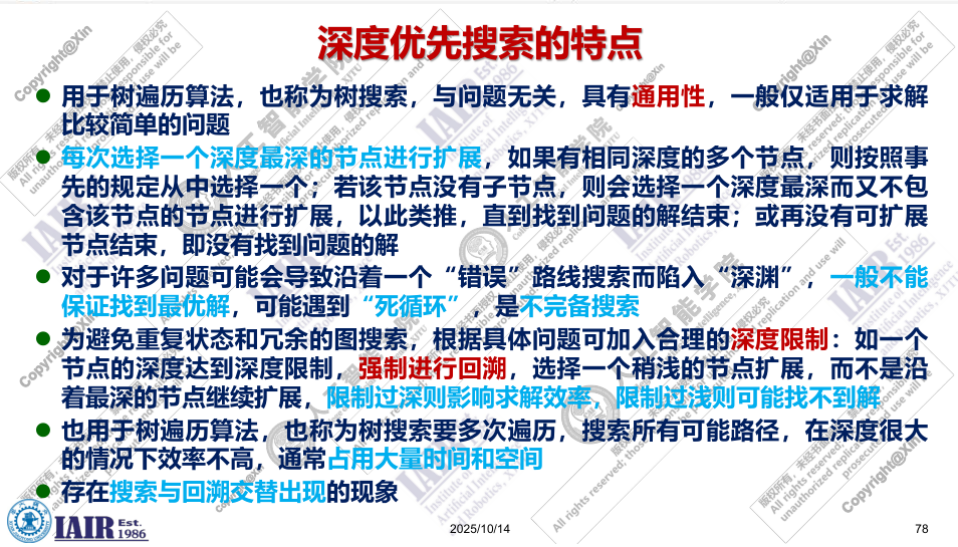
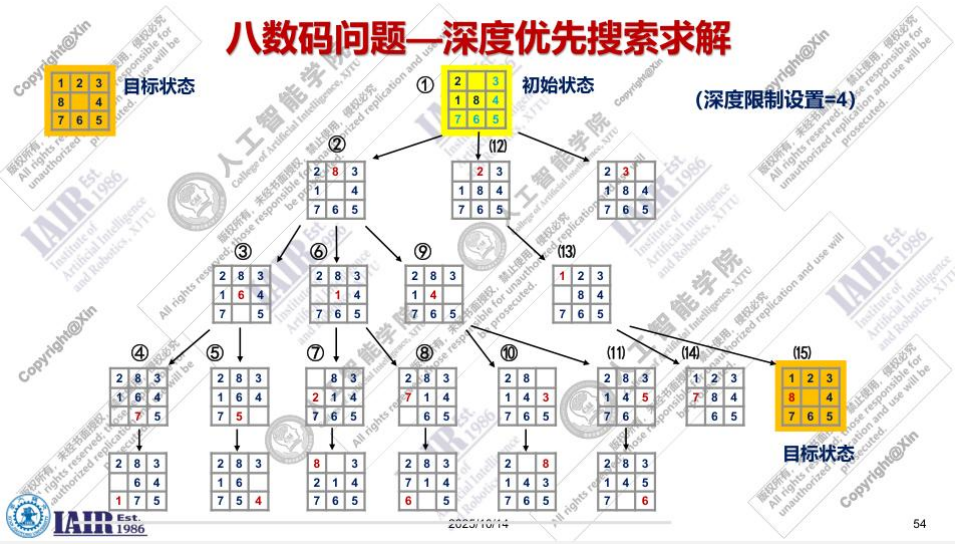
搜索求解

一．盲目搜索（无信息搜索）

1. 深度优先搜索（DFS）



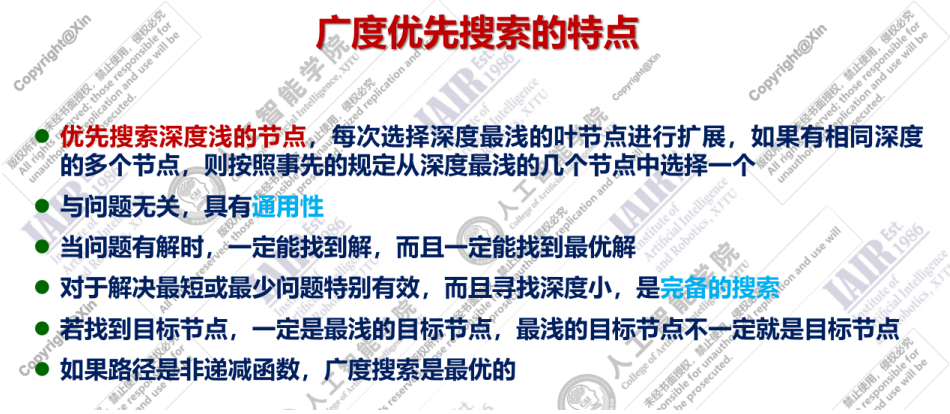


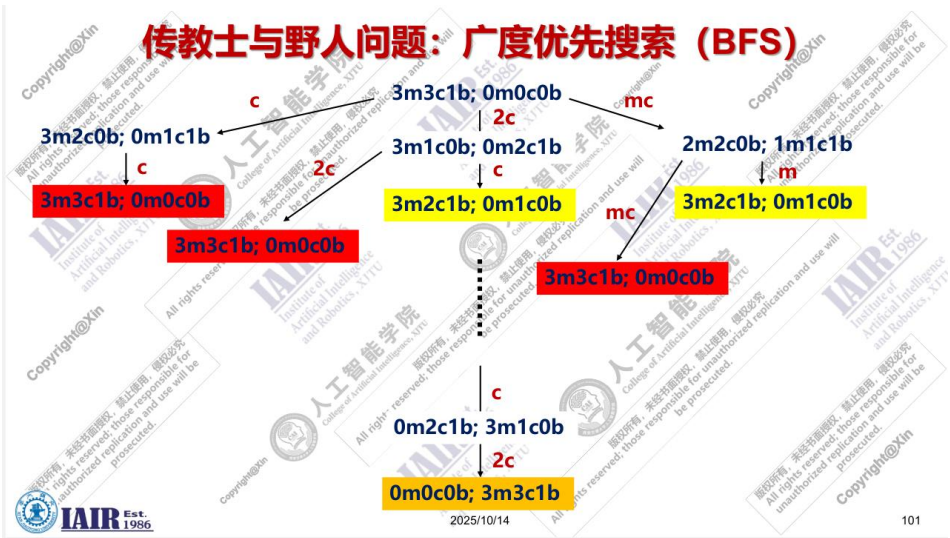
DFS 特例：回溯搜索

特点：不保留完整树结构、适用于没有明确动态规划和递归解法的问题

例子：N 皇后问题

2. 广度优先搜索（BFS）





3. DFS 与 BFS 的对比

深度优先搜索与广度优先搜索比较

	深度优先（DFS）	广度优先（BFS）
完备性	不一定 （若解不在某分支，且该分支又是个无穷分支，则不可能得到解）	完备 （在分支因子b有限情况下）
最优性	不具备	• 最优（如果路径代价是是节点深度的非递增函数的情况下）； • 不一定最优（通常情况下）
时间复杂性	b^m （如果m>b时间复杂度会很高，但如果解决方案密集，可能比BFS快得多）	b^d
空间复杂性	b^m	b^d

b：分支因子；　d：解的深度
m：搜索树的最大深度　l：深度限制

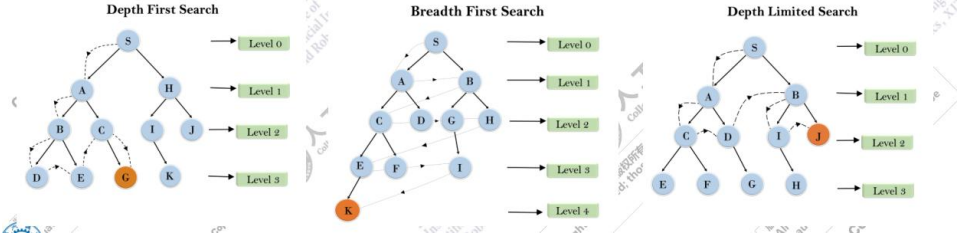
二．启发式搜索（有信息搜索）

1. 一致代价搜索 / Dijkstra 算法

以广度优先搜索为基础，每次扩张代价最小的节点

一致代价搜索（Uniform-cost Search）/Dijkstra算法

- 一致代价搜索：总是扩展路径消耗最小的节点N，N点的路径消耗等于前一节点N-1的路径消耗加上N-1到N节点的路径消耗
- 一致代价搜索是在广度优先搜索上进行扩展的，使用优先级队列并在边缘中的状态发现更小代价的路径时引入的额外的检查，也称为代价一致搜索
- 一致代价搜索实际上是在广度优先搜索的基础上进行扩展，如果每一步的代价全部相等，则相当于广度优先



2. 爬山搜索

每次随机选择一个位置，与邻节点比较并前往更大值。（局部最优解，能否得到全局最优解取决于初值）

3. 贪婪搜索

A. 启发函数和评价函数

启发函数 h(n)评估状态与目标状态的接近程度（越小表示越接近）

已付出代价记为 g(n)

评价函数 f(n)=g(n)+h(n)评价节点的重要程度

B. 贪婪最佳优先搜索

令 f(n)=h(n)，即只考虑与目标状态的接近程度，优先扩展离目标状态最近的节点

1) 局部择优，不是最优搜索

2) 对错误起点敏感，可能有死循环

3) 不完备

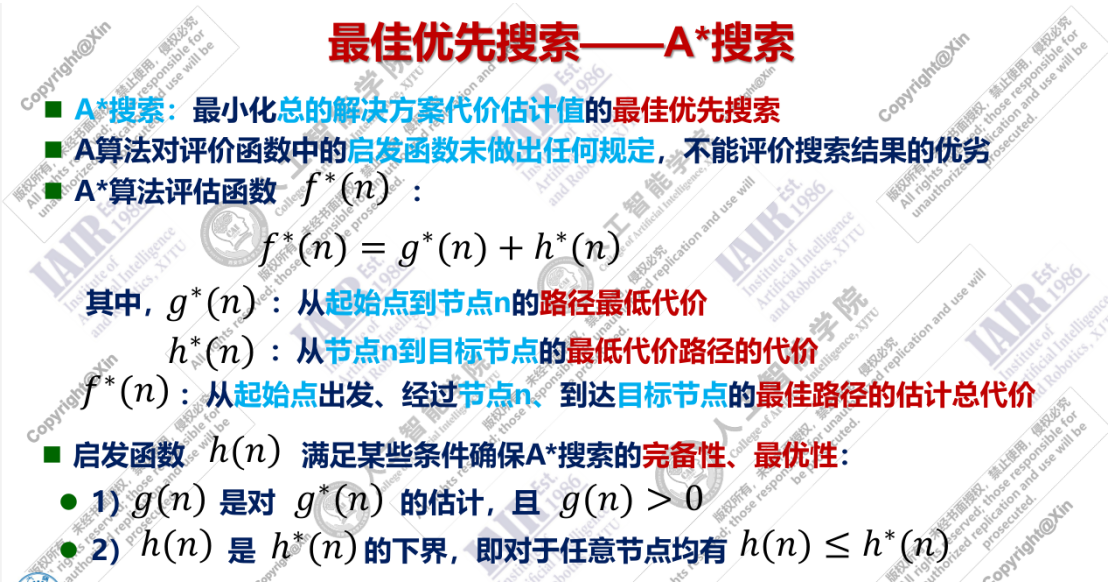
4) 时间/空间复杂度为 O(b^m)

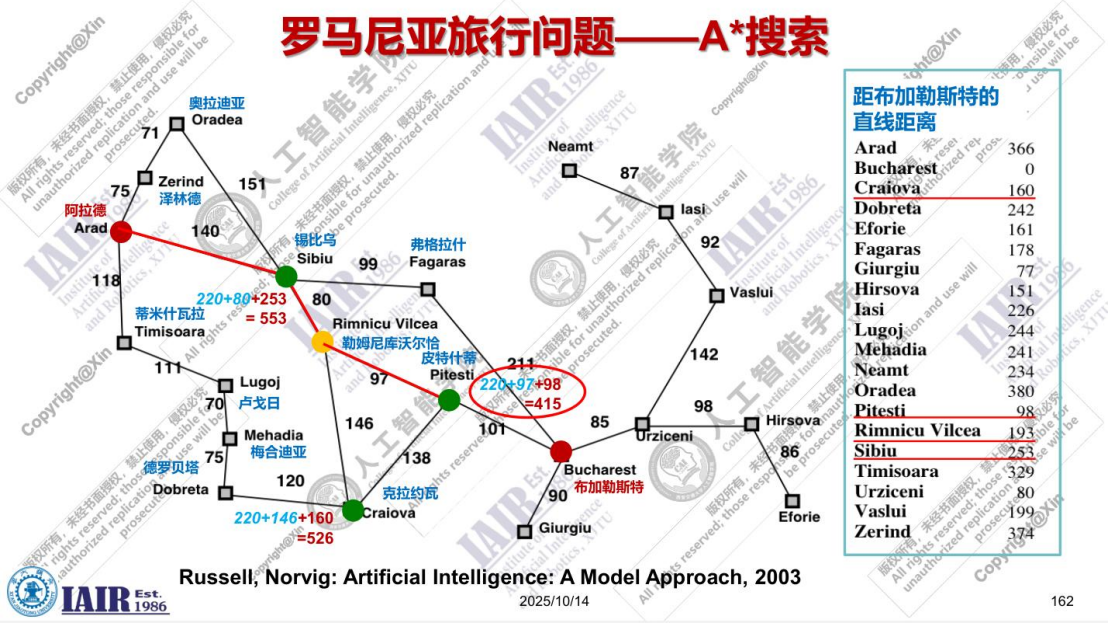
C. A 搜索

令 f(n)=g(n)+h(n)，每次选择具有最小评价函数值的节点扩展

D. A*搜索

对 A 搜索的启发函数 h(n)作了规定：

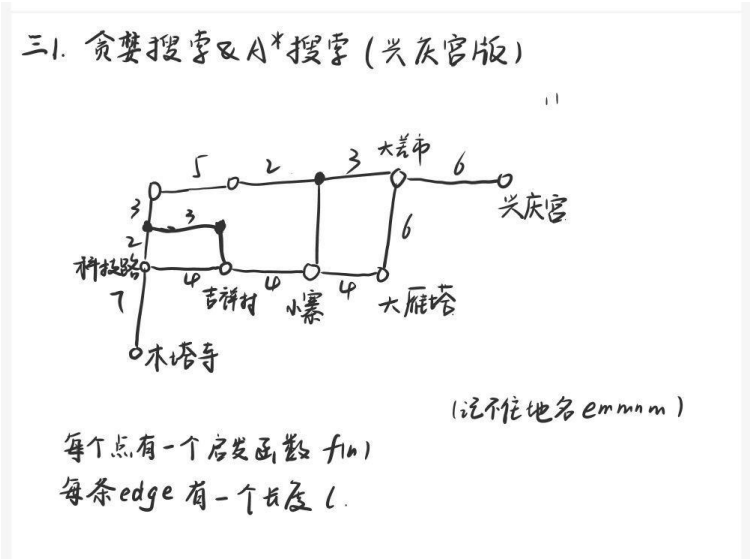




2024 年真题

（选择）盲目搜索包含：深度优先搜索和广度优先搜索

（简答）贪婪搜索和 A*搜索



图并不复杂，可以参考 ppt 上类似题目的求解过程