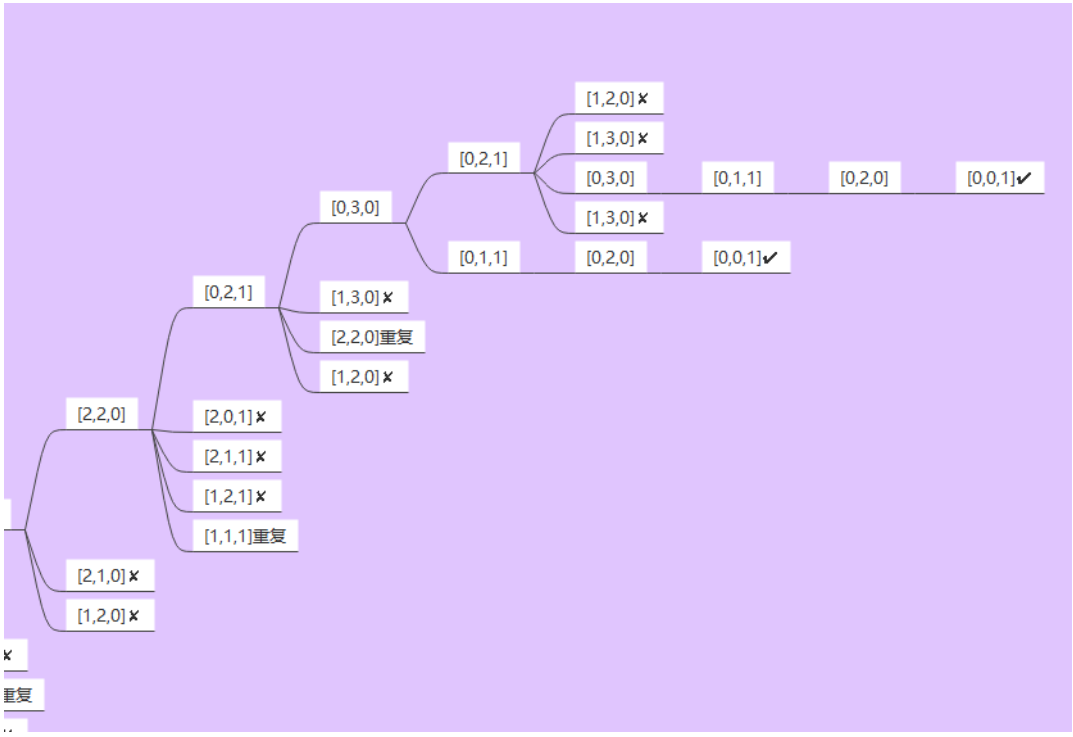
19024075 陶盛皿 Week2 Assignment

1. 问题重述
   在河的左岸有N个传教士、N个野人和一条船，传教士们想用这条船把所有人都运过河去，但有以下条件限制：

（1）修道士和野人都会划船，但船每次最多只能运K个人；
（2）在任何岸边野人数目都不能超过修道士，否则修道士会被野人吃掉。

假定野人会服从任何一种过河安排，请规划出一个确保修道士安全过河的计划。

2. 问题分析
   2.1约束条件：
   ① M≥C 任何时刻两岸、船上都必须满足传教士人数不少于野人数（M=0时除外，既没有传教士）
   ② M+C≤K 船上人数限制在K以内
3. Diagram：见 传教士和野人.png
4. 深度优先搜索程序： 见 DFS-missionaries and cannibals problem.py
   - 不是一个查看所有重复状态的好方法，因为深度优先搜索和过度优先搜索算法都具有完备性，需要遍历所有结点，空间和时间复杂度都是O(bd)。
   - 虽然状态空间简单，但是不同状态和移动情况重复性太多了，在程序中会表现出，例如：



   的循环状态（形如[0,x,1]结构每次运回一个野人[0,x+1,0]，再运去两个野人[0,x-1,1])因为是完备性搜索会增加大量分支，造成空间和时间的浪费（因为太多了，所以我图里几个结点没写完就直接简略到结果了)，所以对人来说很难解决。

---

state：1. 初始状态；2. 转移模型函数的返回装填，result(state, action): 在状态s下执行行动a后达到的状态。

state space：从初始状态，通过行动函数和状态转移函数可以达到的所有状态的集合，是一个有向图或网络结构。状态空间中的一条路径就是指通过行动连接起来的一个状态序列。

search tree：是一种树形数据结构，既有链表的快速插入与删除操作的特点，又有数组快速查找的特点，是agent在解决问题时计算action的结构。

search node：每个结点包括四个元素：node.state（对应状态空间中的状态）、node.parent（该结点的父节点）、node.action（父结点生成该结点时所采取的行动）、node.path-cost（代价或者代价函数，计算路径消耗）。可能的行动序列从搜索树根节点（初始状态）出发，搜索树的连线表示行动，子结点对应状态空间中的状态；子节点作为左右子树的根节点，其子结点又对应执行了可执行的的行动后的状态空间。

goal：在特定应用场景（任务空间）下，agent面临任务环境所要解决的问题，通过机器理性决策并从一组可能的行动中选择形同来试图满足目标。

action：人为pre-programmed的行动集合，即每个状态下可能执行并且是可以执行的行动。

transition model：转移模型是对于每个行动的描述，函数表达式是result(state, action): 在状态s下执行行动a后达到的状态。即等价于后继状态，即在当前状态下执行某一动作达到的新的状态集合。

branching factor：分支因子，是每个结点下出度（最大后继数），即一个状态（根节点）有多少中可采取的行动所能达到的状态（孩子结点）。