西安电子科技大学 XIDIAN UNIVERSITY

人工智能概论

刘若辰
西安电子科技大学 人工智能学院

第二章 状态空间表示以及问题求解

LRC

2.1 问题求解与状态空间表示法

1. 问题的状态空间表示
2. 二阶梵塔难题
3. 猴子摘香蕉问题
4. 其他

引言

- 人工智能的多个研究领域从求解现实问题的过程来看，都可抽象为一个“问题求解”过程，问题求解过程实际上就是一个搜索过程。
- 在搜索过程开始之前，必须先用某种方法或某几种方法的混和来表示问题。
- 问题求解技术主要涉及两个方面：

  问题的表示

  求解的方法
- **状态空间表示法就是用来表示问题及其搜索过程的一种方法。**

- 迷宫问题

- 走迷宫是人们熟悉的一种游戏，右图是一个9个格子迷宫。
- S_0是迷宫入口，S_g是出口
- 迷宫问题就是要找出从入口到出口的路径
- 约束：每次只能走一步

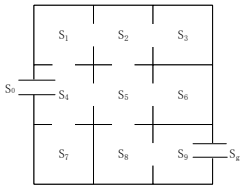
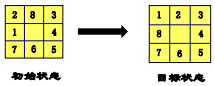


图2.1迷宫

- 八数码难题

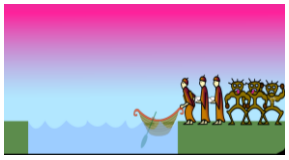
在3×3的棋盘，摆有八个棋子，每个棋子上标有1至8的某一数字。棋盘上还有一个空格，与空格相邻的棋子可以移到空格中。



如何将棋盘从某一初始状态变成最后的目标状态？

- 传教士野人问题

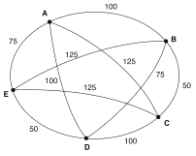
有三个传教士M和三个野人C过河，只有一条能装下两个人的船，在河的一方或者船上，如果野人的人数大于传教士的人数，那么传教士就会有危险.能不能提出一种安全的渡河方法呢?



- 货郎担问题—TSP

货郎担问题：从A城市出发，遍历所有城市返回到A城市最短路径

约束：每个城市只经过一次



问题

- 调度
- 导航
- 分配
- 路径规划
- 游戏
- .
- .
- .

问题的状态空间表示

- 状态空间方法：基于解答空间的问题表示和求解方法，它是以状态和算符为基础来表示和求解问题的。
- 主要包括三个要素：
  - 状态
  - 算符
  - 状态空间
- 典型的例子：下棋、迷宫、各种游戏等。

问题的状态空间表示

- 状态（State）：表示问题求解过程中每一步问题状况的数据结构，一般用向量表示：

$$S_k=(S_{k0}, S_{k1}, \ldots)$$

式中每个元素为向量的分量，称为状态变量

- 每一个分量给予确定的值时，得到一个具体的状态。
- 任何一种类型的数据结构都可以用来描述状态，只要它有利于问题求解。
- 在程序中用字符、数字、记录、数组、结构、对象等表示状态。

问题的状态空间表示

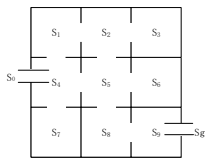
- **迷宫问题的状态：**

- 以每个格子作为一个状态,并用其标识符作为其表示。

该迷宫的状态可表示为：$S_k=(S_i)$

S_i可以取值为$\{S_0, S_1, S_2, S_3, \ldots, S_g\}$

- 初始状态：S_0
- 目标状态：S_g

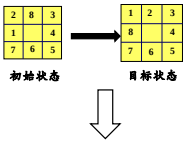


- **八数码问题的状态:**

➢ 将棋局

表2.1

X_1	X_2	X_3
X_8	X_0	X_4
X_7	X_6	X_5



用向量$S=(X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)$表示状态

X_i为变量，X_i的值就是方格X_i内的数字，取值为{0,1,2…8}，其中0表示空格。

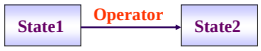
于是，向量S就是该问题的状态表达式。

初始状态:S_0=(028345671)

目标状态:S_g=(012345678)

➢ **算符（Operator）：当对一个问题状态使用某个可用操作时，它将引起该状态中某些分量值的变化，从而使问题从一个具体状态变为另一个具体状态。**

- 算符可理解为状态集合上的一个函数，它描述了状态之间的关系。
- 算符可以是某种操作、规则、 行为、变换、函数、算子、过程等。
- 算符也称为操作，问题的状态也只能经定义在其上的这种操作而改变。



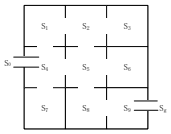
- **迷宫问题的算符**

➢ **以每个格子作为一个状态，并用其标识符作为其表示。**那么，两个标识符组成的序对就是一个算符。于是：

- 初始状态：S_0
- 该迷宫的算符 F *(21)*：

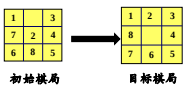
F={(S_0, S_4), (S_4, S_0), (S_4, S_1), (S_1, S_4), (S_1,S_2), (S_2, S_1), (S_2, S_3), (S_3, S_2), (S_2, S_5), (S_5, S_2), (S_4, S_7), (S_7, S_4), (S_4, S_5), (S_5, S_4), (S_5, S_6), (S_6, S_5), (S_5, S_8), (S_8, S_5), (S_8, S_9), (S_9, S_8), (S_9, S_g)}

- 目标状态：S_g



- **八数码难题的算符**

对于八数码难题：



➢ **制定操作算符集：**

- **直观方法——为每个棋牌制定一套可能的走步：左、上、右、下四种移动。这样就需32个操作算子。**
- **简易方法——仅为空格制定这4种走步，因为只有紧靠空格的棋牌才能移动。空格移动的唯一约束是不能移出棋盘。**

问题的状态空间表示

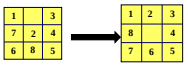
➢ **问题的状态空间（State space）：用来描述一个问题的全部状态以及这些状态之间的相互关系。**

➢ 状态空间常用一个三元组表示：（S，F，G）

- S：为问题的所有初始状态的集合；
- F：为操作的集合；
- G：为目标状态的集合。

问题的状态空间表示

● **状态空间图**



➢ **从初始棋局开始，试探由每一合法走步得到的各种新棋局，然后再走一步而得到的下一组棋局。这样继续下去，直至达到目标棋局为止。**

➢ **把初始状态可达到的各状态所组成的空间设想为一幅由各种状态对应的节点组成的图。这种图称为状态空间图。图中每个节点是它所代表的棋局。首先把适用的算符用于初始状态，以产生新的状态；算符可以看成状态空间图的边。**

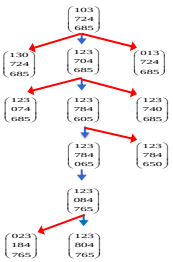
图2.2八数码难题部分状态空间图

问题的状态空间表示

➢ **状态图示法**

状态空间的图示形式称为状态空间图(有向图，)。其中图的节点表示状态，图的边表示算符

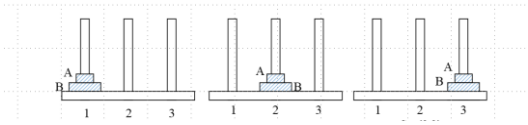
➢ **问题求解过程就是寻求图的某一路径的问题，实际上是一个搜索过程。**



二阶梵塔难题

➢ **问题描述：设有三根柱子，它们的编号分别是1号、2号和3号。**

- **在初始情况下，1号柱子上穿有A、B两个盘子，A比B小，A位于B的上面。**
- **要求把这两个盘子全部移到另一根柱子上，而且规定每次只能移动一个盘子，任何时刻都不能使大的位于小的上面。**
- **如何将A、B移到2号或3号柱子上面?**



二阶梵塔难题—状态

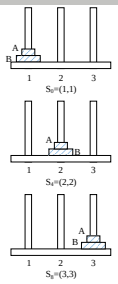
➢ 状态：设$S_K=(S_{K0},S_{K1})$表示问题的状态，

- S_{K0} 表示盘A所在的柱号，取值1, 2, 3
- S_{K1} 表示盘B所在的柱号，取值1, 2, 3

➢ 全部可能的状态，共有以下9种：

- S_0=(1,1), S_1=(1,2), S_2=(1,3),
- S_3=(2,1), S_4=(2,2), S_5=(2,3),
- S_6=(3,1), S_7=(3,2), S_8=(3,3).

➢ 问题的初始状态集合S={S_0}, 目标集合为G={S_4,S_8}



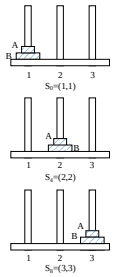
二阶梵塔难题—算符

➢ 算符： A(i,j), B(i,j)

- A(i,j): A盘从柱i移到柱j;
- B(i,j): B盘从柱i移到柱j

➢ 全部可能的算符，共有12种：

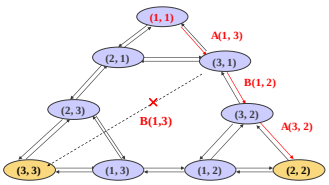
- A(1,2), A(1,3), A(2,1), A(2,3),A(3,1), A(3,2),
- B(1,2), B(1,3), B(2,1), B(2,3), B(3,1), B(3,2)



二阶梵塔难题—状态空间图

➢ 状态空间图

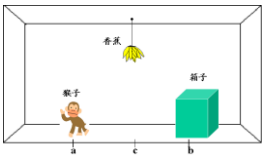
- 从初始节点(1, 1)到目标节点(2, 2)及(3, 3)的任何一条路径都是问题的一个解。
- 最短的路径长度是3，它由3个操作组成。
- 如从(1, 1)开始，通过使用操作A(1, 3)、B(1, 2)及A(3, 2)，可到达(2, 2)。



2.3 二阶梵塔难题的状态空间图

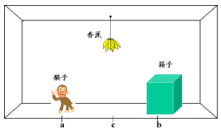
猴子摘香蕉的问题

➢ 问题描述：在一个房间内有一只猴子，一个箱子和一束香蕉。香蕉挂在天花板下方，但猴子的高度不足以碰到它。那么这只猴子怎样才能摘到香蕉呢？



猴子摘香蕉的问题—状态

- 用一个四元表列（W，Y，x，z）表示这个问题状态
  - W 猴子的水平位置
  - x 当猴子在箱子顶上时取x = 1；否则取x = 0
  - Y 箱子的水平位置
  - z 当猴子摘到香蕉时取z=1；否则取z=0
- 初始和目标状态为：
  - S0：（a，b，0，0） 初始状态
  - S3：（c，c，1，1） 目标状态



猴子摘香蕉的问题—操作（算符）

- goto(U)表示猴子走到水平位置U，或者用产生式规则表示为：

(W, Y, 0, 0) —goto(U)→ (U, Y, 0, 0)

- pushbox(V)表示把箱子推到水平位置V，既有：

(W, W, 0, z) —pushbox(V)→ (V, V, 0, z)

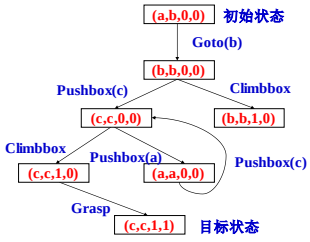
- climbbox猴子爬上箱顶，既有：

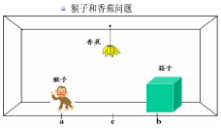
(W, W, 0, z) —climbbox→ (W, W, 1, z)

- grasp猴子摘到香蕉，既有：

(c, c, 1, 0) —grasp→ (c, c, 1, 1)

猴子摘香蕉的问题—状态空间图





解序列为：{Goto(b), Pushbox(c), Climbbox, Grasp}

旅行商问题—状态

旅行商问题 (Traveling-Salesman Problem,TSP)

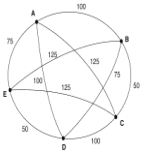
设有n个互相可直达的城市，某推销商准备从其中的A城出发，周游各城市一遍，最后又回到A城。要求为该推销商规划一条最短的旅行路线。

该问题的状态为以A打头的已访问过的城市序列：A…

S_0: A

S_g: $A, \ldots, A$

其中“…”为其余n-1个城市的一个序列。



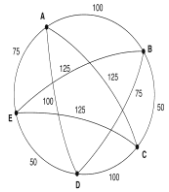
旅行商问题 —算符

- 规则1

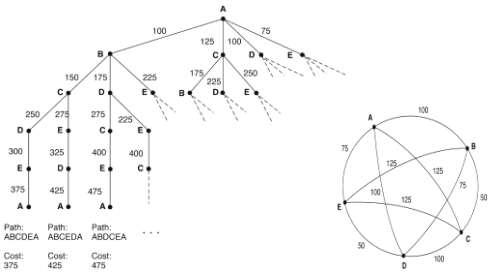
如果当前城市的下一个城市还未去过，则去该城市，并把该城市名排在已去过的城市名序列后端。

- 规则2

如果所有城市都去过一次，则从当前城市返A城，把A也添在去过的城市名序列后端。



旅行商问题 —状态空间图



- 小结

1. 确定采用有几个分量的数组来描述问题的所有可能的状态
2. 确定算符。算符可以是过程如迷宫问题，可以表示走步，如八数码难题，也可以是函数如猴子摘香蕉问题
3. 从问题的初始状态出发如何按条列一条路径，或求优路径到达目标状态