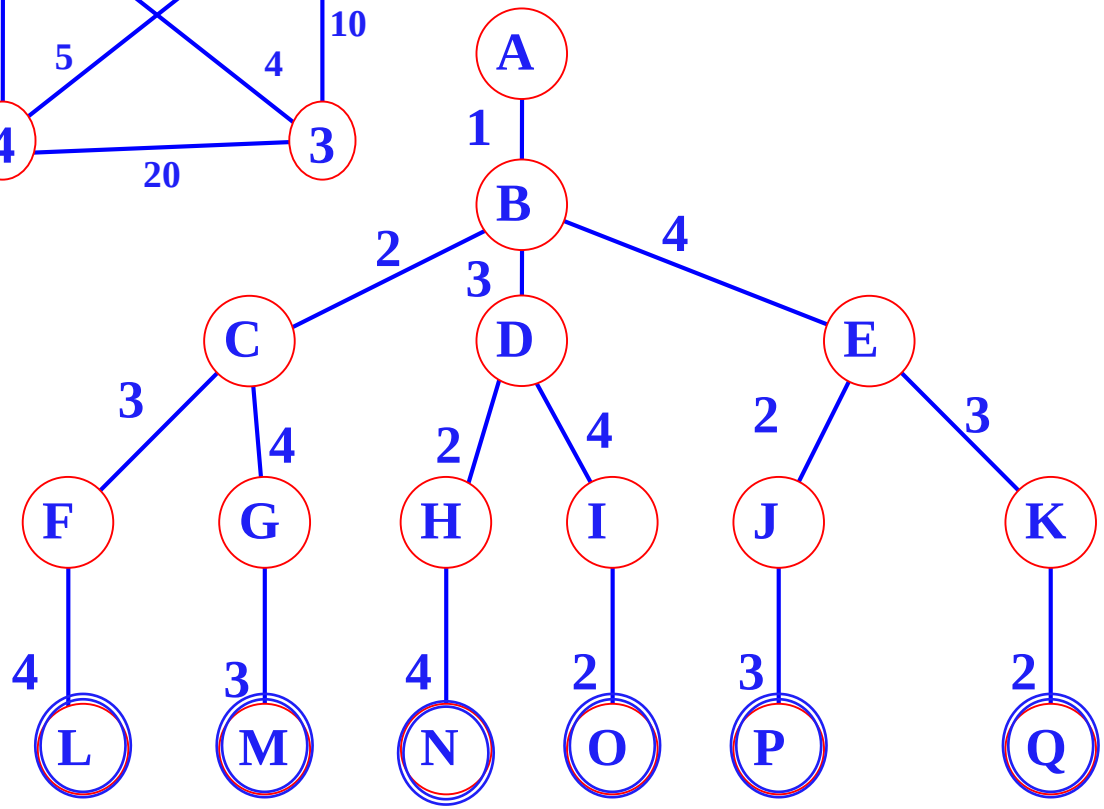


B	0
C	30
D	4
E	6
F	40
G	35
H	14
I	24
J	11
K	26
L	60
M	55
N	19
O	29
P	21
Q	36

图的基本算法——广度优先搜索

队列式(FIFO)分支限界法





L	60	66
M	55	59
N	19	25
O	29	59
P	21	25
Q	36	66

图的基本算法——广度优先搜索

队列式(FIFO)分支限界法

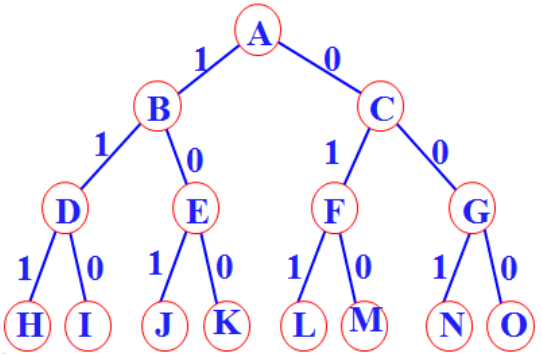
分支限界法通常包含一下三个步骤：

- 针对所给问题，定义问题的解空间。
- 确定易于搜索的解空间结构
- 以广度优先的方式搜索界空间，并在搜索过程中用剪枝函数避免无效搜索。

图的基本算法——练习题

背包问题：

W = [16 , 15 , 15] ， P = [45 , 25 , 25], C = 30 。右图是解空间逻辑树。



图的搜索问题1.

(1) 深度优先搜索时，其中不被搜索的叶节点是

(2) 深度优先搜索时，产生最优解的路径是

(3) 广度优先搜索时，其中不被搜索的叶节点是

图的搜索问题2. 解空间是每个分量取值为0或1的n维的向量空间。

(1) 深度优先搜索时，内存开销是0(　　)。

(2) 广度优先搜索时，内存开销是0(　　)。

(3) 深度优先搜索时，最坏时间复杂性是0(　　)。

(4) 深度优先搜索时，最坏时间复杂性是0(　　)。

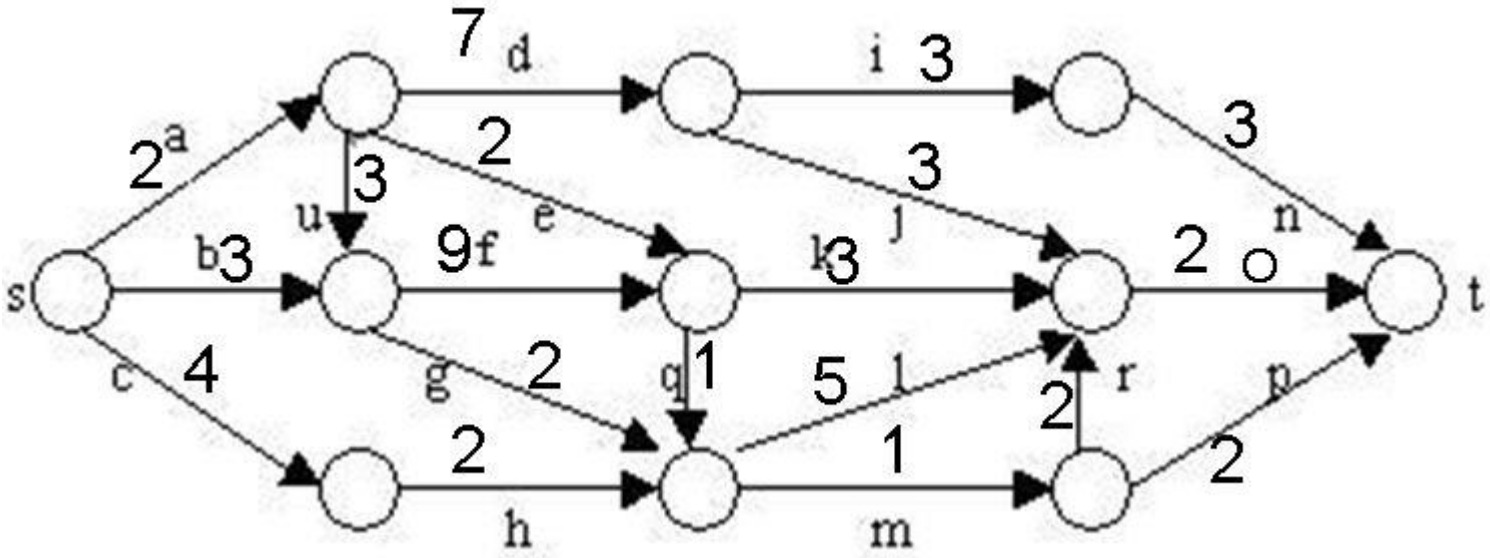
(5) 深度优先搜索时，内存中先生成树之后进行搜索。

(6) 广度优先搜索时，内存中先生成树之后进行搜索。

单源最短路径问题

1. 问题描述

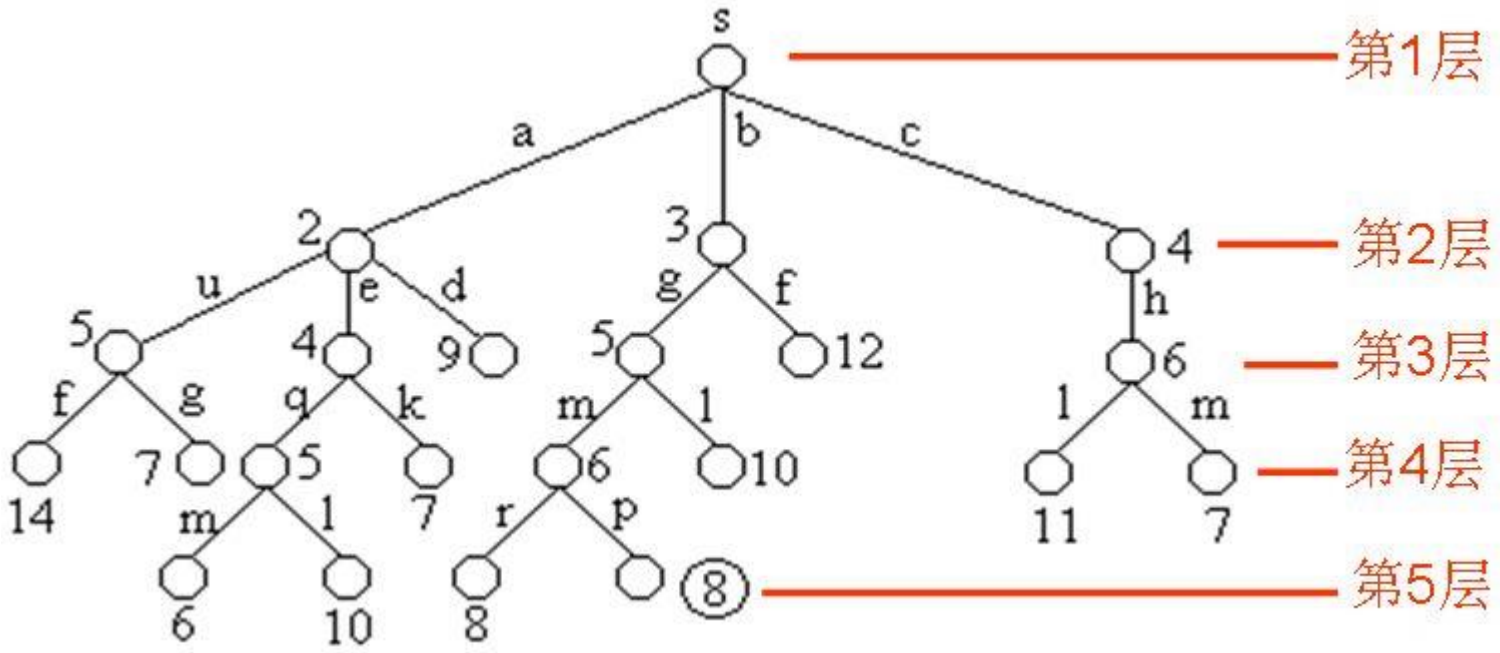
下面以一个例子来说明单源最短路径问题：在下图所给的有向图G中，每一边都有一个非负边权。要求图G的从源顶点s到目标顶点t之间的最短路径。



单源最短路径问题

1. 问题描述

下图是用优先队列式分支限界法解有向图G的单源最短路径问题产生的解空间树。其中，每一个结点旁边的数字表示该结点所对应的当前路长。



单源最短路径问题

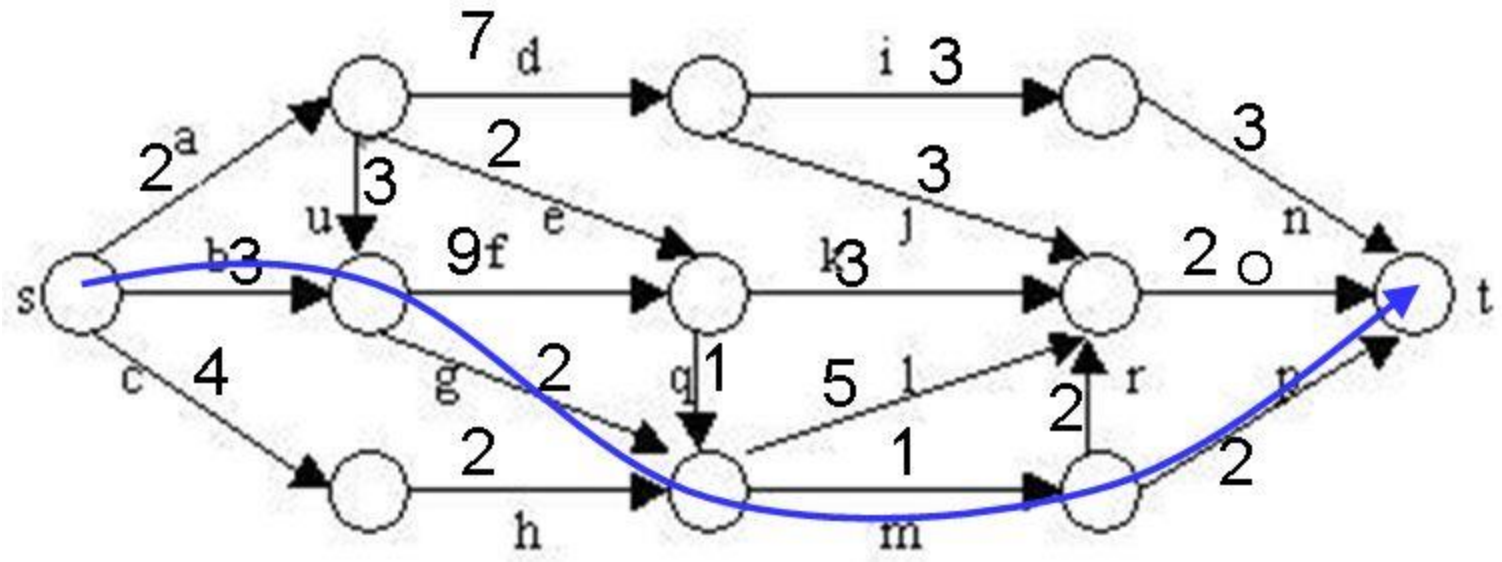
2. 算法思想

解单源最短路径问题的优先队列式分支限界法用一极小堆来存储活结点表。其优先级是结点所对应的当前路长

算法从图G的源顶点s和空优先队列开始。结点s被扩展后，它的儿子结点被依次插入堆中。此后，算法从堆中取出具有最小当前路长的结点作为当前扩展结点，并依次检查与当前扩展结点相邻的所有顶点。如果从当前扩展结点i到顶点j有边可达，且从源出发，途经顶点i再到顶点j的所相应的路径的长度小于当前最优路径长度，则将该顶点作为活结点插入到活结点优先队列中。这个结点的扩展过程一直继续到活结点优先队列为空时为止。

单源最短路径问题

2. 算法思想

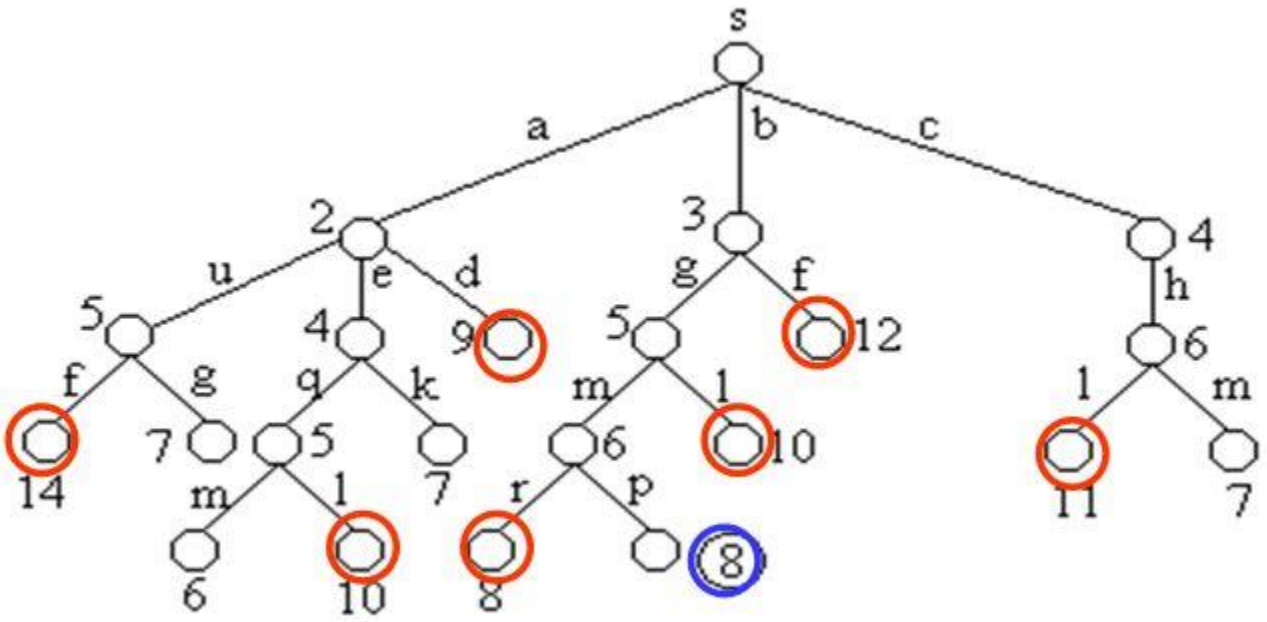


单源最短路径问题

3. 剪枝策略

在算法扩展结点的过程中，一旦发现一个结点的下界不小于当前找到的最短路长，则算法剪去以该结点为根的子树。

在算法中，利用结点间的控制关系进行剪枝。从源顶点s出发，2条不同路径到达图G的同一顶点。由于两条路径的路长不同，因此可以将路长长的路径所对应的树中的结点为根的子树剪去。

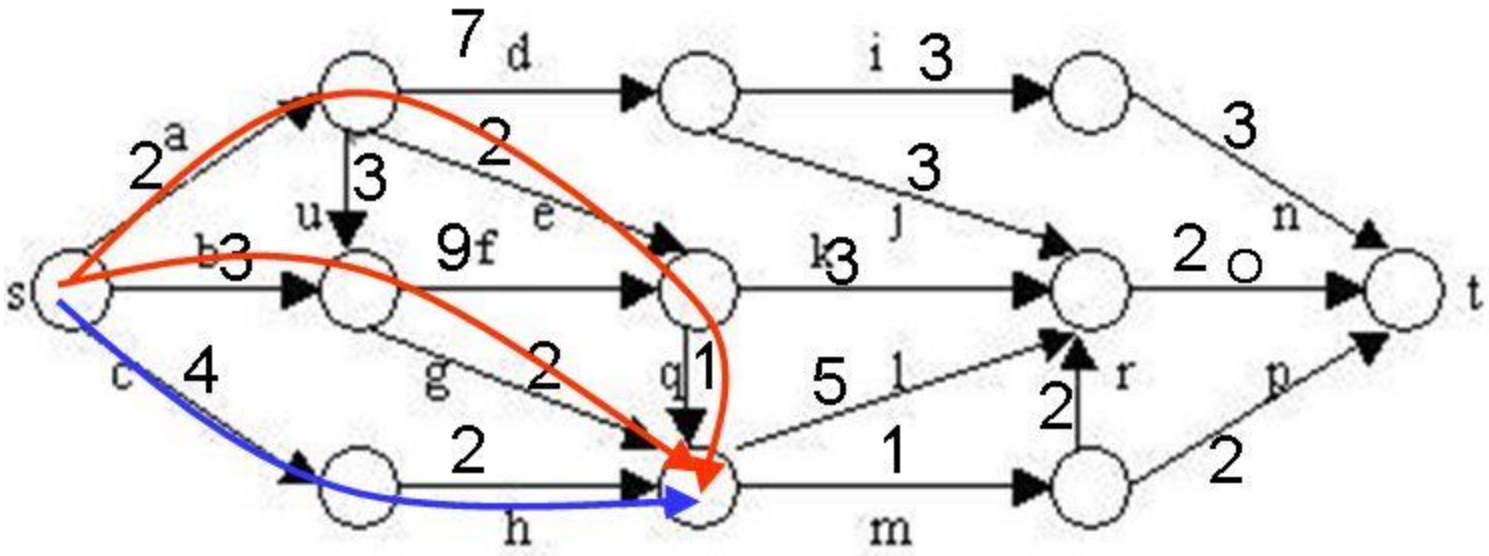


目前的最短路径是8，一旦发现某个结点的下界不小于这个最短路进，则剪枝

单源最短路径问题

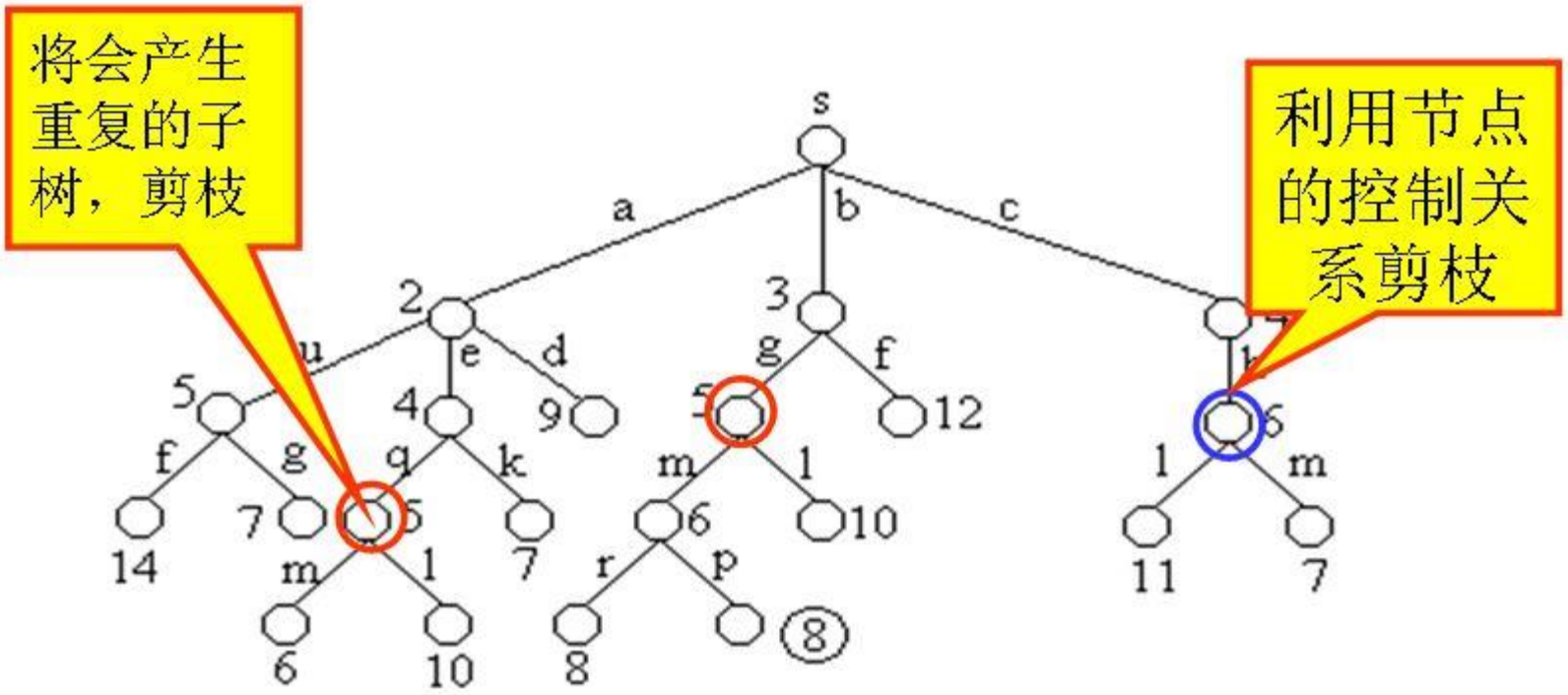
3. 剪枝策略

同一个结点选择最短的到达路径：



单源最短路径问题

3. 剪枝策略

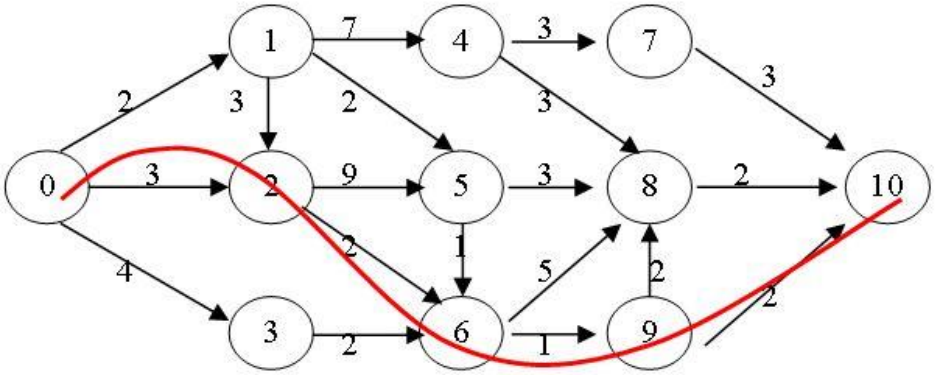


在算法中，利用结点间的控制关系进行剪枝。从源顶点s出发，2条不同路径到达图G的同一顶点。由于两条路径的路长不同，因此可以将路长长的路径所对应的树中的结点为根的子树剪去

单源最短路径伪码

```
while (true) {
    for (int j = 1; j <= n; j++)
      if ((c[E.i][j]<inf)&&(E.length+c[E.i][j]<dist[j])) {
        // 顶点i到顶点j可达，且满足控制约束
        dist[j]=E.length+c[E.i][j];
        prev[j]=E.i;
        // 加入活结点优先队列
        MinHeapNode<Type> N;
        N.i=j;
        N.length=dist[j];
        H.Insert(N);}
    try {H.DeleteMin(E);}        // 取下一扩展结点
    catch (OutOfBounds) {break;}  // 优先队列空
    }
}
```

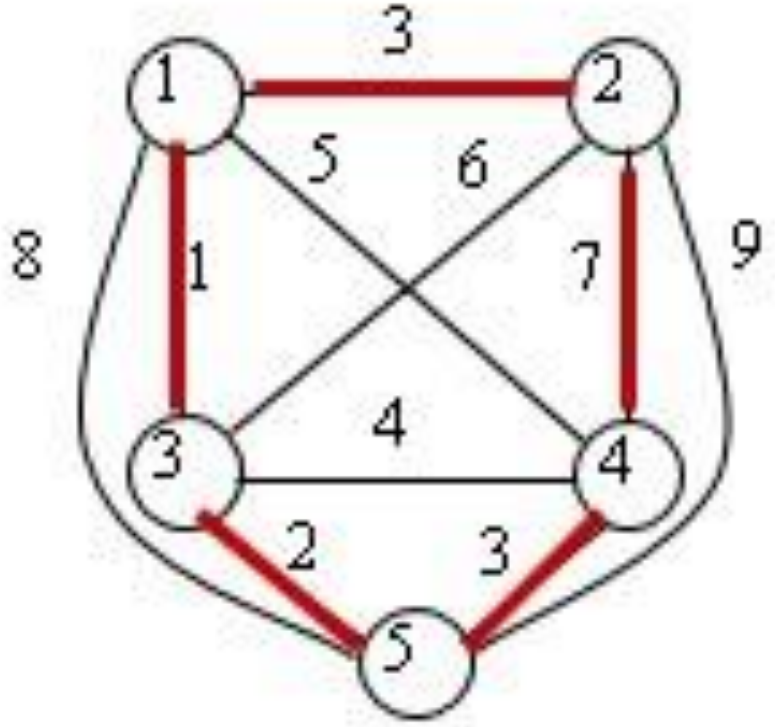
顶点I和j间有边，且此路径长小于原先从原点到j的路径长



测试结果

```
min weight : 8
path: 0 2 6 9 10
```

旅行售货员问题另一种解法



无向图对应的代价矩阵

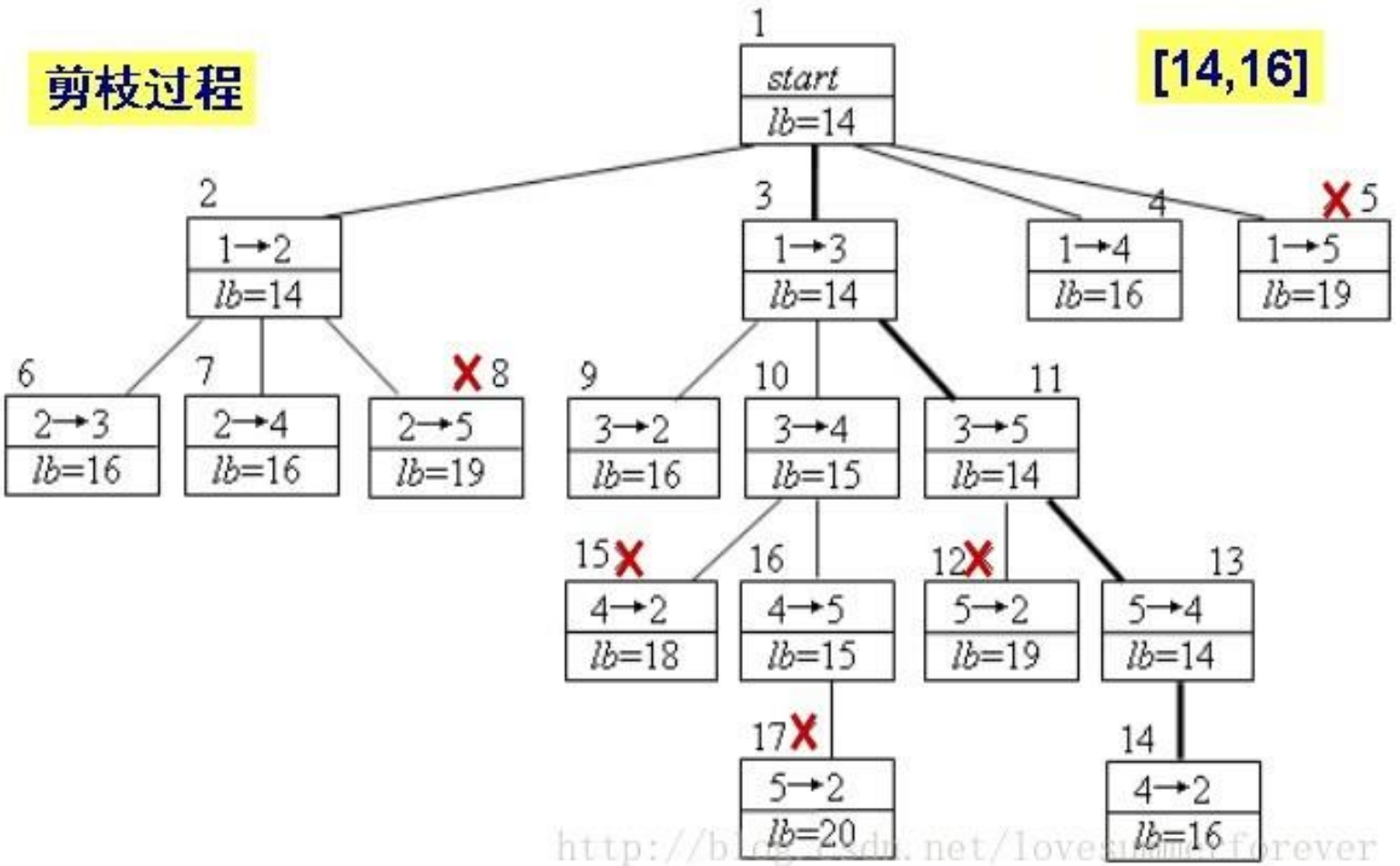
$$C=\begin{pmatrix} \infty & 3 & 1 & 5 & 8 \\ 3 & \infty & 6 & 7 & 9 \\ 1 & 6 & \infty & 4 & 2 \\ 5 & 7 & 4 & \infty & 3 \\ 8 & 9 & 2 & 3 & \infty \end{pmatrix}$$

代价矩阵是1到1,1到2,1到3,1到4,1到5距离写在第一行，第二行为2到1,2到2,2到3,2到4，、、、依次

旅行售货员问题另一种解法

- 找到目标函数的界。上界为，采用贪心算法求得上界，从节点1开始到节点3—>5—>4—>2—>1,路径，即为图中红色圈的路径，其路径长度为C=1+2+3+7+3=16。
  下界为矩阵中每行中两个最小的相加，所有的行加起来的和的一半。((3+1)+(3+6)+(1+2)+(3+4)+(2 +3))/2=14
  所以求得界为[14,16]。
- 计算每个节点的限界值。
  计算目标函数（限界函数），lb分为三部分，第一部分是经过路径的长度相加的2倍，加上第二部分离着路径首尾节点最近的距离相加（不在已知路径上的），加上第三部分除了路径上节点，矩阵中两个最短的距离相加，最后这三部分和相加，得到的结果除以2便是每个节点的限界值。
- 画出PT图。

旅行售货员问题另一种解法



根据上述所述得到最优解1–>3–>5–>4–>2–>1

https://blog.csdn.net/lovesummerforever/article/details/18622127

分支限界法总结

回溯法是从根节点出发，按照深度优先的策略搜索问题的解空间树，在搜索过程中，如果某点所代表的部分解不满足约束条件，则对该节点为根的子树进行剪枝；否则继续按照深度优先的策略搜索以该结点为根，当搜索到一个满足的约束条件的叶子结点时，就找到了一个可行解。

分支限界法首先要确定一个合理的限界函数（bound funciton），并根据限界函数确定目标函数的界[down ,up]，按照广度优先策略搜索问题的解空间树，在分直结点上依次扩展该结点的孩子结点

- 分别估算孩子结点的目标函数可能值，如果某孩子结点的目标函数可能超出目标函数的界，则将其丢弃；
- 否则将其加入待处理结点表（简称PT表）
- 依次从表PT中选取使目标函数取得极值的结点成为当前扩展结点
- 重复上述过程，直到得到最优解。

分支界限法练习

- 通过应用范例学习分支界限法的设计策略。
- （1）布线问题
- （2）最大团问题;
- （3）装载问题
- （4）电路板排列问题
- （5）批处理作业调度问题