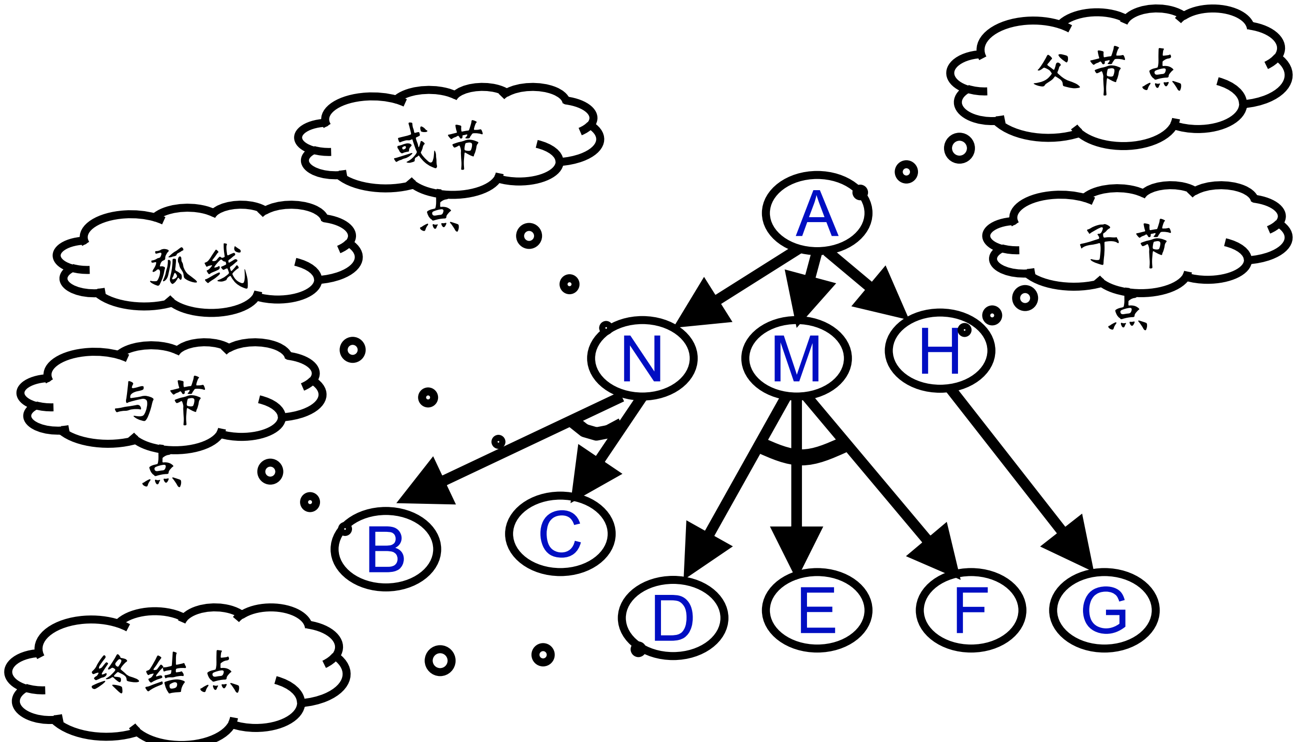
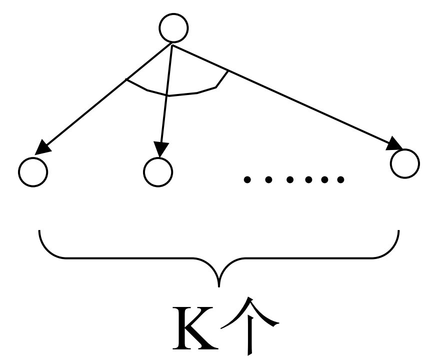
与或图搜索

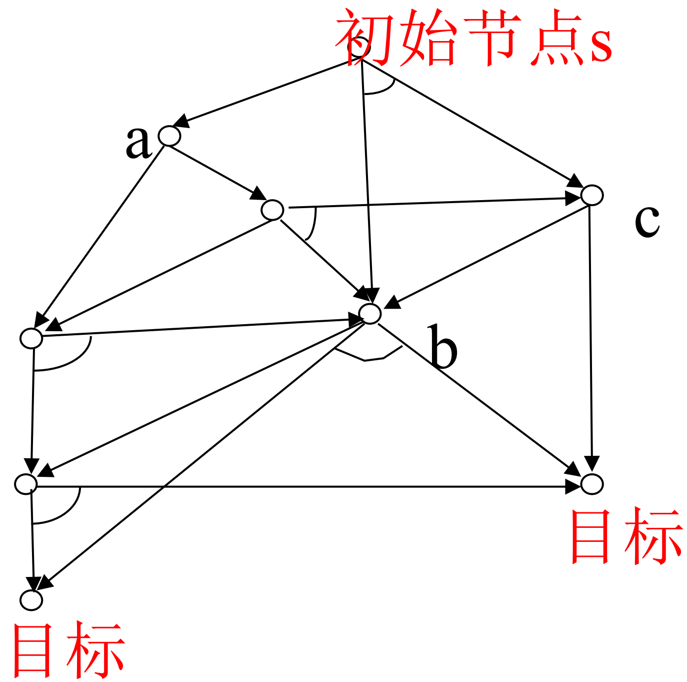
与或图表示



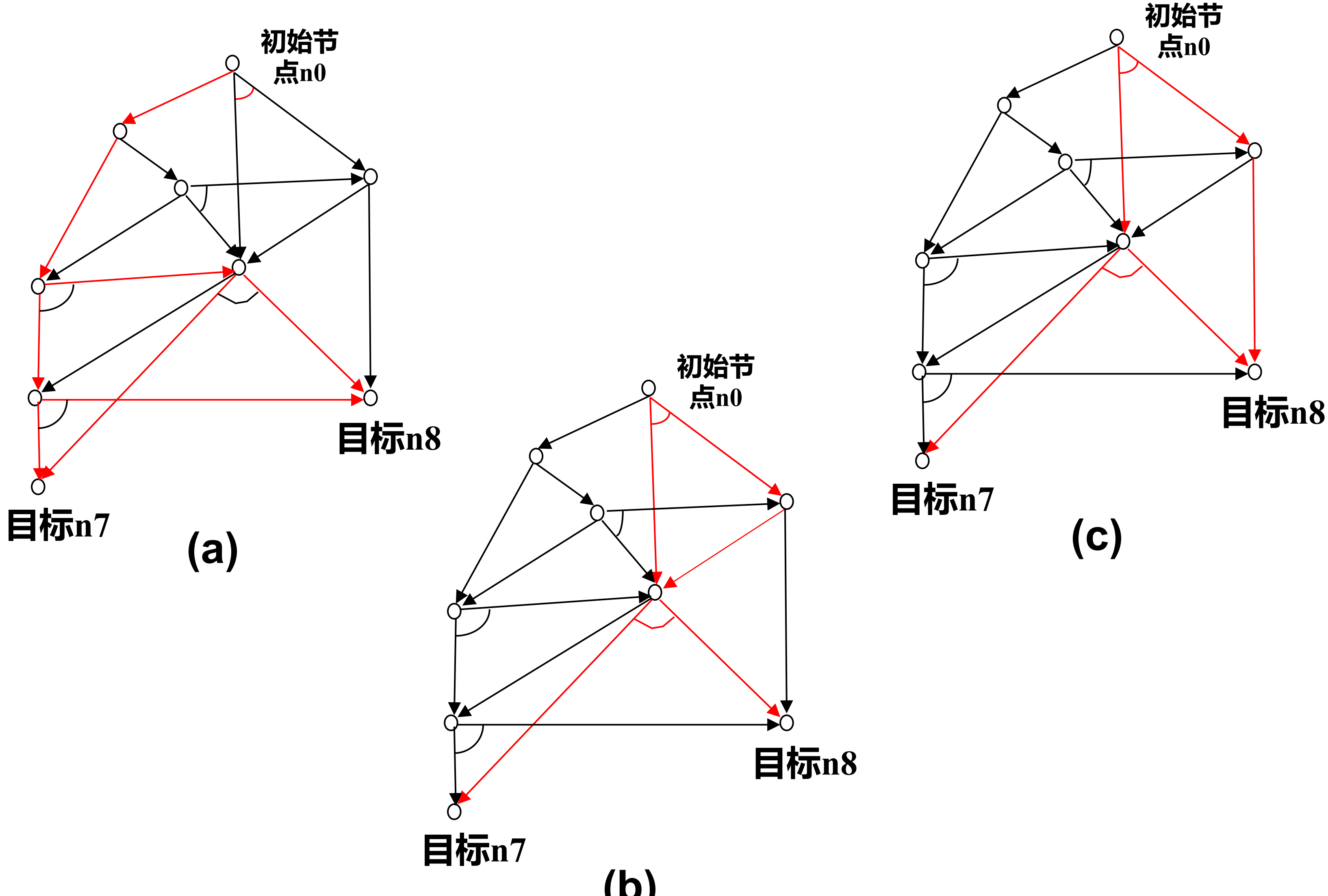
- **与或图是一个超图，节点间通过连接符连接。**
- **K-连接符：**

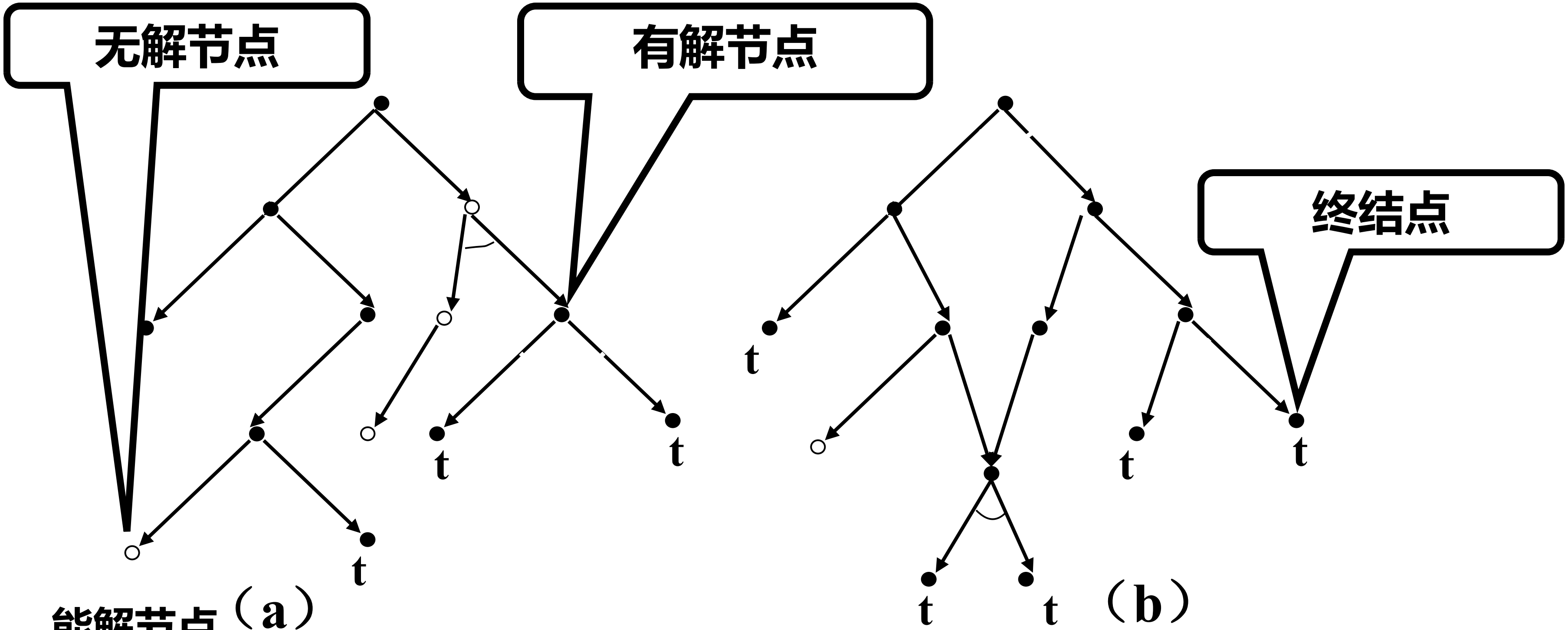


与或图搜索问题



n0→ {n7,n8}的3个解图





能解节点

- **终节点是能解节点**
- **若非终节点有“或”子节点时，当且仅当其子节点至少有一能解时，该非终节点才能解。**
- **若非终节点有“与”子节点时，当且仅当其子节点均能解时，该非终节点才能解。**

不能解节点

- **没有后裔的非终节点是不能解节点。**
- **若非终节点有“或”子节点，当且仅当所有子节点均不能解时，该非终节点才不能解。**
- **若非终节点有“与”子节点时，当至少有一个子节点不能解时，该非终节点才不能解。**

耗散值的计算

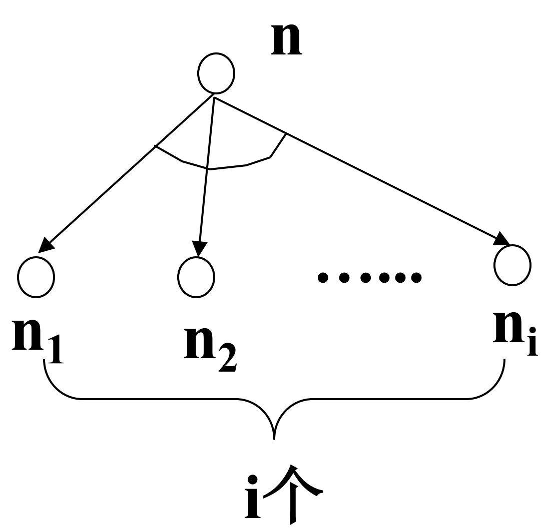
1.若n是N的一个元素, 则k(n, N) =0

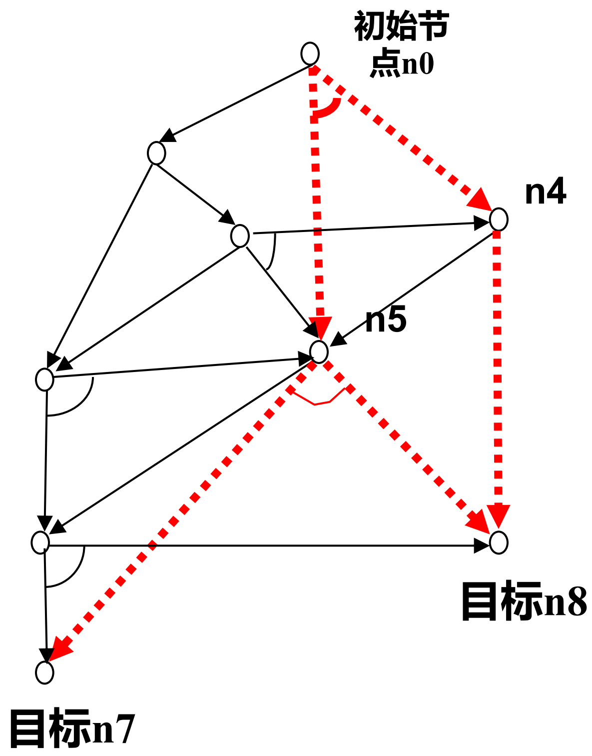
2.若n是一个外向连接符指向后继结点{n1,..ni}

$$k(n, N) = C_n + k(n_1, N) + \ldots + k(n_i, N)$$

其中：N为终节点集

C_n为连接符的耗散值





解图(c)

搜索解图(c)耗散值的递归计算:

n0=2+k(4, N)+k(5, N)

k(5, N)= min(2+k(7,N)+k(8, N),…)

= 2

k(4, N)= min(1+k(5, N), 1+k(8,N))

= min(3, 1)=1

N0= 2+1+2=5

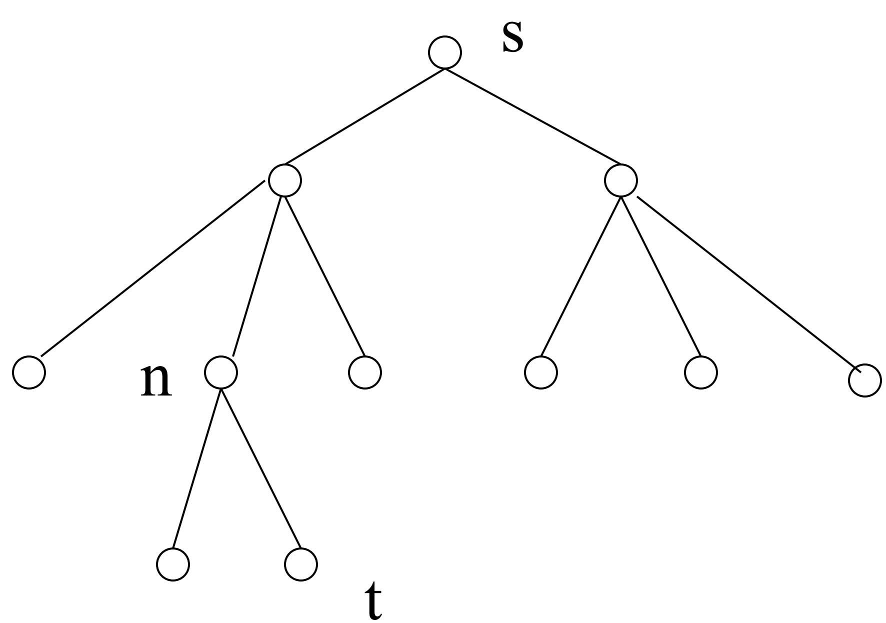
同理可计算得：

(a)的解图耗散值为8

(b)的解图耗散值为7

具有最小耗散值的解图称为最佳解图,其值也用h*(n)标记.上例中的h*(n)=5

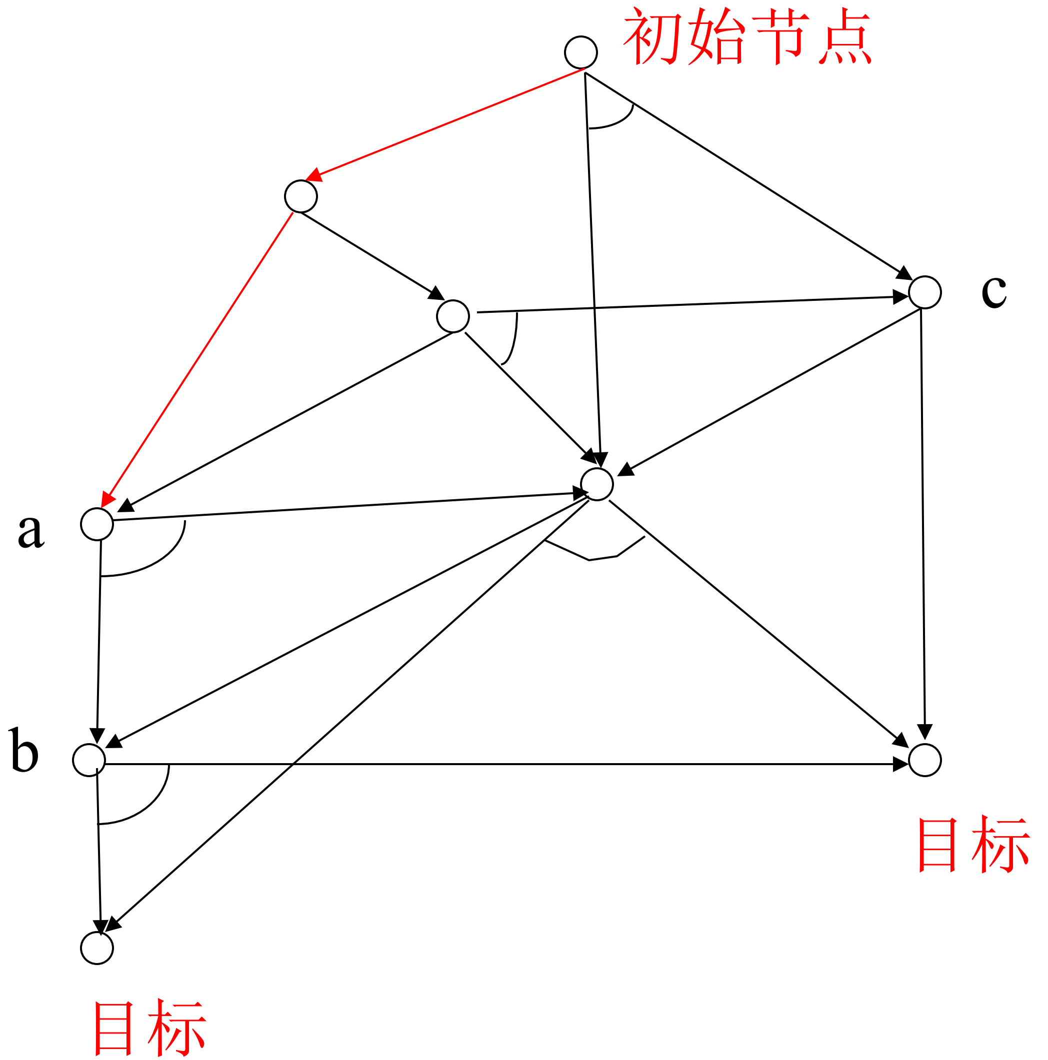
普通图搜索的情况



f(n) = g(n) + h(n)

对n的评价实际是对从s经过n到目的地这条路径的评价

与或图: 对局部图的评价



与或图搜索:AO*算法

两个过程

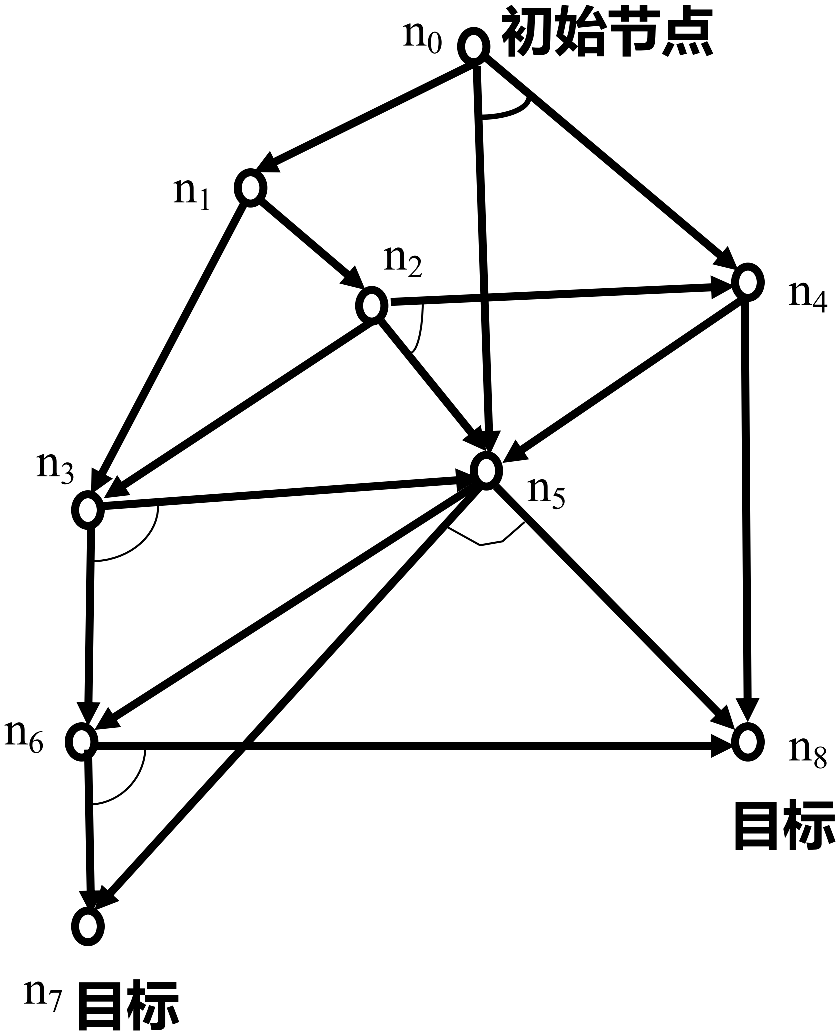
- **图生成过程，即扩展节点**

  自顶向下, 从最优的局部途中选择一个节点扩展

- **计算耗散值的过程**

  自下向顶, 对当前的局部图重新计算耗散值

AO*算法搜索例子

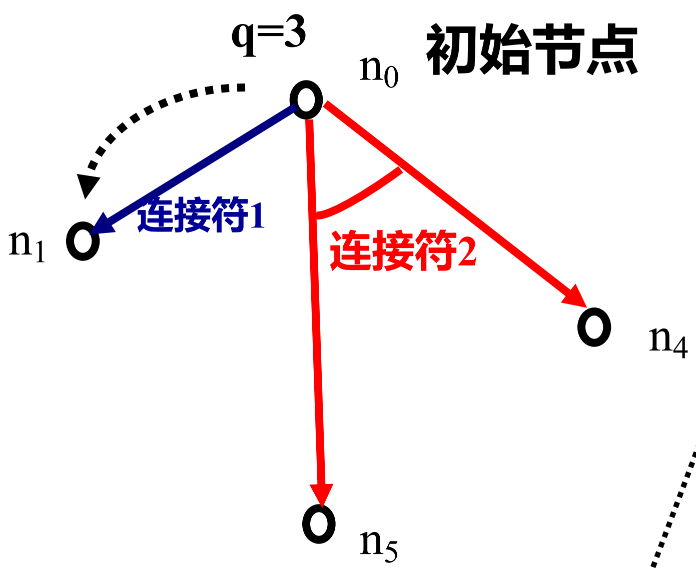


其中：

$h(n_0)=3$
$h(n_1)=2$
$h(n_2)=4$
$h(n_3)=4$
$h(n_4)=1$
$h(n_5)=1$
$h(n_6)=2$
$h(n_7)=0$
$h(n_8)=0$

设：K连接符的耗散值为K

AO*算法搜索例子



即h(n)

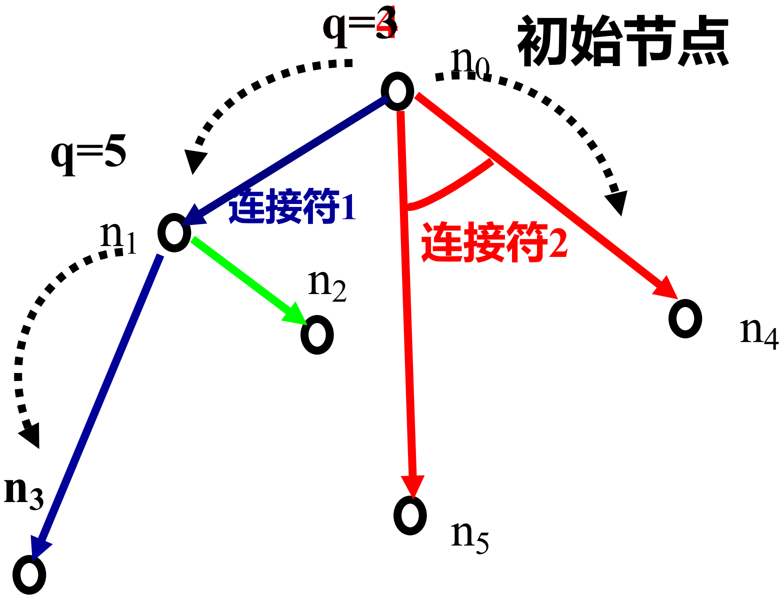
G中只有一个结点n0

第一个大循环(扩展结点),直到n0是SOLVED:

1. 找到待扩展的局部图G'{n0}
2. n=G'中任意结点, 此时n=n0
3. 扩展结点n=n0, 生成后继结点集合{n1,n4,n5}, q(n4)=1, q(n5)=1, q(n1)=2, 都不是终结点, 把结点加到G中
4. 小循环(修改结点耗散值),直到S为空:
   a. S={n=n0}
   b. 保证n的后代不在S中
   c. 取出m=n0的连接符有两条,
      计算q1(m)=1+q(n1)=1+2=3
      q2(m)=2+q(n5)+q(n4)=2+1+1=4
      令q(m)=q(n0)=min(q1,q2)=3
   d.修改指针到q1对应的连结符上去
   e.如果n1为非SOLVED,则m=n0也为非SOLVED
   f.如果m=n0为SOLVED,或者q(m)被修改过,则需要也对m的父结点进行修改, 即将m的父结点加到S中
   g.小循环结束

5.大循环结束

AO*算法搜索例子



G={n0,n1,n4,n5}

第二个大循环(扩展结点),直到n0是SOLVED:

1. 找到待扩展的局部图G'{n0,n1}
2. n=G'中非终结点, 此时n=n1
3. 扩展结点n=n1, 生成后继结点集合{n2,n3}, q(n2)=4, q(n3)=4, 都不是终结点,把结点加到G中
4. 小循环(修改结点耗散值),直到S为空:
   - a. S={n=n1}
   - b. 保证n的后代不在S中
   - c. 取出m=n1的连接符有两条,
     计算q1(m)=1+q(n3)=1+4=5
     q2(m)=1+q(n2)=1+4=5
     令q(m)=q(n0)=min(q1,q2)=5
   - d.修改指针到q1对应的连结符上去
   - e.如果n3非SOLVED,则m=n1也为非SOLVED
   - f. q(m=n1)被修改过,则需要也对m的父结点进行修改, 即将m=n1的父结点n0加到S中
   - g.小循环结束
5. 大循环结束

继续小循环:

…

c. m=n0的连接符有两条,计算

q1(m)=1+h(n1)=1+5=6

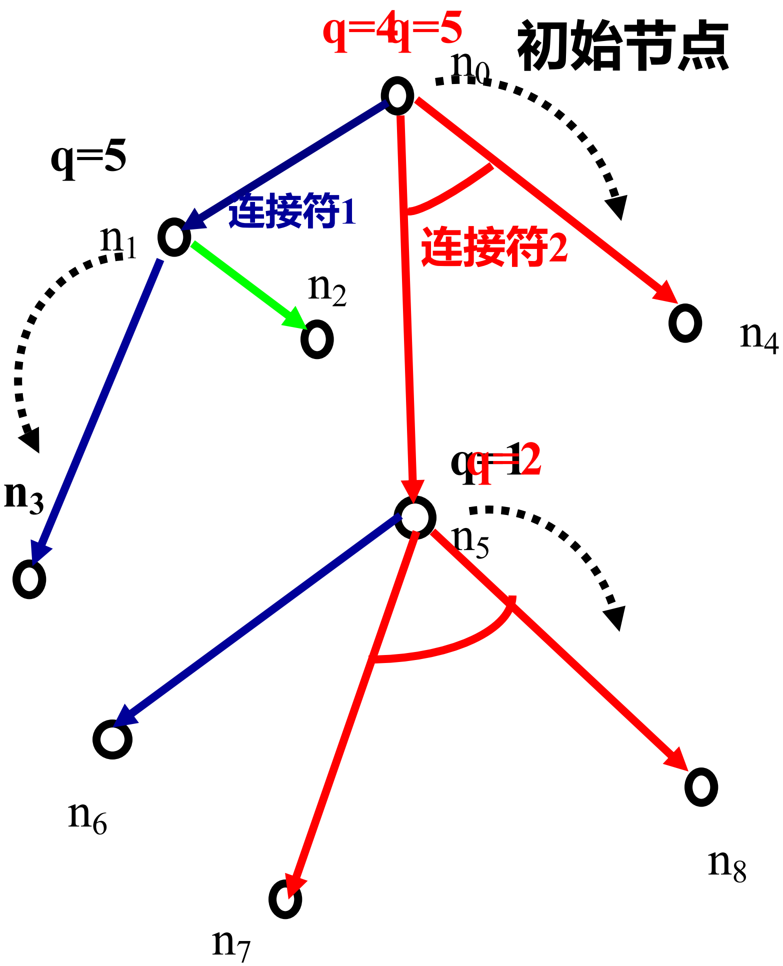
q2(m)=2+h(n5)+h(4)=4

令 q(m)=q(n1)=min(q1,q2)=4

d.修改指针到q2对应的连结符上去

…

AO*算法搜索例子

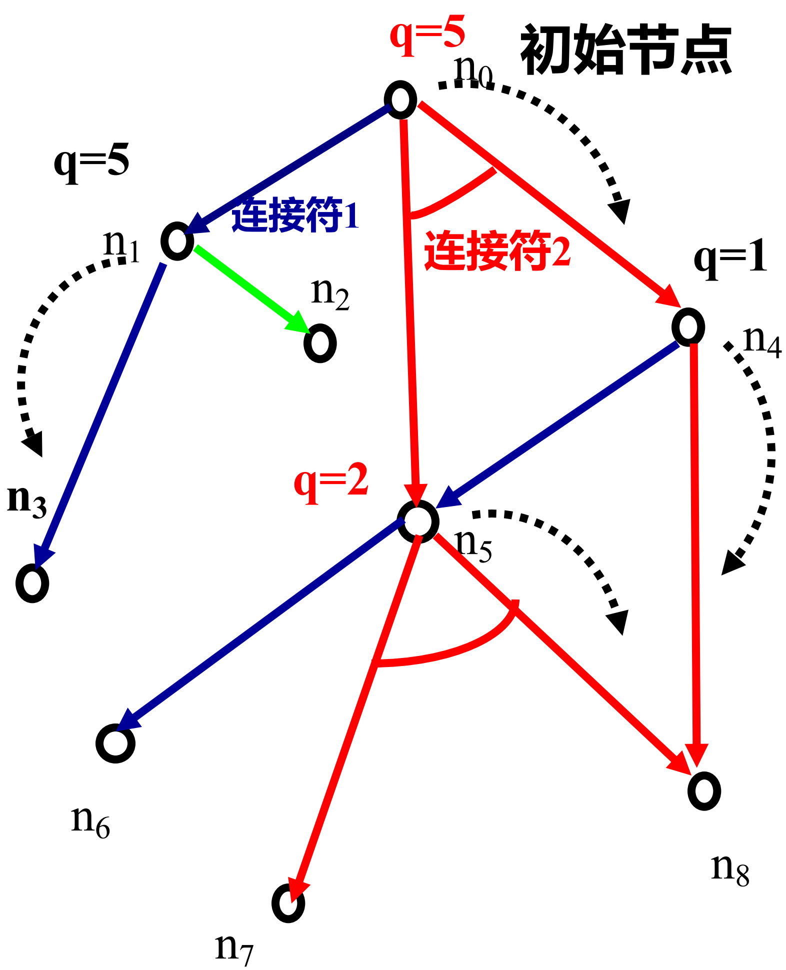


G={n0,n1,n2,n3,n4,n5}

第三个大循环(扩展结点),直到n0是SOLVED:

1. 找到待扩展的局部图G’{n0,n4,n5}
2. n=G’中非终结点, 此时n=n5
3. 扩展结点n=n5, 生成后继结点集合{n6,n7,n8}, q(n6)=2, q(n7)=0, q(n8)=0 把结点加到G中
4. 小循环(修改结点耗散值),直到S为空:
   a. S={n=n5}
   b. 保证n的后代不在S中
   c. m=n5的连接符有两条,
      计算q1(m)=1+q(n6)=1+2=3
      q2(m)=1+q(n7)+q(n8)=2+0+0=2
      令q(m)=q(n5)=min(q1,q2)=2
   d.修改指针到q2对应的连结符上去
   e.n7,n8为SOLVED,则m=n5也为SOLVED
   f. q(m=n5)被修改过,则需要也对m的父结点进行修改, 即将m=n5的父结点n0加到S中
   g.小循环结束
5.大循环结束

AO*算法搜索例子



在重新计算n0耗散的小循环中:

…

由于n4,n5为SOLVED, 则n0为SOLVED q(n0)=5, 找到解

…

G={n0,n1,n2,n3,n4,n5,n6,n7,n8}

第四个大循环(扩展结点),直到n0是SOLVED:

1. 找到待扩展的局部图G'{n0,n4,n5,n7,n8}
2. n=G'中非终结点, 此时n=n4
3. 扩展结点n=n4, 生成后继结点集合{n5, n8}, q(n5)=2, q(n8)=0
4. 小循环(修改结点耗散值),直到S为空:
   a. S={n=n4}
   b. 保证n的后代不在S中
   c. m=n5的连接符有两条,
      计算q1(m)=1+q(n5)=1+2=3
      q2(m)=1+q(n8)=1+0=1
      令q(m)=q(n4)=min(q1,q2)=1
   d.修改指针到q2对应的连结符上去
   e. n8为SOLVED,则m=n4也为SOLVED
   f. m=n4也为SOLVED,则需要也对m的父结点进行修改, 即将m=n1的父结点n0加到S中
   g.小循环结束
5.大循环结束

AO*算法的最优性:

若$s\rightarrow N$存在解图,当$h(n)\le h^*(n)$,且$h(n)$满足单调限制条件,则AO*一定能够找到最佳解图, 即AO*具有可采纳性

单调限制条件指对于图中从结点到$n\rightarrow\{n1,\ldots,nk\}$的每一个连接符都施加限制$h(n)\le C+h(n1)+..+h(nk)$,如果同时有$h(ti)=0(ti\in N)$,那么单调限制意味着h是h*的下界范围,即对所有的结点n有$h(n)\le h^*(n)$