# 人工智能：搜索技术 II

饶洋辉
计算机学院,
中山大学
raoyangh@mail.sysu.edu.cn
http://cse.sysu.edu.cn/node/2471

课件来源：中山大学刘咏梅教授；多伦多大学Sheila McIlraith教授

# 启发式搜索

- A*搜索
- A*的性质
- 如何构造启发式函数
- 练习

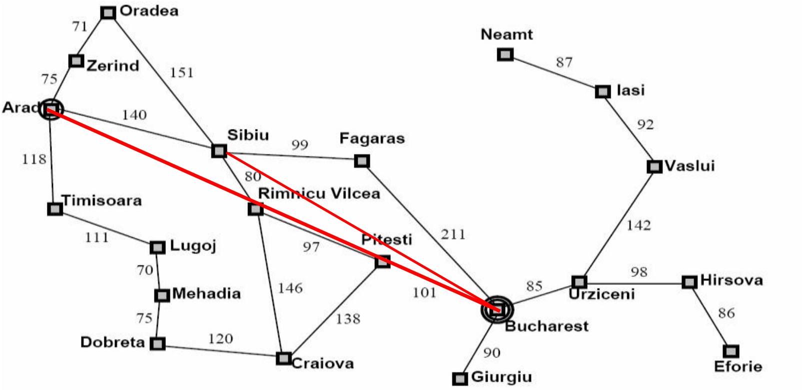
# 动机

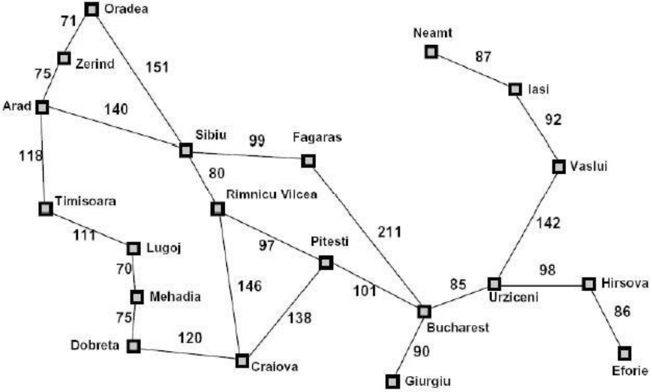
- 在盲目搜索中，我们没有考虑边界上的节点哪一个更具有“前景”（promising)
- 例如在一致代价搜索时，我们总是扩展从初始状态到达当前状态的成本最小的那条路径，却没有考虑过从当前状态点沿着当前路径到达目标路径的成本
- 但是，在许多情况下，我们可以有额外的知识来衡量当前节点，例如可以知道当前节点到达目标节点的成本

# 启发式搜索

- 对于一个具体问题，构造专用于该领域的启发式函数 $h(n)$, 该函数用于估计从节点$n$到达目标节点的成本
- 要求对于所有满足目标条件的节点$n$，$h(n) = 0$
- 在不同的问题领域中，对上述的成本的估计有不同的方法。即，启发式函数是随领域不同而不同的

# 启发式函数示例: 直线距离（欧氏距离）






| Straight−line distance to Bucharest | |
|---|---|
| Arad | 366 |
| Bucharest | 0 |
| Craiova | 160 |
| Dobreta | 242 |
| Eforie | 161 |
| Fagaras | 178 |
| Giurgiu | 77 |
| Hirsova | 151 |
| Iasi | 226 |
| Lugoj | 244 |
| Mehadia | 241 |
| Neamt | 234 |
| Oradea | 380 |
| Pitesti | 98 |
| Rimnicu Vilcea | 193 |
| Sibiu | 253 |
| Timisoara | 329 |
| Urziceni | 80 |
| Vaslui | 199 |
| Zerind | 374 |

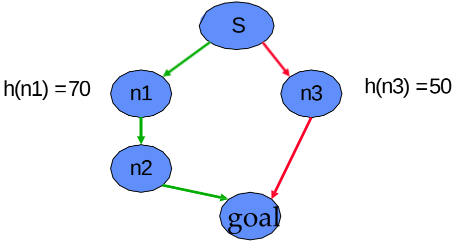
# 贪心最佳优先搜索(Greedy BFS)

- 利用启发式函数$h(n)$ 来对边界上的节点进行排序
- 我们贪心地希望找到成本最低的解
- 但是，这种做法忽略了从初始状态到达节点$n$的成本
- 因此这种做法可能“误入歧途”，选择了离初始状态很远（成本很高），但根据$h(n)$看起来离目标状态很近的节点

→ step cost = 10

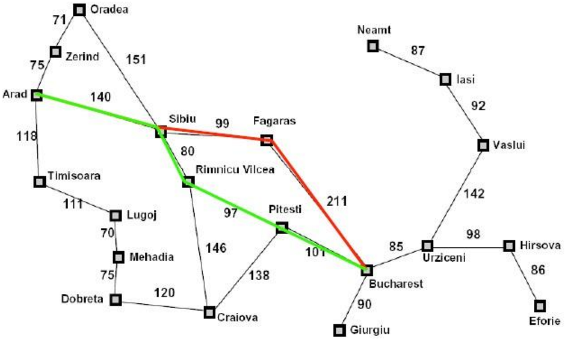
→ step cost = 100

[S]
[n3,n1]
[Goal, n1]



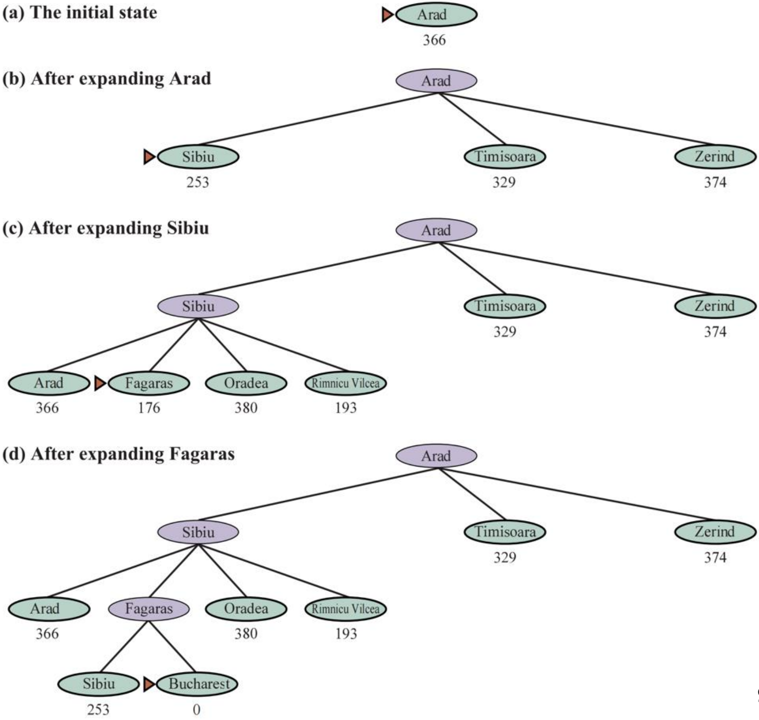

因此Greedy BFS 既不是完备的，也不是最优的。

# 示例




| Straight−line distance to Bucharest | |
|---|---|
| Arad | 366 |
| Bucharest | 0 |
| Craiova | 160 |
| Dobreta | 242 |
| Eforie | 161 |
| Fagaras | 178 |
| Giurgiu | 77 |
| Hirsova | 151 |
| Iasi | 226 |
| Lugoj | 244 |
| Mehadia | 241 |
| Neamt | 234 |
| Oradea | 380 |
| Pitesti | 98 |
| Rimnicu Vilcea | 193 |
| Sibiu | 253 |
| Timisoara | 329 |
| Urziceni | 80 |
| Vaslui | 199 |
| Zerind | 374 |

Arad-Sibiu-RV-Pitesli-Bucharest:
140 + 80 + 97 + 101 = 140 + 278 = 418
Arad-Sibiu-Fagaras-Bucharest: 140 + 99 + 211 = 140 + 310 = 450

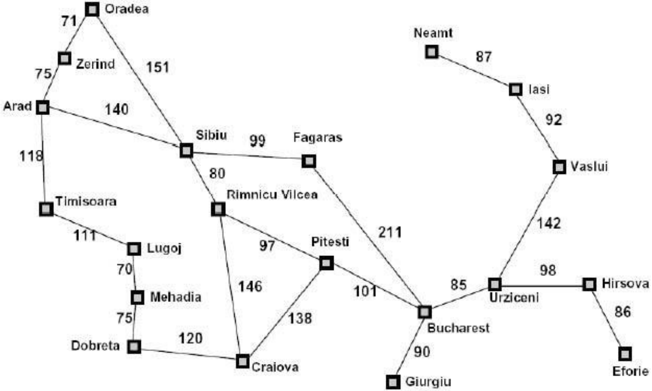


(a) The initial state
Arad
366
(b) After expanding Arad
Arad
Sibiu
253
Timisoara
329
Zerind
374
(c) After expanding Sibiu
Arad
Sibiu
Timisoara
329
Zerind
374
Arad
366
Fagaras
176
Oradea
380
Rimnicu Vilcea
193
(d) After expanding Fagaras
Arad
Sibiu
Timisoara
329
Zerind
374
Arad
366
Fagaras
Oradea
380
Rimnicu Vilcea
193
Sibiu
253
Bucharest
0

# A*搜索

- 定义评价函数$f(n) = g(n) + h(n)$
- $g(n)$是从初始节点到达节点$n$的路径成本
- $h(n)$是从$n$节点到达目标节点的成本的启发式估计值
- 因此，$f(n)$是经过节点$n$从初始节点到达目标节点的路径成本的估计值
- 利用节点对应的$f(n)$值来对边界上的节点进行排序

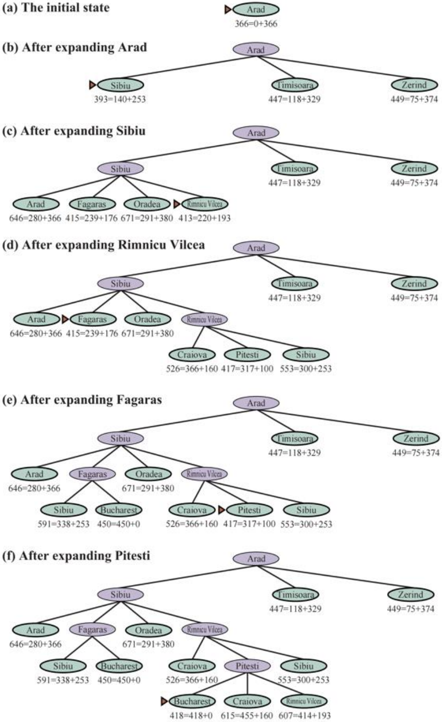
# 示例




Straight−line distance to Bucharest

| | |
|---|---|
| Arad | 366 |
| Bucharest | 0 |
| Craiova | 160 |
| Dobreta | 242 |
| Eforie | 161 |
| Fagaras | 178 |
| Giurgiu | 77 |
| Hirsova | 151 |
| Iasi | 226 |
| Lugoj | 244 |
| Mehadia | 241 |
| Neamt | 234 |
| Oradea | 380 |
| Pitesti | 98 |
| Rimnicu Vilcea | 193 |
| Sibiu | 253 |
| Timisoara | 329 |
| Urziceni | 80 |
| Vaslui | 199 |
| Zerind | 374 |

Arad-Sibiu-RV-Pitesli-Bucharest: 418


(a) The initial state
Arad
366=0+366
(b) After expanding Arad
Arad
Sibiu
393=140+253
Timisoara
447=118+329
Zerind
449=75+374
(c) After expanding Sibiu
Arad
Sibiu
Timisoara
447=118+329
Zerind
449=75+374
Arad
646=280+366
Fagaras
415=239+176
Oradea
671=291+380
Rimnicu Vilcea
413=220+193
(d) After expanding Rimnicu Vilcea
Arad
Sibiu
Timisoara
447=118+329
Zerind
449=75+374
Arad
646=280+366
Fagaras
415=239+176
Oradea
671=291+380
Rimnicu Vilcea
Craiova
526=366+160
Pitesti
417=317+100
Sibiu
553=300+253
(e) After expanding Fagaras
Arad
Sibiu
Timisoara
447=118+329
Zerind
449=75+374
Arad
646=280+366
Fagaras
Oradea
671=291+380
Rimnicu Vilcea
Sibiu
591=338+253
Bucharest
450=450+0
Craiova
526=366+160
Pitesti
417=317+100
Sibiu
553=300+253
(f) After expanding Pitesti
Arad
Sibiu
Timisoara
447=118+329
Zerind
449=75+374
Arad
646=280+366
Fagaras
Oradea
671=291+380
Rimnicu Vilcea
Sibiu
591=338+253
Bucharest
450=450+0
Craiova
526=366+160
Pitesti
Sibiu
553=300+253
Bucharest
418=418+0
Craiova
615=455+160
Rimnicu Vilcea
607=414+193

# $h(n)$的条件：可采纳性

- 假设$c(n_1 \rightarrow n_2) \geq s > 0$。每个状态转移（每条边）的成本是非负的，而且不能无穷地小
- 假设$h^*(n)$ 是从节点$n$到目标节点的最优路径的成本（当节点$n$到目标节点不连通时，$h^*(n) = \infty$）
- 当对于所有节点$n$，满足$h(n) \leq h^*(n)$，$h(n)$是可采纳的
- 所以，可采纳的启发式函数低估了当前节点到达目标节点的成本，使得实际成本最小的最优路径能够被选上
- 因此，对于任何目标节点$g$，$h(g) = 0$

# 一致性 (单调性)

对于任意节点$n_1$和$n_2$，若

$$h(n_1) \leq c(n_1 \rightarrow n_2) + h(n_2)$$

则$h(n)$具有一致性/单调性

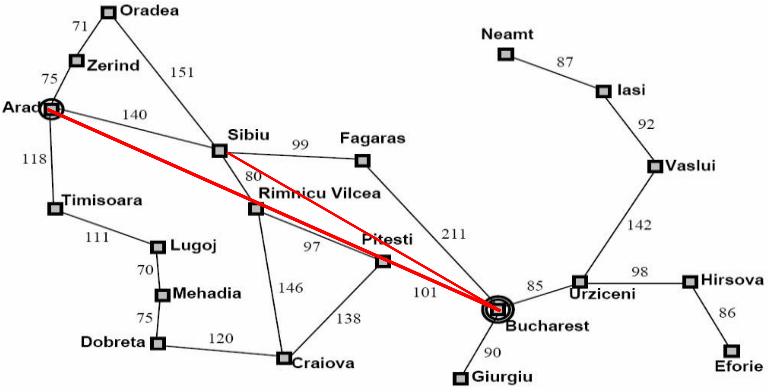
满足一致性的启发式函数也一定满足可采纳性（证明如下）

Case 1: 从节点$n$没有路径到达目标节点，则可采纳性一定成立

Case 2: 假设$n = n_1 \rightarrow n_2 \rightarrow \ldots \rightarrow n_k$ 是从节点$n$到目标节点的一条最优路径。可以使用数学归纳法证明对于所有的$i$，$h(n_i) \leq h^*(n_i)$

大部分的可采纳的启发式函数也满足一致性/单调性

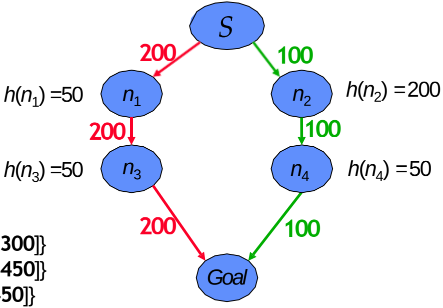
# 示例：直线距离



Oradea
Neamt
Zerind
Iasi
Arad
Sibiu
Fagaras
Vaslui
Rimnicu Vilcea
Timisoara
Pitesti
Lugoj
Mehadia
Urziceni
Hirsova
Bucharest
Dobreta
Craiova
Giurgiu
Eforie
71
75
151
140
118
99
80
87
92
142
111
70
75
120
146
97
138
101
211
85
98
86
90

# 示例1：可采纳但不具备单调性的启发式函数

因为$h(n_2) > c(n_2 \rightarrow n_4) + h(n_4)$，下面的启发式函数不是单调的，但是却是可采纳的




→ step cost = 200
→ step cost = 100

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

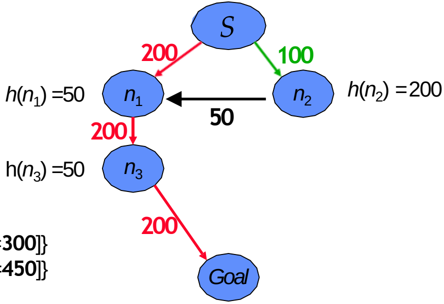
{$S$} → {$n_1$ [200+50=**250**], $n_2$ [200+100=**300**]}
→ {$n_2$ [100+200=**300**], $n_3$ [400+50=**450**]}
→ {$n_4$ [200+50=**250**], $n_3$ [400+50=**450**]}
→ {*Goal* [300+0=**300**], $n_3$ [400+50=**450**]}

$S \rightarrow n_2 \rightarrow n_4 \rightarrow$ *Goal*。虽然确实可以找到最优路径，但是在搜索过程中错误地忽略了$n_2$而去扩展$n_1$

# 示例2：可采纳但不具备单调性的启发式函数

因为$h(n_2) > c(n_2 \rightarrow n_1) + h(n_1)$，下面的启发式函数不是单调的，但是却是可采纳的

→ step cost = 200
→ step cost = 100
→ step cost = 50




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

$\{S\} \rightarrow \{n_1\ [200+50=\mathbf{250}],\ n_2\ [200+100=\mathbf{300}]\}$
$\rightarrow \{n_2\ [100+200=\mathbf{300}],\ n_3\ [400+50=\mathbf{450}]\}$
$\rightarrow \{n_3\ [400+50=\mathbf{450}]\}$
$\rightarrow \{Goal\ [600+0=\mathbf{600}]\}$

采用环检测：$S \rightarrow n_1 \rightarrow n_3 \rightarrow Goal$
$n_2$

最优路径：$S \rightarrow n_2 \rightarrow n_1 \rightarrow n_3 \rightarrow Goal$

# 时间和空间复杂度

$h(n) = 0$时，对于任何$n$这个启发式函数都是单调的。A$*$搜索会变成一致代价搜索

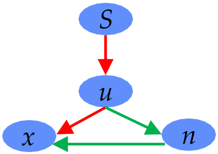
因此一致代价的时间/空间复杂度的下界也适用于A*搜索。即，A*搜索仍可能是指数复杂度，除非我们能才找到好的$h$函数

# 可采纳性意味着最优性

- 假设最优解的成本是$C^*$
- 最优解一定会在所有成本大于$C^*$的路径之前被扩展到
- 成本≤ $C^*$的路径的数量是有限的
- 因此最终可以检测到最优解

# 可采纳性意味着最优性

最优解一定会在所有成本大于$C^*$的路径之前被扩展到




$$p = S \rightarrow u \rightarrow x$$
$$p^* = S \rightarrow u \rightarrow n \rightarrow x$$

证明:

- 假设$p^*$是一个最优解的路径；$p$是一条满足$c(p) > c(p^*)$的路径，而且路径$p$在$p^*$之前被扩展
- 那么扩展到路径$p$时，肯定会有一个$p^*$上的节点$n$处在边界上(前提?)
- 因为$p$在$p^*$之前被扩展，则对于$p$路径上的最后那个节点（假设为目标节点）$x$：$f(x) \le f(n)$
- 因此

  $c(p) = f(x) \le f(n) = g(n) + h(n) \le g(n) + h^*(n) = c(p^*)$
- 和$c(p) > c(p^*)$相矛盾

# 环检测的影响

- 如果启发式函数只有可采纳性，不一定能在使用了环检测之后仍保持最优性
- 为了解决这个问题，必须对于之前遍历过的节点，记录其扩展路径的成本。这样的话，若出现到达已遍历过节点但成本更低的路径，则需重新扩展而不能剪枝
- 但是，启发式函数的单调性可以保证我们在第一次遍历到一个节点时，就是沿着到这个节点的最优路径扩展的
- 因此，只要启发式函数具备单调性，就能在进行环检测之后仍然保持最优性

# 单调的启发式函数

**命题1：**一条路径上的节点的$f$函数值是非递减的

证明:

$$f(n_1) = g(n_1) + h(n_1)$$
$$f(n_2) = g(n_2) + h(n_2) = g(n_1) + c(n_1 \rightarrow n_2) + h(n_2)$$
$$h(n_1) \leq c(n_1 \rightarrow n_2) + h(n_2)$$

# 单调的启发式函数

**命题2：** 如果节点$n_2$在节点$n_1$之后被扩展，则有

$$f(n_1) \le f(n_2)$$

**证明：**

有以下两种情况:

- 当$n_1$被扩展，$n_2$还在边界上。由于$n_2$在$n_1$之后扩展，说明$f(n_1) \le f(n_2)$
- 当$n_1$被扩展，$n_2$的祖先节点$n_3$在边界上，则$f(n_1) \le f(n_3)$。再根据命题1，$f(n_1) \le f(n_3) \le f(n_2)$

# 单调的启发式函数

**命题3：**在遍历节点$n$时，所有$f$值小于$f(n)$的节点都已经被遍历过了

证明：

- 假设存在路径$p = n_1 \rightarrow n_2 \ldots \rightarrow n_k$还没有被遍历过，但$f(n_k) < f(n)$
- 其中，$n_k$ 是路径$p$上最后被遍历的节点
- 路径$p$上的节点$n_{i+1}$应该已经在$n$被探索时的边界上了，因此 $f(n) \le f(n_{i+1})$
- 根据命题1，$f(n_{i+1}) \le f(n_k)$
- 因此$f(n) \le f(n_k)$，与假设矛盾

# 单调的启发式函数

**命题4：** A*搜索第一次扩展到某个状态，就是沿着最小成本的路径进行扩展的

证明：

假设路径$p = n_1 \rightarrow n_2 \ldots \rightarrow n_k = n$是第一条被发现的到达$n$的路径

假设路径$p_j = m_1 \rightarrow m_2 \ldots \rightarrow m_j = n$是第二条被发现的到达$n$的路径

假设$c(p)$是通过$p$路径到达$n$节点的成本

假设$c(p_j)$是通过$p_j$路径到达$n$节点的成本

根据命题2，$c(p) + h(n) \leq c(p_j) + h(n)$

因此 $c(p) \leq c(p_j)$

# IDA*

A*搜索 和宽度优先搜索（BFS）或一致代价搜索（UCS）一样存在潜在的空间复杂度过大的问题

IDA* - 迭代加深的A*搜索与迭代加深搜索一样用于解决空间复杂度的问题

就像迭代加深算法，但用于划定界限的不是深度，而是使用$f$值（$g$+$h$）

在每次迭代时，划定的界限是$f$值超过上次迭代的界限最少的节点的$f$值

# A*搜索：总结

- 定义一个评价函数为$f(n) = g(n) + h(n)$
- 我们使用$f(n)$函数来对边界上的节点进行排序
- 可采纳性：$h(n) \le h^*(n)$
- 单调性：对于任意节点$n1$和$n2$：
  $$h(n_1) \le c(n_1 \to n_2) + h(n_2)$$
- 启发式函数具有单调性说明其也具有可采纳性
- 启发式函数具有可采纳性说明其也具有最优性 (无环检测)
- A*搜索具有指数级的空间复杂度

# 单调的启发式函数：总结

1. 一条路径上的节点的 $f$ 值应该是非递减的
2. 如果$n_2$节点在$n_1$节点之后扩展，那么$f(n_1) \leq f(n_2)$
3. 当节点$n$被扩展时，$f$值小于节点$n$的路径都被扩展过了
4. A*搜索第一次扩展到某个状态，就是沿着最小成本的路径进行扩展的
5. 只要启发式函数是单调的，就能在进行环检测之后仍然保持最优性