

搜索和问题求解

黄书剑





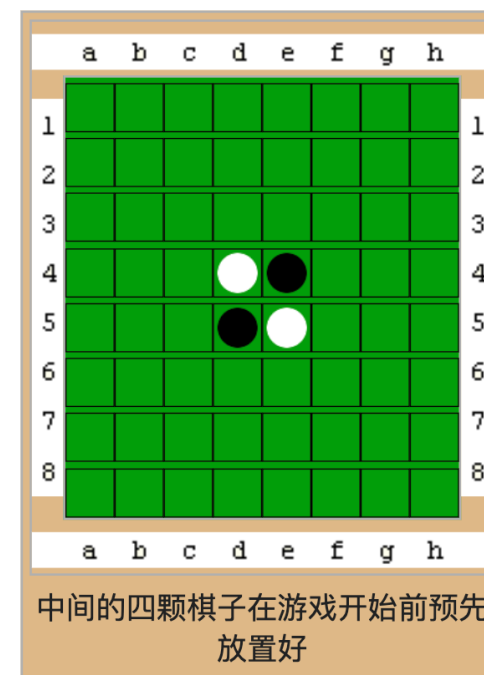
搜索相关的问题（一）

- 求所有的3位水仙花数
 - 每个位上的数字的3次幂之和等于该数本身
 - $1^3 + 5^3 + 3^3 = 153$
- 判断一个数是否为质数
- 求100以内所有的质数
- 哥德巴赫猜想1742
 - 任一大于2的整数都可以写成三个质数之和
 - 任一大于2的偶数都可以写成两个质数之和（欧拉）
- * a+b 问题（殆素数法）
 - 华罗庚, "3+4" "2+3" "1+5" "1+4" 王元、潘承洞, "1+2" 陈景润

求解过程（搜索）：
对一定范围内数字进行
遍历+条件判断

搜索相关的问题（二）

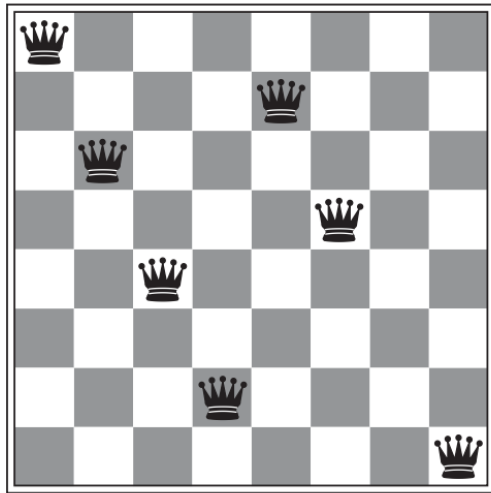
- 黑白棋落子问题（求可能落子位置）
 - 落子和棋盘上任一枚己方的棋子在一条直线上夹着对方棋子
 - 否则不能落子，改由对方落子



求解过程（搜索）：
对一定范围内的状态进行遍历+条件判断

搜索相关的问题（三）

- 八皇后问题
 - 把八个皇后放在棋盘上 (8×8)



求解过程（搜索）：
构造搜索空间进行遍历+条件判断

搜索相关的问题（三）

- 斗地主出牌问题（求可能出牌选择）
 - 牌型规定：
 - 单、对、三、三加一、三加二、顺子、炸弹



求解过程（搜索）：
构造搜索空间进行遍历+条件判断



简单搜索问题

- **确定搜索空间：**
 - 搜索空间较为明确：范围不大、容易穷举
 - 搜索空间不够清晰：构造搜索空间包含所有可能的解
- **搜索方法：**
 - 对搜索空间内的所有状态进行遍历+条件判断
- **优化：**
 - 减少搜索空间大小、提高遍历效率.....



基于知识/规则的条件判断

- 如何判断状态/动作是否合法/可行 (Valid/Applicable) ?
 - 水仙花数的定义
 - 质数的定义
 - 皇后布局的规则
 - 黑白棋的某个落子合法性-黑白棋落子规则
 - 斗地主的某种出牌合法性-斗地主出牌规则
 -

条件判断：
根据知识/规则进行合法性或可行性判断

搜索相关的问题（四）

- 华容道
 - 将曹操移至出口
- 8-puzzle: 将数字有序排列



7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

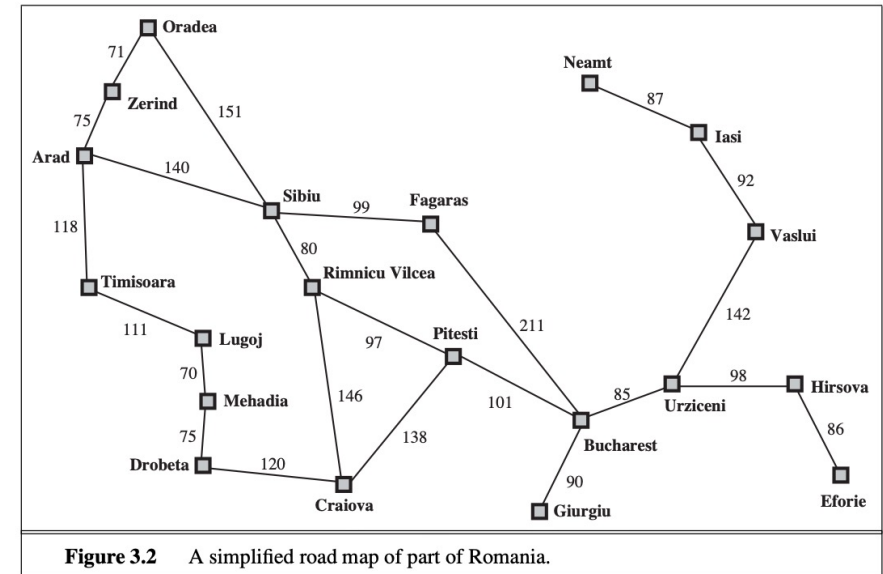
Goal State

求解：需要找到一个能够达到目标状态的动作序列

8-puzzle示例来自于：Artificial Intelligence: A Modern Approach

搜索相关的问题（四）

- 最短路径问题
 - 部分点之间有路径
 - 列表或字典进行存储
 - Arad -> Urziceni
 - Drobeta -> Neamt
 -
- 华容道（最小移动次数）

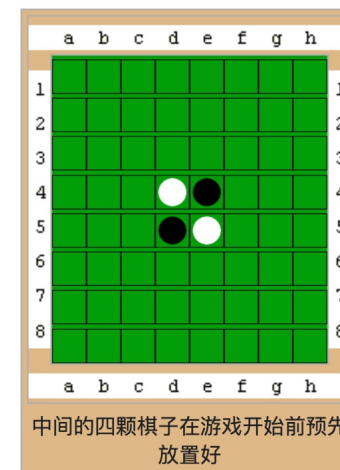


示例来自于：Artificial Intelligence: A Modern Approach

求解：需要找到所有能够达到目标状态的动作序列，并从中找到符合条件的一个序列

搜索相关的问题（五）

- 黑白棋自动对弈
 - 在终局时保有最多棋子
- 斗地主自动对弈
 - 最先出完手牌



求解：需要找到一个能够在博弈中达到目标状态的动作序列（目标达成还取决于对手的行动）



相对复杂的搜索问题

- 搜索空间复杂、难以直接遍历
 - 包含多个动作构成的序列
 - 每个动作都对结果产生影响
- 如何构造一个搜索问题?
 - 初始状态 (initial state)
 - 动作 (action)
 - 判断动作是否可行 (applicable)
 - 状态转换 (state transmission)
 - 穷举所有的可能性?

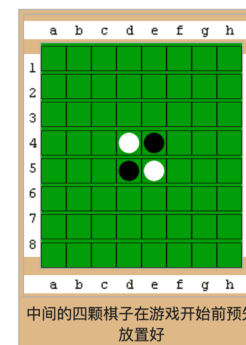
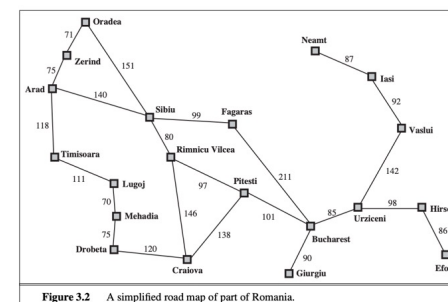


7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State



动作"搜索"

- 一般情况下，存在一个或多个合法的动作：
 - 如：当前状态下，所有落子方法（黑白棋）、所有出牌方法（斗地主）、所有可以移动的城市（最短路径）、所有可以移动的人物（华容道）
- 动作空间一般相对较小，可以通过穷举方式进行搜索
- 动作合法性可以通过知识/规则进行条件判断
- （类似于前述简单搜索问题）

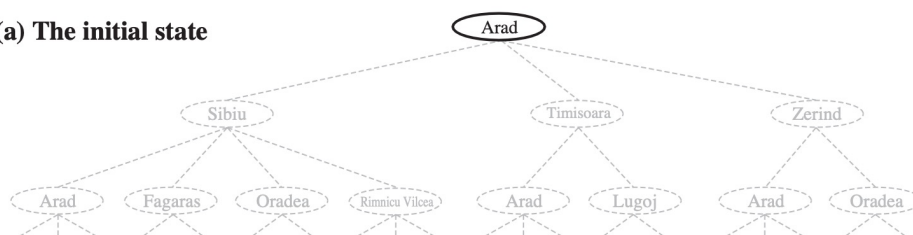
状态搜索

- 从初始状态出发
- 当前状态下，执行某合法动作，状态发生改变 (transmission)
 - 影响动作和合法性的状态
 - 如：当前的对弈格局、当前所处的城市等
 - 影响终局形势的状态 (代价cost/奖赏reward)
 - 如：剩余手牌数、移动步数等
- 重复上条步骤，直到达到终止状态结束
 - 搜索结果：所有经过的状态/动作构成的序列
- 如果存在多条到达终止状态的路径，根据条件进行选择
 - 如：最短移动路径、最少移动步数、最多剩余棋子数等

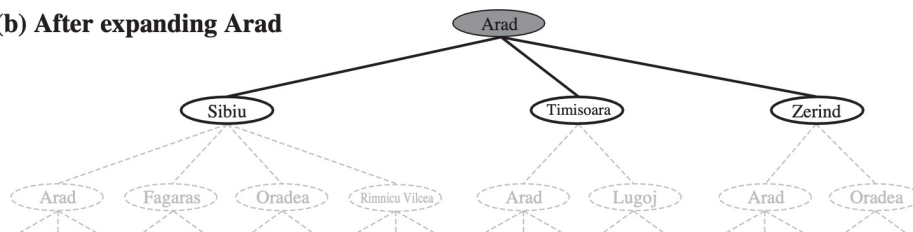
状态搜索示例

- 从给定城市出发进行搜索

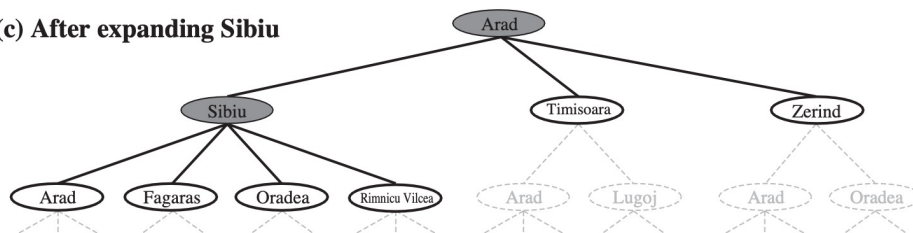
(a) The initial state



(b) After expanding Arad



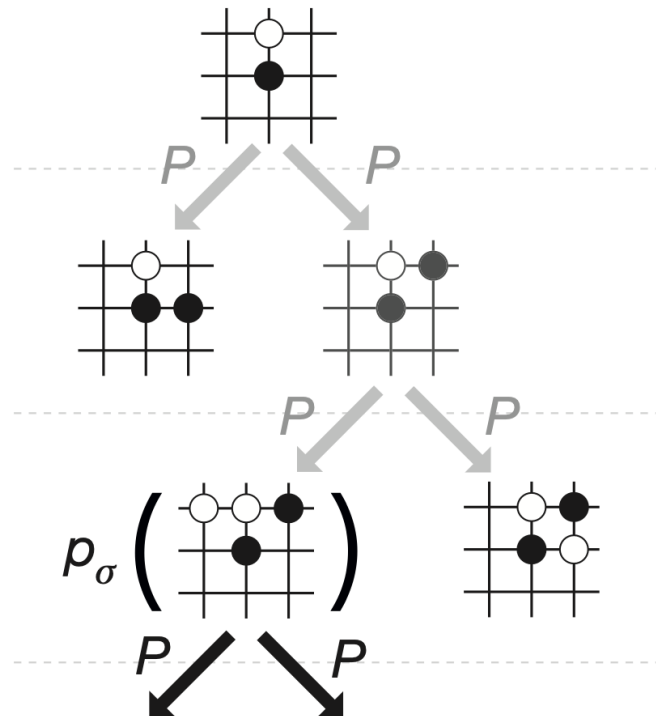
(c) After expanding Sibiu



示例来自于: Artificial Intelligence: A Modern Approach

状态搜索示例

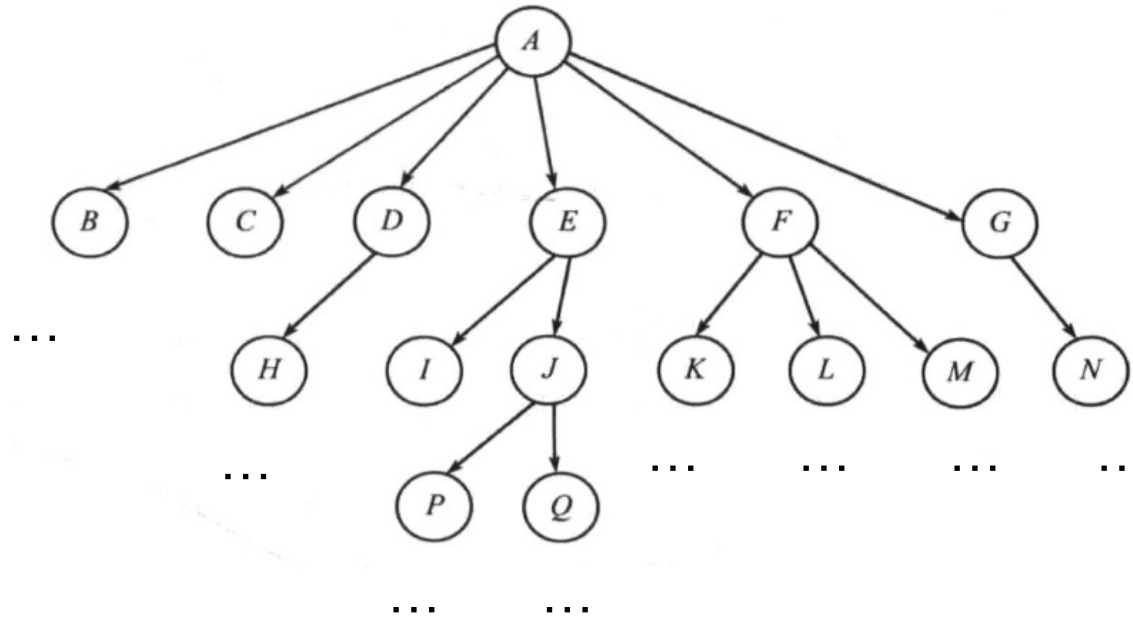
- 从给定状态出发进行搜索



from AlphaGo Paper: Mastering the game of Go with deep neural networks and tree search

状态搜索树

- 通用情形：在一个树形结构中进行搜索
 - A为初始状态，某一个或多个状态为终止状态
 - 每个状态可执行动作不同
 - 执行动作后状态相应发生变化



搜索基本方法

- **Uninformed Search (Blind Search)**
 - 如果能够高效遍历所有可能，则可得到确定的解（穷举）
 - 重点：不重复、不遗漏
 - 设计遍历的方法、顺序（深度优先、广度优先）
- **Informed Search**
 - 每次选择当前看来最好的解（贪心搜索，Greedy Search）
 - 如：吃子最多的落子（黑白棋）、距离最短的城市（最短距离）、出牌最多的走法（斗地主）
 - 不一定能够保证得到正确的解
 - *保证最优选择的方法：A* Search
- **Completeness (可行解)、Optimal (最优解)、时空复杂度**
- **综合利用知识可能会得到更快的搜索速度，比如优先级**

阶段任务



- 使用搜索方法求解一些简单的问题

